

**HCL OneTest™ Server**  
**10.2 Documentation**  
**June 2021**



## Special notice

Before using this information and the product it supports, read the information in [Notices on page mxxiii](#).

# Contents

<b>Chapter 1. Release Notes.....</b>	<b>8</b>
Product description.....	8
What's new.....	8
Installing and upgrading.....	11
Known issues.....	11
Contacting HCL support.....	12
<b>Chapter 2. System Requirements.....</b>	<b>13</b>
Hardware.....	14
Operating systems and Containers.....	14
Host prerequisites.....	15
Supported software.....	16
<b>Chapter 3. Getting Started Guide .....</b>	<b>19</b>
Server overview.....	19
Supported versions of assets and resources.....	20
Tests supported on HCL OneTest Server.....	21
HCL OneTest Data overview.....	23
Task flow: Test runs and results.....	31
Task flow: Server access from desktop clients.....	33
Accessibility features.....	35
<b>Chapter 4. Administrator Guide.....</b>	<b>36</b>
Installation of the server software.....	36
Installation of the server software on Red Hat OpenShift.....	37
Installation of the server software on Ubuntu.....	58
Installation of the server software on Azure Kubernetes Service.....	71
Server software upgrade methods.....	82
Server software upgrade on Red Hat OpenShift.....	82
Server software upgrade on Ubuntu.....	87
Backup and restoration of the server data.....	88
Backup and restoration of the server data on Red Hat OpenShift.....	89
Backup and restoration of the server data on Ubuntu.....	91
Backup and restoration of the server data on Azure Kubernetes Service.....	96
Uninstallation of the server software.....	100
Uninstalling the server software from Red Hat OpenShift.....	101
Uninstalling the server software from Ubuntu.....	102
Uninstalling the server software from Azure Kubernetes Service.....	103
Configuration of the server software.....	104
User administration.....	104
Certificate authority: Importing and extending lists.....	116
Copying third-party application Jars to Kubernetes.....	119
Changing the password seed.....	122
Server extensions.....	124
Team space administration.....	128
Team space overview.....	128
Creating a team space.....	131
Tasks in a team space.....	132

Tasks in projects in a team space.....	136
HCL license portal.....	138
<b>Chapter 5. Test Author Guide.....</b>	<b>147</b>
Datasets overview.....	147
Creating a dataset.....	148
Editing a dataset.....	149
Publishing a dataset.....	157
Viewing a dataset.....	158
Deleting a dataset.....	160
Generation of test data.....	161
Schema fabrication.....	161
Schema design overview.....	170
Test data generation from structured schemas.....	190
Status of generated test data.....	237
<b>Chapter 6. Test Execution Specialist Guide.....</b>	<b>244</b>
System modeling.....	244
Task flows for working with a system model.....	247
Adding a repository to a team space.....	248
Creating a system model.....	251
Creating components.....	252
Creating child components.....	253
Creating linkages between components.....	255
Deleting linkages between components.....	257
Modifying components.....	259
Deleting components.....	260
Publishing changes to the system model.....	261
Associating resources with components.....	262
Viewing resources associated with components.....	263
Removing resources associated with components.....	267
Publishing changes of associating resources with components.....	268
Viewing the system model.....	269
Deleting a system model.....	270
Deleting a repository that is added to a team space.....	270
Prerequisites to running tests.....	271
Test run considerations for AFT Suites.....	272
Test run considerations for API Suites.....	273
Test run considerations for schedules.....	276
Considerations for using Jaeger traces in reports.....	277
Test run considerations for JMeter tests.....	278
Test run considerations for JUnit tests.....	279
Test run considerations for Postman tests.....	280
Management of agents.....	282
Test run considerations for running tests on remote agents.....	282
Viewing agents that are registered with HCL OneTest™ Server.....	283
Adding an agent to a project.....	284
Working with agent capabilities.....	286
Management of Docker hosts.....	288



Test run considerations for running tests on remote Docker hosts.....	289
Setting up a remote Docker host.....	291
Setting up a secure remote Docker host.....	292
Copying third-party application Jars to a remote Docker host.....	294
Registering a remote Docker host.....	296
Viewing remote Docker hosts that are registered with HCL OneTest™ Server.....	298
Adding a remote Docker host.....	299
Editing configurations.....	300
Deleting a remote Docker.....	302
Unregistering a remote Docker host from HCL OneTest™ Server.....	304
Test run configurations.....	305
Configuring a test for a quick run.....	305
Configuring an AFT Suite run.....	306
Configuring an API Suite run.....	314
Configuring a run of a Compound Test that contains HTML tests.....	322
Configuring a run of a Compound Test that contains Web UI tests.....	330
Configuring a run of a Compound Test that contains mobile tests.....	339
Configuring a run of a Compound Test that contains performance tests.....	347
Configuring a JMeter test run.....	355
Configuring a JUnit test run.....	359
Configuring a Postman test run.....	364
Configuring a run of a Rate Schedule or VU Schedule.....	368
Running tests by using Data Fabrication.....	376
Management of running tests.....	378
Viewing the state of test assets.....	378
Viewing the progress of running test assets.....	381
Checking logs.....	384
Resetting the configuration settings for a test run.....	385
Stopping test runs.....	386
Canceling scheduled test runs.....	389
Management of virtualized services.....	390
Prerequisites for running virtual services.....	392
Viewing intercepts that are registered with a team space on HCL OneTest™ Server.....	397
Viewing virtual service resources.....	398
Configuring a run of a virtual service.....	401
Running HTTP virtual services without using proxies.....	413
Viewing running instances of virtual services.....	418
Viewing configurations of running instances of virtual services.....	423
Modifying configurations of running instances of virtual services.....	428
Viewing routing rules of the virtual services.....	430
Viewing usage statistics of virtual services.....	431
Stopping virtual services.....	434
Test results.....	435
Test results and reports overview.....	436
Resource monitoring service.....	446
Resource monitoring capabilities.....	449
Resource monitoring agents.....	462

Managing sources.....	472
Configuration of a change management system.....	479
Configuration of Jira as a change management system.....	480
Integrating with other applications.....	486
Integration plugin compatibility matrix.....	486
Integration with Azure DevOps.....	486
Integration with HCL® Launch.....	494
Integration with Jenkins.....	514
Integration with UrbanCode Deploy.....	524
Integration with other applications.....	544
Managing access to the server .....	565
<b>Chapter 7. Test Manager Guide.....</b>	<b>567</b>
Team space management.....	567
Viewing team spaces.....	567
Adding members to a team space.....	568
Modifying the configuration of a team space .....	571
Viewing the configuration of a team space .....	573
Becoming a team space member.....	575
Managing members and their roles in a team space.....	576
Deleting a team space.....	579
Test assets and a server project.....	580
Working with projects in a team space.....	580
Repository considerations for a server project.....	581
Adding a project.....	582
Adding repositories to a server project.....	583
Secrets configuration .....	585
Adding users to a project.....	585
Viewing projects.....	586
Becoming a project member.....	587
Managing access to server projects.....	588
Archiving or unarchiving server projects.....	590
Deleting server projects.....	591
Project overview.....	591
Managing repositories.....	592
Adding repositories to a server project.....	593
Updating the authentication credentials of the repository.....	594
Deleting a repository.....	595
Refreshing repositories manually.....	595
Selecting the global branch in a project.....	597
Creating webhooks.....	598
Protecting API test assets by using secrets.....	598
Managing secrets collections.....	599
Granting access to members or member roles.....	602
Creating a secret in a secrets collection.....	601
Managing secrets.....	604
Managing an encrypted dataset.....	605
Creating a classification.....	605

Editing or deleting a classification.....	606
Moving an encrypted dataset to another classification.....	608
Removing a dataset from the classification.....	608
Granting classification access to members or members roles.....	609
Managing notifications.....	611
Managing in-app notifications.....	612
Managing email notifications.....	617
<b>Chapter 8. Reference Guide.....</b>	<b>624</b>
Files.....	624
Connections.....	624
Project-level connections.....	624
Connection testing.....	624
Resource adapters.....	624
Schema design.....	631
Introduction to schema design.....	631
Type properties.....	633
Item restrictions.....	694
Components.....	696
Partitioning.....	705
Type inheritance.....	710
Schema analyzer.....	713
Distinguishable objects.....	714
Functions and expressions.....	733
<b>Chapter 9. Troubleshooting Guide.....</b>	<b>987</b>
Troubleshooting issues.....	987
Troubleshooting issues.....	996
Schema designing error or warning messages.....	996
Compile-time error messages.....	1014
Configuring log files.....	1016
Audit log overview.....	1016
Commands used in HCL OneTest Data.....	1018
Troubleshooting installation issues.....	1019
Security Considerations.....	mxxi
Notices.....	mxxiii
Index.....	1027

# Chapter 1. Release notes for HCL OneTest™ Server

This document includes information about What's new, installation and upgrade instructions, known issues, and contact information of HCL Customer Support.

## Contents

- [Product description on page 8](#)
- [What's new on page 8](#)
- [Installing and upgrading on page 11](#)
- [Known issues on page 11](#)
- [Contacting HCL support on page 12](#)

## Product description

You can find the description of HCL OneTest™ Server.

HCL OneTest™ Server is a server that includes capabilities such as project and role-based security, Docker-based distribution and installation, and running of test cases. For more information about the server, see [HCL OneTest Server overview on page 19](#).

## What's new

You can find information about the features introduced in this release of HCL OneTest™ Server.

The following sections list the new features, enhancements or other changes made in this release.

### • **Installing server software on Azure Kubernetes Service**

You can now install HCL OneTest™ Server on Azure Kubernetes Service that has a Kubernetes environment to run functional, integration, and performance tests. See [Installation of the server software on Azure Kubernetes Service on page 71](#).

### • **Creating and maintaining multiple team spaces**

Previously, HCL OneTest™ Server supported only one logical partition as the initial team space. Now, you can create and maintain multiple team spaces in HCL OneTest™ Server. The initial team space is available for all licensed users of HCL OneTest™ Server before you create any other team space.

You can perform the following tasks in team spaces:

- Configure licenses or licensed users for each team space.
- View all team spaces created on HCL OneTest™ Server and the team spaces that you can access from the **Team Space Dashboard** page.
- Register Dockers, agents, intercepts, or Resource Monitoring agents with a team space before you use them in your projects of that team space.
- Add team space members with the following roles:

- *Team Space Owner*
- *Project Creator*
- *Architect*
- *Member*
- Isolate the projects from the users who are not members of the team space.

See [Team space overview on page 128](#).

- **Configure a repository with any default branch in a team space**

Now, you can configure any branch in a repository as the default branch in a team space.

- **Associating test suites to the components of the System Model**

You can now associate test suites to the components of the System Model in addition to the virtual resources. After you associate the test suite, you can then click the test suite to view them on the **Execution** page.

See [Viewing resources associated with components](#).

- **Starting or stopping virtual services associated with components**

After you associate virtual services with components in the **System Model** page, you can then navigate to the **Resources** or **Instances** page to either start an instance or stop a running instance.

See [Viewing resources associated with components](#).

- **Viewing and filtering test assets by their associated components on the Execution page**

The components in the **System Model** page that are associated with test assets can now be viewed in the **Components** column on the **Execution** page. You can also use the component as a parameter in a filter rule to obtain test assets associated with a specific component. You can return to the **System Model** page by clicking the component displayed for the test asset in the **Components** column on the **Execution** page.

- **Viewing and stopping virtual service instances associated with components**

After you start an instance of the virtual services that are associated with components in the **System Model** page, you can then navigate to the **Instances** page to either view the running instance or stop a running instance. You can also view the components that are associated with running instances of virtual services from the **Instances** page.

See [Viewing running instances of virtual services on page 418](#).

- **Introducing the Concurrent Virtual Services licenses**

Now, you need a separate Concurrent Virtual Services license to run an instance of the virtual services on HCL OneTest™ Server.

See [Server software licensing information on page 139](#).

- **Viewing running instances associated with components**

After you have associated virtual services with components in the System Model and started instances, you can view the instances that are associated with the components from the **Instances** page. You can also filter the view to display running instances by using components as a criteria in the filters.

- **Hiding inactive instances from the virtual service Instances page view**

You can now hide the virtual service instances that are stopped or those that failed, from the display on the virtual service **Instances** page via a single button.

See [Viewing running instances of virtual services on page 418](#).

- **Stopping multiple running instances of virtual services**

You can now stop multiple or all running instances of virtual services from the **Instances** page.

See [Stopping virtual services on page 434](#).

- **Navigating to the Progress page to view the state of test asset runs**

You can now navigate to the **Progress** page from the **Execution** page to view the runs that were started, scheduled, or completed by clicking the **Show in the Progress page** icon that is enabled for the specific test asset.

See [Viewing the state of test assets on page 378](#).

- **Stopping or Canceling multiple test runs**

You can now stop multiple or all running test assets from the **Progress** page. You can also cancel multiple or all scheduled runs of test assets from the Progress page.

See [Stopping test runs on page 386](#) or [Canceling scheduled test runs on page 389](#).

- **Defining resource monitoring sources for a team space**

As a team space owner, you can define resource monitoring sources in the team space so that they can be shared between multiple projects in the team space. During a performance schedule execution with resource monitoring labels, the sources defined in a team space are collected with the sources that are defined in a project whenever they match the provided labels. As a user of a team space, you can use the resource monitoring sources that are defined in the team space.

- **Agents listed for selection based on capabilities**

When you install static agents and add the agents to your project on HCL OneTest™ Server, the agents based on the capabilities that match with the agent specified in the test assets are listed for an override. The agents with the best matching capabilities are listed on the top followed by other agents with mismatched capabilities.

- **Adding categories and properties to user-defined capabilities**

The user-defined capabilities feature is enhanced. Now, you can group user-defined capabilities under categories that you create. You can also add name-value pairs as properties for the capabilities.

See [Working with agent capabilities on page 286](#).

The following sections list the new features, enhancements or other changes made in HCL® OneTest™ Data:

- **Establishing relationship among schemas based on referential integrity**

Previously, you were able to only view the relationships among JDBC-supported schemas on the Schema Canvas page. You can now establish parent-child relationships among multiple non-JDBC schemas on the Schema Canvas page.

See [Establishing a parent-child relationship among schemas](#).

- **Generating DDL statements for multiple schemas**

You can now generate the Data Definition Language (DDL) statements for single or multiple non-JDBC schemas. You can use the DDL statements to create a table in your database. Each job of generating DDL statements for the selected schemas generates a SQL file.

See [Generating DDL statements for schemas on page 231](#).

- **Generating the test data by using multiple schemas**

Previously, you were able to generate the test data for only multiple JDBC-supported schemas. You can now generate the test data for multiple non-JDBC schemas by maintaining referential integrity. You can also modify the relationships among non-JDBC schemas on the **Schema Canvas** page.

You can view the relationship among the selected schemas on the **Schema Canvas** page.

See [Generating test data for selected schemas on page 228](#).

## Installing and upgrading

You can find information about the installation and upgrade instructions for HCL OneTest™ Server.

### Installing HCL OneTest™ Server

For instructions about installing the software, see [Installation of the server software on page 36](#).

### Upgrading of HCL OneTest™ Server

For instructions about upgrading the software, see [Server software upgrade methods on page 82](#).

## Known issues

You can find information about the known issues identified in this release of HCL OneTest™ Server.

**Table 1. Download documents and technotes**

Product	Download document	Knowledge Base
HCL OneTest™ Server	<a href="#">Release document</a>	<a href="#">HCL support</a>

The knowledge base is continually updated as problems are discovered and resolved. By searching the knowledge base, you can quickly find workarounds or solutions to problems.

## Contacting HCL support

- For technical assistance, contact [HCL Customer Support](#).
- Before you contact HCL support, you must gather the background information that you might need to describe your problem. When you describe a problem to the HCL support specialist, be as specific as possible and include all relevant background information so that the specialist can help you solve the problem efficiently. To save time, know the answers to these questions:
  - What software versions were you running when the problem occurred?
  - Do you have logs, traces, or messages that are related to the problem?
  - Can you reproduce the problem? If so, what steps do you take to reproduce it?
  - Is there a workaround for the problem? If so, be prepared to describe the workaround.



# Chapter 2. System Requirements

This document includes information about hardware and software requirements for HCL OneTest™ Server.

## Contents

- [Hardware on page 14](#)
- [Operating systems and Containers on page 14](#)
  - [Operating systems on page 15](#)
  - [Containers on page 15](#)
- [Host prerequisites on page 15](#)
  - [Licensing on page 16](#)
  - [Web browsers on page 16](#)
- [Supported software on page 16](#)
  - [Development tools on page 17](#)
  - [Integrations on page 17](#)
  - [Runtime environment on page 17](#)
- [Disclaimers on page 13](#)

## Disclaimers

This report is subject to the Terms of Use and the following disclaimers:

The information contained in this report is provided for informational purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this publication, it is provided AS IS without warranty of any kind, express or implied, including but not limited to the implied warranties of merchantability, non-infringement, and fitness for a particular purpose. In addition, this information is based on HCL's current product plans and strategy, which are subject to change by HCL without notice. HCL shall not be responsible for any direct, indirect, incidental, consequential, special or other damages arising out of the use of, or otherwise related to, this report or any other materials. Nothing contained in this publication is intended to, nor shall have the effect of, creating any warranties or representations from HCL or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of HCL software.

References in this report to HCL products, programs, or services do not imply that they will be available in all countries in which HCL operates. Product release dates and/or capabilities referenced in this presentation may change at any time at HCL's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. Discrepancies found between reports and other HCL documentation sources may or may not be attributed to different publish and refresh cycles for this tool and other sources. Nothing contained in this report is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in any specific sales, revenue growth, savings or other results. You assume sole responsibility for any results you obtain or decisions you make as a result of this report.

Notwithstanding the Terms of Use users of this site are permitted to copy and save the reports generated from this tool for such users own internal business purpose. No other use shall be permitted.

## Hardware

You can find information about the hardware requirements for HCL OneTest™ Server.

Hardware	Platform	Requirement	Notes
Disk space	Container	256 GB	Additional resources are required by the cluster for the execution of test assets.
Memory	Container	32 GB	Additional resources are required by the cluster for the execution of test assets.
Processor	Container	8 CPUs	Additional resources are required by the cluster for the execution of test assets.
Network	Container	Gigabit (1000) Ethernet	
Other hardware	Container		Operating system support is specified by the supported platforms: Kubernetes and OpenShift. Refer to their associated documentation for a list of supported operating systems.

Related information

[System Requirements on page 13](#)

## Operating systems and Containers

You can find details about the supported operating systems and containers.

### Contents

- [Operating systems on page 15](#)
- [Containers on page 15](#)

### Bit version support

Different parts of a product might run on the same operating system but support different application bitness.

For example, one part of the product might run only in 32-bit mode, whereas another might support 64-bit tolerate mode.

Bitness	Description
64-Tolerate	The product or part of the product runs as a 32-bit application in the 64-bit platforms listed as supported.
64-Exploit	The product or part of the product runs as a 64-bit application in the 64-bit platforms listed as supported.

## Operating systems

Operating system	Hardware	Bitness	HCL OneTest™ Server	Notes
Ubuntu Server 18.04 LTS	x86-64	64-Exploit	✓	Running K3s v1.20.6
Ubuntu Server 20.04 LTS	x86-64	64-Exploit	✓	Running K3s v1.20.6

## Containers

You can find details about the supported containers.

Container	Runtime	Version	Notes
HCL OneTest™ Server	OpenShift Container Platform	4.5	The Container Platform that is supported by OpenShift V4.5.24.

Related information

[System Requirements on page 13](#)

## Host prerequisites

You can find the prerequisites that support the operating capabilities for HCL OneTest™ Server.

## Contents

- [Licensing on page 16](#)
- [Web browsers on page 16](#)

## Licensing

License Server	Version	HCL OneTest™ Server
FlexNet Operations LLS	2017.02	✓
	2021.05	✓

## Web browsers

Browser	Version	HCL OneTest™ Server
Apple Safari	13 or later	✓
Google Chrome	78 or later	✓
Microsoft Edge	80 or later	✓
Microsoft Edge Chromium	86 or later	✓
Mozilla Firefox (including ESR versions)	68 or later	✓

Related information

[System Requirements on page 13](#)

## Supported software

You can find details about the additional software that are supported.

### Contents

- [Development tools on page 17](#)
- [Integrations on page 17](#)
- [Runtime environment on page 17](#)

## Development tools

Supported software	Version	HCL OneTest™ Server
Git	2.19.0	✓
	2.21.0	✓
	2.22.0	✓

## Integrations

Supported software	Version	HCL OneTest™ Server
Azure	Kubernetes 1.19.9	✓
Azure Plugins	Cloud version, latest	✓
HCL® Launch	7.1	✓
Jenkins	2.227	✓
Jira	8.16	✓

## Runtime environment

Supported software	Version	HCL OneTest™ Server	Notes
Helm CLI	3.5.4	✓	Required locally for Helm.
	3.5.2	✓	
Jaeger Operator		✓	Required if tests contribute trace information. The ver-

Supported software	Version	HCL OneTest™ Server	Notes
			sion depends on the target platform.
OpenShift Container Platform CLI (oc)	4.6	✓	Required locally for OpenShift Container Platform.
	4.5	✓	
OpenShift Container Platform (OCP)	4.5	✓	The Container Platform that is supported by OpenShift V4.5.24.
OpenShift Container Storage (OCS)	4.5	✓	The storage class must support ReadWriteMany (RWX) so that executions can be performed on all nodes.

---

Related information

[System Requirements on page 13](#)

# Chapter 3. Getting Started Guide

This guide provides an overview of HCL OneTest™ Server. You can find the task flows to get you started with HCL OneTest™ Server. This guide is intended for new users.

## HCL OneTest™ Server overview

HCL OneTest™ Server brings together test data, test environments, and test runs and reports into a single, web-based browser for testers and non-testers.

HCL OneTest™ Server provides the following capabilities:

### **Web-based continuous testing platform**

HCL OneTest™ Server is a web-based continuous testing platform built on modern, cloud native technologies that enables test teams to run a breadth of tests that includes API, functional, and performance tests and to benefit from a holistic view of test progress. HCL OneTest™ Server is built on Docker for easy deployment. You can scale your testing on the cloud or remote on premise systems with native Docker, Kubernetes, and IBM Red Hat OpenShift support.

### **Role-based access and security**

Security is a key concern for HCL clients and therefore, HCL OneTest™ Server brings a comprehensive, role-based access control scheme to the server with project owners assigning specific member's key permissions (by using roles), for example managing test data or working with secrets such as passwords.

### **Running of tests from the server by using Docker containers**

Server-based running of tests is the starting point for HCL OneTest™ Server. For members of a project with the appropriate role, HCL OneTest™ Server enables direct running of tests from the browser by using transient Docker containers.

### **Connected agents for existing performance agents**

Agent owners can connect existing performance agents to the server and add them to a project for running schedules and Accelerated Functional Testing (AFT) Suites on the current infrastructure.

### **Project overview statistics**

The **Overview** page for HCL OneTest™ Server offers you a quick, simple view on the state of testing for your projects.

### **Project home page**

The home page lists your projects and other projects, which makes it easy to manage different projects within an organization.

### **Reporting and the Resource Monitoring Service**

HCL OneTest™ Server provides the home for capabilities that previously were hosted on HCL® Quality Server. Reporting and the Resource Monitoring Service are in HCL OneTest™ Server and provide a more

direct relationship with their related projects. These capabilities also benefit from the project level, role-based access controls. You can view unified test results to help you make informed business decisions.

### Test data authoring

Beyond the concept of a project held in a Git repository for a simple location of tests and related assets, you can do full concurrent editing of test data sets directly from HCL OneTest™ Server. This true multiuser capability enables team members to collaborate more easily as well as try out data changes without impacting the rest of the team. When satisfied with the results, team members can push their changes. You can also fabricate sample data to perform tests by using the data generator supported in HCL OneTest™ Server.

### Integration with DevOps tools

You can integrate HCL OneTest™ Server with various popular DevOps tools like Jenkins, UrbanCode™ Deploy, or HCL® Launch and Microsoft Azure DevOps to get more value from your DevOps pipelines. You can use UrbanCode™ Deploy, or HCL® Launch to define a deployment process that automatically triggers test cases and have those test insights available directly within UrbanCode Velocity. With the Microsoft Azure DevOps integration, you can also define a DevOps pipeline that includes direct execution of tests by using the HCL OneTest™ Server scalable infrastructure. HCL OneTest™ Server also integrates with Jira for defect tracking and GitHub for software source management and version control.

## Supported versions of assets and resources

You can find information about the versions of assets, resources, agents and Dockers that are supported in HCL OneTest™ Server. You can also find information about the versions of the desktop clients that are supported.

The following table lists the versions of assets or resources created in the desktop clients along with the agents, Dockers, or proxies that are supported in HCL OneTest™ Server:

Resource	Supported	Not supported
Assets or resources created in HCL OneTest™ UI	Assets or resources created in HCL OneTest™ UI which are of versions earlier or the same as HCL OneTest™ Server.	Assets or resources created in any later version of HCL OneTest™ UI that is later than the version of HCL OneTest™ Server.
Assets or resources created in HCL OneTest™ API	Assets or resources created in HCL OneTest™ API which are of versions earlier or the same as HCL OneTest™ Server	Assets or resources created in any later version of HCL OneTest™ API that is later than the version of HCL OneTest™ Server.
Assets or resources created in HCL OneTest™ Performance	Assets or resources created in HCL OneTest™ Performance which are of	Assets or resources created in any later version of HCL OneTest™ Per-



Resource	Supported	Not supported
	versions earlier or the same as HCL OneTest™ Server	formance that is later than the version of HCL OneTest™ Server.
HCL OneTest™ Performance Agents	HCL OneTest™ PerformanceAgents which is the same version as HCL OneTest™ Server	Before you install the agents that are of the current version, see <a href="#">Management of agents on page 282</a> .
Dockers on a remote host computer	Image of HCL OneTest™ Server which is the same version as HCL OneTest™ Server that you want to use to run tests, must be installed on the Docker on a remote host computer.	Before you set up Dockers remotely, see <a href="#">Management of Docker hosts on page 288</a> .
HTTP proxies	Installation of the proxy component from HCL® Quality Server which is the same version as HCL OneTest™ Server.	Before you use HTTP proxies, see <a href="#">Setting up the HTTP proxy on page 395</a> .

If there is a mismatch in the versions of the assets, resources, agents, or images used in Dockers with the version of HCL OneTest™ Server that you are using, any of the following events can occur:

- Warnings or Errors are displayed when you add a repository to a project in your team space.
- Warnings or Errors are displayed when you open a project that you migrated from a previous version of HCL OneTest™ Server.
- Warnings or Errors are displayed when you add or manually refresh a repository to which you added assets or resources that were created on a desktop client with a version previous to or later than the version of HCL OneTest™ Server that you want to use.
- Test runs that you start or schedule on an agent or Docker fail to run.
- You cannot view the proxies that are registered with HCL OneTest™ Server on the **Agents and Intercepts** page.

## Tests supported on HCL OneTest™ Server

You can find information about the types of tests that you can configure for a test run in a project on HCL OneTest™ Server. You can also find information about the desktop clients in which you can create the test.

The following table lists the test types that are supported and those tests that are not supported for running on HCL OneTest™ Server:

Product	Supported test runs	Not supported test runs
HCL OneTest™ UI	<ul style="list-style-type: none"> <li>• Accelerated Functional Testing (AFT) Suites</li> <li>• Compound Tests that contain any of the following types of tests:               <ul style="list-style-type: none"> <li>◦ Web UI tests</li> <li>◦ Traditional HTML tests</li> <li>◦ Mobile tests</li> </ul> </li> </ul>	<p>Traditional non-HTML tests.</p> <p>An independent Web UI test and a mobile test outside of a Compound Test or an AFT Suite.</p>
HCL OneTest™ API	<p>API Suites that include the following Test Suites:</p> <ul style="list-style-type: none"> <li>• Test Suites that contain scenarios with references to local stubs.</li> <li>• Test Suites that contain stubs that virtualize services run in any Kubernetes cluster or in the Kubernetes cluster that hosts HCL OneTest™ Server.</li> </ul> <p>Stubs</p>	<p>Test Suites that have tests with subscribe actions that operate in the watch mode.</p> <p>A stand-alone API test case cannot be run independently. An API test case must be part of a Test Suite.</p>
HCL OneTest™ Performance	<ul style="list-style-type: none"> <li>• Compound Tests that contain performance tests</li> <li>• Rate Schedules</li> <li>• VU Schedules</li> </ul>	<p>A single test script of any test extension.</p> <p>Tests that belong to 32-bit test extensions and SOA Quality included in VU Schedule, Rate Schedule, and Compound Test.</p>
Postman	Postman resources	
JMeter	JMeter tests	
JUnit	JUnit tests	



**Note:** You must create compound tests for the same type of tests. For example, you can create a compound test that has different mobile tests that use different mobile testing platforms.



**Restriction:** You cannot merge folders that contain different test projects in the same project directory. For example, you cannot create a project directory that contains a Performance test project and an API Suite project. You must create separate project directories for each test type. For example, you must create the Performance test in the project directory as `/MyPerfProject/` and the API Suite in the project directory as `/MyAPIProject`, before you commit the projects to the remote repository.

---

Related information

Tests configurations and test runs

## HCL OneTest Data overview

HCL® OneTest™ Data is an automated and customizable tool to generate the test data. HCL® OneTest™ Data is one of the components of HCL OneTest™ Server and can be accessed as Data Fabrication in the HCL OneTest™ Server GUI.

You can generate the test data in various file formats such as Excel files, Native file, Comma-Separated Values (CSV), JavaScript Object Notation (JSON), and Extensible Markup Language (XML).

The test data is the core component of any application testing. Creating test data manually is very time-consuming. With HCL® OneTest™ Data generator, you can generate huge volumes of intelligent data to perform application testing without the risk of data leaks or privacy issues. The generated test data can drive the testing of any application by using HCL OneTest suite.

HCL® OneTest™ Data provides the following capabilities:

- Generates real-time data automatically.
- Provides powerful built-in API to match the data with the generated datatype.
- Supports flexibility to model the data.
- Supports both single and multi-tenancy.

HCL® OneTest™ Data provides the following benefits:

- Protects data privacy.
- Improves the efficiency and accuracy of predefined data and real-time data generation.
- Avoids extraction of real and potentially sensitive information from production system.

You can find more information about the tasks that you must perform to generate the test data by using HCL® OneTest™ Data.

## Data design environment overview

The data design environment is a graphical interface to design all its components and data generation rules to generate test data on user demand. To design the components for data generation, you must log in to HCL OneTest™ Server, and then create a project.

You use various components to structure data in HCL® OneTest™ Data. The components of data design environment are:

- Files
- Connections
- Schemas

The definitions of data stored in the schemas are used to define a map. The map specifies the transformation logic in the form of rules. The map identifies the data definition of a schema, validates the type properties of data by using rules, and compiles the rules to generate the real-time test data.

You can refer to workspace management to manage workspace in HCL® OneTest™ Data.

---

### Related information

[Test assets and a server project on page 580](#)

## Task flows: HCL OneTest Data

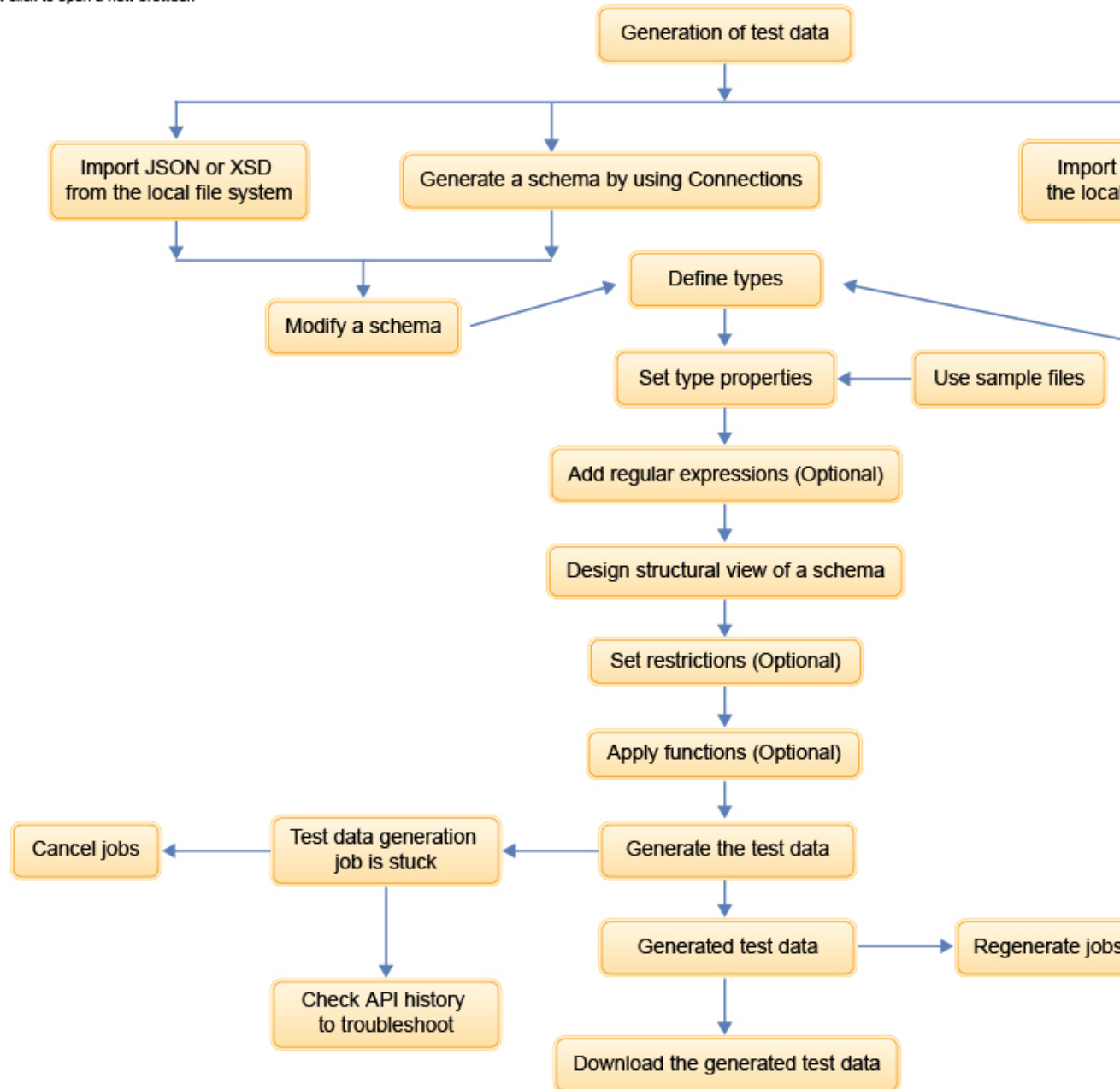
You can use the task flow diagram to get started with HCL® OneTest™ Data. After you select your project in HCL OneTest™ Server and navigate to the **Data Fabrication** page, you can complete the tasks in sequence to generate the test data. You can click the links to get more information about the tasks.

### Generating the test data by using HCL OneTest Data

You can use the task flow diagram to help you to generate the test data.



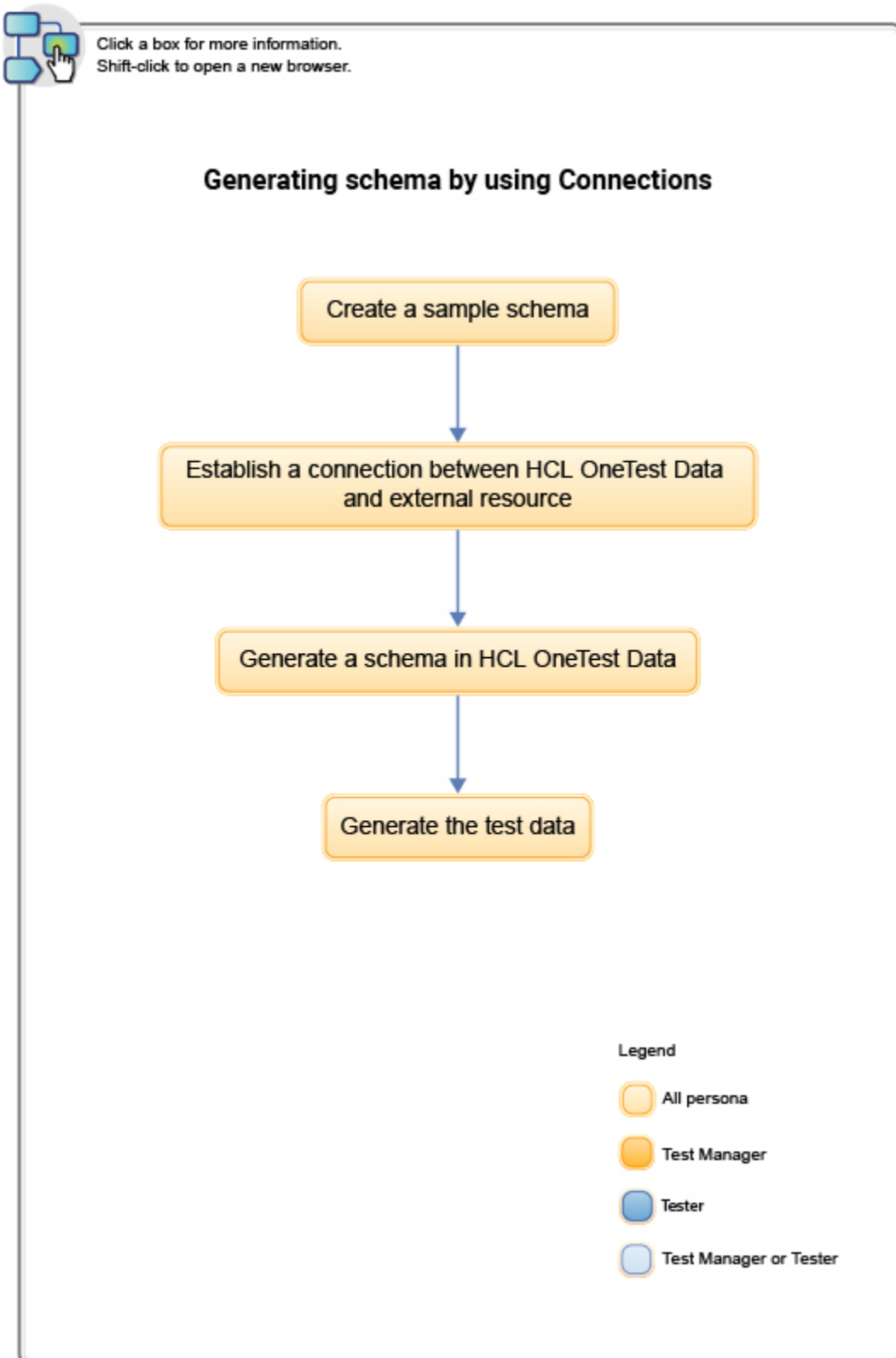
Click a box for more information.  
Shift-click to open a new browser.



1. [Generation of test data on page 23](#)
2. [Import JSON or XSD from the local file system on page 162](#)
3. [Generate a schema by using Connections on page 164](#)
4. [Import data from the local file system on page 166](#)
5. [Create a schema on page 165](#)
6. [Modify a schema on page 170](#)
7. [Design a schema on page 170](#)
8. [Define types on page 171](#)
9. [Set type properties on page 172](#)
10. [Use sample files on page 179](#)
11. [Add regular expressions on page 177](#)

## Generating schema by using Connections

You can use the task flow diagram to generate a schema by using **Connections** in the **Data Fabrication** page.



1. [Create a sample schema on page 192](#)
2. [Establish a connection between HCL OneTest Data and external resource on page 164](#)
3. [Generate a schema in HCL OneTest Data on page 195](#)
4. [Generate the test data on page 196](#)

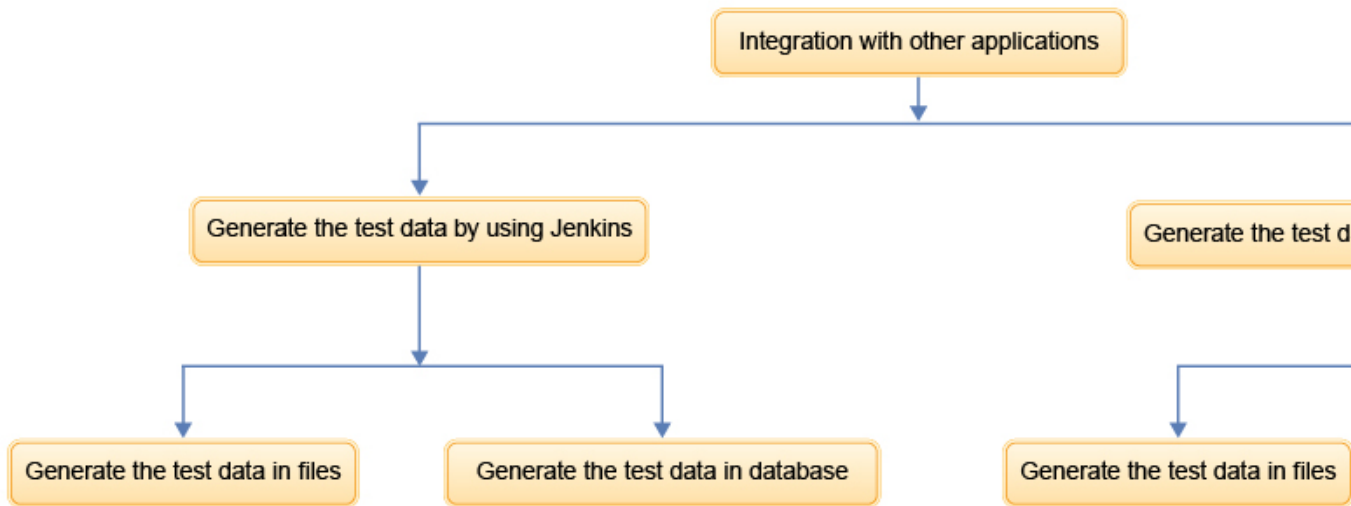
## **Integrating HCL OneTest Data with other applications**

You can use the task flow to integrate HCL® OneTest™ Data with other applications such as Jenkins and UrbanCode™ Deploy.





Click a box for more information.  
Shift-click to open a new browser.



1. [Integration with other applications on page 544](#)
2. [Generating the test data by using Jenkins on page 544](#)
3. [Generating the test data by using UrbanCode Deploy on page 550](#)
4. [Generating the test data by using Jenkins on page 544](#)
5. [Generating the test data by using Jenkins on page 544](#)
6. [Generating the test data by using UrbanCode Deploy on page 550](#)
7. [Generating the test data by using UrbanCode Deploy on page 550](#)

## Project management

You can create a project to manage the data design environment that is used to generate the test data. If the data design environment that you created is no longer required, then you can archive or delete the project. If you do want to use an archived project, you can unarchive it.

While you manage your project, the following components of HCL® OneTest™ Data are impacted:

- Projects
- Schemas
- Jobs
- Data files
- API history
- Map files
- Files
- Connections
- Actions

If you want to modify the name of your HCL® OneTest™ Data project, then you can rename the project in HCL OneTest™ Server.



**Note:** When you delete any archived project, then that project is deleted based on the configured job settings.

---

Related information

[Configuring job settings on page 240](#)

## Workspace management

You can view and manage all the data design components such as files, connections, or schema of your project on the **Workspace** page.

**Quick Links** is a list of links that enables you to access the frequently used features of HCL® OneTest™ Data. You can export or import the project by using the list of links in **Quick Links**. You can also export or import a selective list of components.

## Importing a project or components

When you have a project or any component of a project into your local file system and you want to reuse it for another project, you can import those artifacts from the local file system.

### Before you begin

- You must have a project or any component of a project in your local file system.
- The project or component that you want to import must be in a zip format.

1. From the list of links in **Quick Links**, click **Import into workspace**.

**Result**

The **Import** dialog box appears.

2. Select a file you want to import into your workspace and click **Import**.

**Results**

The selected file is imported successfully into the workspace.



**Note:** If the project file you selected to import already exists in the workspace, the import process fails.

## Exporting a project or components

When you want to reuse your existing project for another project, you can export either the selected components or an entire project to your local file system.

**Before you begin**

You must have a project.

1. To export the entire project, click **Export current workspace** from the list of links in **Quick Links**.

**Result**

The project with all its artifacts is exported to your local file system.

2. To export the selective components from a project, click **Selective export of current workspace**.

**Result**

The **Selective Export** dialog box appears.

3. Select schemas, files, or connections from the drop-down list. You can select multiple components to export.
4. Click **OK** to export.

**Results**

The selected file is exported successfully from the workspace.

## Task flow: Test runs and results in HCL OneTest™ Server

You can use the task flow diagram to get started with HCL OneTest™ Server. After you install the software, you can complete the tasks in sequence to run test assets in HCL OneTest™ Server and view and analyze the test results.

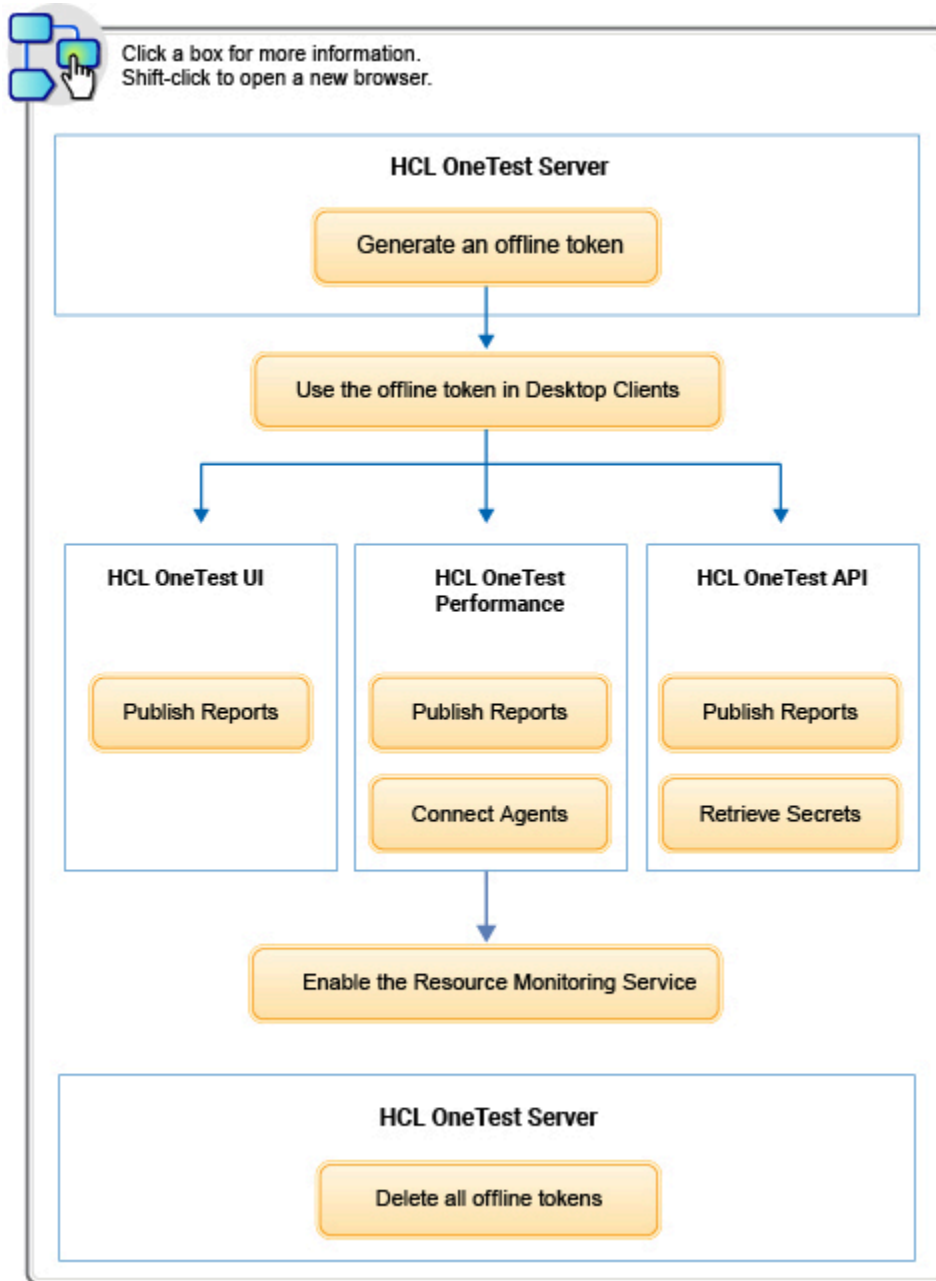
You can click the links to get more information about the tasks. The diagram also provides the tasks to be performed by personas. HCL OneTest™ Server supports user roles.



1. Default user administration on page 105
2. Test assets and a server project on page 580
3. Adding a project on page 582
4. Adding repositories to a server project on page 583
5. Protecting API test assets by using secrets on page 598
6. Managing an encrypted dataset on page 605
7. Adding users to a project on page 585
8. Adding an agent to a project
9. Tests configurations and test runs

## Task flow: HCL OneTest™ Server access from desktop clients

You can use the task flow diagram to help you access the server from the different desktop clients to retrieve secrets, publish reports, or enable resource monitoring agents. You can perform these tasks after you install and set up the server.



1. [Managing access to HCL OneTest Server on page 565](#)
2. [Managing access to HCL OneTest Server on page 565](#)
3. [Test results and reports overview on page 436](#)
4. [Test results and reports overview on page 436](#)
5. [Test results and reports overview on page 436](#)
6. [Retrieving secrets](#)
7. [Managing access to HCL OneTest Server on page 565](#)
8. [Configuring HCL OneTest Performance Agent](#)
9. [Enabling of Resource Monitoring services for a schedule](#)

---

Related information

[Managing access to server projects on page 588](#)

## Accessibility features

Accessibility features help users who have physical disabilities, such as visual and, hearing impairment, or limited mobility, to use the software products successfully.

### Accessibility compliance

The product documentation is published by using Oxygen XML WebHelp Responsive. To understand the accessibility compliance status for Oxygen XML WebHelp Responsive, refer to [WebHelp Responsive VPAT Accessibility Conformance Report](#).

### Accessing UI elements

HCL OneTest™ Server supports navigation in the UI by using different methods such as a mouse, keyboard, or touchpad.

You can use the keyboard keys such as **Tab**, arrow keys such as **UP**, **DOWN**, **LEFT**, and **RIGHT** to navigate to the different pages in the **Navigation** pane or to the different action labels in the right pane on the UI.

# Chapter 4. Administrator Guide

This guide describes how to install HCL OneTest™ Server software.

After you install the software, you can perform administration tasks such as license configuration, user management, security, memory and disk usage management, back up and restore user data, and other tasks that a server administrator can perform. This guide is intended for administrators.

## Installation of the server software

To get started with HCL OneTest™ Server, you must first install the server software.

You can install HCL OneTest™ Server on the following platforms:

- Red Hat OpenShift
- Ubuntu Server (using k3s)
- Azure Kubernetes Service (AKS)

### Installation task flows

#### Red Hat OpenShift Container platform

You must perform the following tasks to install HCL OneTest™ Server:

1. Set up a Red Hat OpenShift Container Platform.
2. Install the Jaeger operator to trace test logs and Jaeger-based reports when you run tests.
3. Install the OpenShift Container Platform command-line interface (CLI) and the Helm software.
4. Install HCL OneTest™ Server.

#### Ubuntu server

You must perform the following tasks to install HCL OneTest™ Server:

1. Set up an Ubuntu Server.
2. Install the OpenSSH server and the Helm software.
3. Set up the Kubernetes environment (k3s) by using the provided script.
4. Install HCL OneTest™ Server.

#### Azure Kubernetes Service

You must perform the following tasks to install HCL OneTest™ Server:



1. Create an Azure subscription and set up the AKS cluster.
2. Install the Helm software.
3. Install the kubectl and Azure command-line interface (CLI) software.
4. Install NGINX Ingress Controller.
5. Install HCL OneTest™ Server.

## Installation of the server software on Red Hat OpenShift

You can find information about the tasks that you can perform to install HCL OneTest™ Server software on the Red Hat OpenShift platform.

## Prerequisites for installing the server software on Red Hat OpenShift

You must complete certain tasks before you install HCL OneTest™ Server on the Red Hat OpenShift platform.

The following sections describe each prerequisite in detail:

- [Internet access on page 37](#)
- [Harbor repository credentials on page 58](#)
- [Red Hat OpenShift platform on page 38](#)
- [Mandatory software on page 38](#)
- [Service virtualization through Istio on page 39](#)
- [Service virtualization usage metrics on page 39](#)

### Internet access

You must have access to the internet to install HCL OneTest™ Server.

### Harbor repository credentials

- **New customers:**

You must create a [support ticket](#) to get the credentials that are required to access the product binaries from the Harbor repository. For more information, refer to [How to create an HCL Support case](#).

- **Existing customers:**

You must use your existing Harbor repository credentials to access the product binaries.

## Red Hat OpenShift platform

You must set up the Red Hat OpenShift platform, and then install the following components in your IT infrastructure:

- Red Hat OpenShift Container Platform V4.5 or V4.6. For more information, refer to the [Red Hat OpenShift V4.5](#) or [Red Hat OpenShift V4.6](#) documentation.
- Dynamic Volume Provisioning that supports ReadWriteOnce (RWO) and ReadWriteMany (RWX) access modes. For more information, refer to the Dynamic provisioning section in the [Red Hat OpenShift](#) documentation.
- Jaeger operator to trace test logs and Jaeger-based reports when you run tests. For more information, refer to [Installing Jaeger](#).
- **Optional:** Red Hat Service Mesh V2.0, if Istio recording and service virtualization features are required. For more information, refer to the Installing Red Hat OpenShift Service Mesh section in the [Red Hat OpenShift](#) documentation.

For more information about the service virtualization feature, see [Management of virtualized services on page 390](#).

- **Optional:** You can configure the OpenShift software-defined networking (SDN) to provide a unified cluster network that enables the communication between pods across the OpenShift Container Platform cluster. The default installation process includes the NetworkPolicy resources and these resources are acted upon only if you configure the SDN appropriately. For more information, refer to the OpenShift SDN default CNI network provider section in the [RedHat OpenShift](#) documentation.

You must have access to the OpenShift cluster with cluster administrator privileges. You can run the following command on the OpenShift command-line interface to gain access:

```
oc login -u kubeadmin -p {password} https://api.{openshift-cluster-dns-name}:6443
```

For more information about specific versions of software requirements, see [System Requirements on page 13](#).

## Mandatory software

You must install the following software on Red Hat OpenShift:

- Helm V3.5.2. For more information, refer to the Installing a Helm chart on an OpenShift Container Platform cluster section in the [Red Hat OpenShift](#) documentation.
- OpenShift CLI. For more information, refer to the Getting started with the CLI section in the [Red Hat OpenShift](#) documentation.

## Service virtualization through Istio



**Disclaimer:** This release contains access to the Istio virtualized services feature in HCL OneTest™ Server as a Tech Preview. The Tech Preview is intended for you to view the capabilities of virtualized services offered by HCL OneTest™ Server, and to provide your feedback to the product team. You are permitted to use the information only for evaluation purposes and not for use in a production environment. HCL provides the information without obligation of support and "as is" without warranty of any kind.

To provide your feedback, you must perform the following tasks:

1. Sign up or Log in to the [Ideas portal](#).
2. Click **Add a new idea** to provide your feedback on the Tech Preview features.
3. Select **OneTest TechPreview Programs** in the **Choose a workspace for this idea** field.
4. Provide all the necessary details, and then click **Share Idea**.

The default configuration does not enable service virtualization through Istio. To use the service virtualization feature through Istio, you must configure it appropriately. As a Tech Preview feature, you can virtualize the *bookinfo* sample application.

You must install the *bookinfo* application in the `bookinfo` namespace because HCL OneTest™ Server supports service virtualization through Istio only for the *bookinfo* sample application.

For more information about the *bookinfo* application, refer to the [Istio](#) documentation.

## Service virtualization usage metrics

You must perform the following tasks so that HCL OneTest™ Server shows the correct usage of service virtualization metrics:

- Create the `cluster-monitoring-config` `ConfigMap` object to configure the OpenShift Container Platform (OCP) monitoring stack, if the `ConfigMap` does not exist.

For more information about creating a cluster monitoring config map, based on the version of OCP that you use, refer to the following documentation:

- [Creating a cluster monitoring config map in OCP V4.5](#)
- [Creating a cluster monitoring config map in OCP V4.6](#)
- Enable the user workload monitoring services in the OpenShift cluster to show the service virtualization usage metrics in HCL OneTest™ Server.

For more information about the enabling monitoring of your services, based on the version of OCP that you use, refer to the following documentation:

- [Enabling monitoring of your services in OCP V4.5](#)
- [Enabling monitoring of your services in OCP V4.6](#)

---

#### Related information

[OpenShift Container Platform 4.5](#)

[OpenShift Container Platform 4.6](#)

## Installing the server software on Red Hat OpenShift

You can install HCL OneTest™ Server on the Red Hat OpenShift server that has the Kubernetes Engine environment to run functional, integration, and performance tests. HCL OneTest™ Server combines test data, test environments, and test runs and reports into a single web-based browser for testers and non-testers.

### Before you begin

You must have performed the following tasks:

- Completed tasks provided in the Prerequisites section. See [Prerequisites for installing the server software on Red Hat OpenShift on page 37](#).
- Installed the following software:
  - OpenShift CLI
  - Helm
- Copied the **Secret key** from the Harbor repository.

1. Log in to your OCP cluster as a cluster administrator by running the `oc login` command.
2. Create a namespace in which you want to install the server software by running the following command:

```
oc new-project test-system
```



**Remember:** The `test-system` is the name of the namespace. If you created a namespace by using a different value, then you must use that value in place of `test-system` in all the instances in this procedure.

3. Add the software registry to Helm to access the server install charts by running the following command:

```
helm repo add hclsoftware https://hclcr.io/chartrepo/ot --username {okta-email-address}
--password {harbor-cli-secret}
```



**Note:** You must replace `{okta-email-address}` with the user name of the Harbor repository and replace `{harbor-cli-secret}` with the secret key that you copied from the Harbor repository.

If the user name contains any special characters, such as \$, you must enclose it within single quotes.

4. Run the following command to get the latest updates from the repository:

```
helm repo update
```

5. Run the following command to retrieve the charts:

```
helm pull --untar hclsoftware/hcl-onetest-server --version 5.1020.0
```

6. Create a Secret to pull images that are used by HCL OneTest™ Server by running the following commands:

```
oc create secret docker-registry hclcr.io \
-n test-system \
--docker-server=hclcr.io \
--docker-username={okta-email-address} \
--docker-password={harbor-cli-secret} \
--docker-email=example@abc.com
```



**Notes:**

- You must replace `{okta-email-address}` with the user name of the Harbor repository and replace `{harbor-cli-secret}` with the secret key that you copied from the Harbor repository.

If the user name contains any special characters, such as \$, you must enclose it within single quotes.

- You can replace `example@abc.com` with the email address of the administrator if required.



**Remember:** `test-system` is the name of the namespace. If you created a namespace by using a different value, then you must use that value in place of the `test-system` in all the instances in this procedure.

7. Perform one of the following steps to enable certificates as trusted certificates:

You must go to [Step 7.b on page 42](#) if you use OpenShift Service Mesh service virtualization, a Tech Preview feature.

- a. Perform the following steps to add the Certificate Authority (CA) into a Secret:

- i. Run the following command to verify whether an additional CA is required:

```
curl -sw'#{http_code}' -o/dev/null \
"https://wildcard.$(oc get -n openshift-ingress-operator ingresscontroller default
-ojsonpath='{.status.domain}')
```

If the result of the command is displayed as 503, the CA is already trusted. You must continue with [8 on page 43](#).

If the result of the command is displayed as 000, then CA must be added into a Secret. You must continue with [7.a.ii on page 42](#).

- ii. Run the following command to get the default CA in a PEM format:

```
oc get -n openshift-ingress-operator secret router-ca -ojsonpath='{.data.tls.crt}' | base64 --decode > ca.crt
```

- iii. Run the following command to validate that the CA used to sign the certificate is the same for ingress:

```
curl -sw'%{http_code}' -o/dev/null --cacert ca.crt \
"https://wildcard.${(oc get -n openshift-ingress-operator ingresscontroller default -ojsonpath='{.status.domain}')}"
```

If the result of the command is displayed as 503, then you must continue with next step.

If the result of the command is displayed as 000, then the configuration of the certificate has been customized. You must find the signer of the certificate to continue with the next step.

- iv. Run the following command to create an ingress Secret to store the CA:

```
oc create secret generic -n test-system ingress --from-file=ca.crt=ca.crt
```

- b. Perform the following steps to add the Certificate Authority (CA) into a Secret if you use OpenShift Service Mesh service virtualization, a Tech Preview feature:

- i. Run the following script from the `hcl-onetest-server/files` directory:

```
./files/certificate.sh -n istio-system -s istio-ingressgateway-certs
{openshift-cluster-dns-name}
```

You must replace the `{openshift-cluster-dns-name}` with the ingress DNS name that you selected for the server. You can run the following command to obtain the default value of `openshift-cluster-dns-name`:

```
oc get --namespace=openshift-ingress-operator ingresscontroller/default
-ojsonpath='{.status.domain}'
```

- ii. Run the following command to create an ingress Secret:

```
oc create secret generic -n test-system ingress \
"--from-literal=ca.crt=$(oc get -n istio-system secret istio-ingressgateway-certs
-ojsonpath='{.data.ca.crt}' | base64 --decode)"
```

- iii. Run the following commands to enable an OpenShift route that the product creates for the Istio gateway:

```
cat <<EOF | oc apply -n istio-system -f - >/dev/null
apiVersion: maistra.io/v1
```

```

kind: ServiceMeshMemberRoll
metadata:
  name: default
spec:
  members:
  - test-system
EOF

```

When some components such as static agents or Docker agents want to communicate with HCL OneTest™ Server, the component presents its certificate to the server to verify its identity. HCL OneTest™ Server trusts the component only if it is signed by a recognized and trusted CA. Therefore, you must add the signed CA into a trust by placing it in a Secret to enable certificates as trusted certificates.

8. Perform the following steps to install the server software:

- a. Run the following commands to update the runAsUser and fsGroup to match the Security Context Constraints (SCCs):

```

sed -i -e "s/runAsUser: 1001/runAsUser: $(oc get project test-system -oyaml \
| sed -r -n 's# *openshift.io/sa.scc.uid-range: *([0-9]*)/.*\#1#p')/g;
s/fsGroup: 1001/fsGroup: $(oc get project test-system -oyaml \
| sed -r -n 's# *openshift.io/sa.scc.supplemental-groups: *([0-9]*)/.*\#1#p')/g"
hcl-onetest-server/values-openshift.yaml

```

HCL OneTest™ Server is compatible with the restricted SCC. You must run this command to ensure that the runAsUser and fsGroup strategies match with the SCC policy.

- b. Perform one of the steps described in the following table to install the server software based on your requirement:

Step description	Step no
To install the server software	Perform <a href="#">8.b.i on page 44</a>
To install the server software and enable the OpenShift Service Mesh service virtualization through Istio, a Tech Preview feature	Perform <a href="#">8.b.ii on page 44</a> and <a href="#">8.b.iv on page 44</a>
To install the server software, enable the OpenShift Service Mesh service virtualization through Istio, a Tech Preview feature, and enable Jaeger for performance and Web UI tests logs	Perform <a href="#">8.b.iii on page 44</a> and <a href="#">8.b.iv on page 44</a>
To install the server software and enable Jaeger for performance and Web UI tests logs	Perform <a href="#">8.b.v on page 45</a>

- i. Run the following command to install the server software:

```
helm install {my-ots} ./hcl-onetest-server -n test-system \
-f hcl-onetest-server/values-openshift.yaml \
--set global.persistence.rwxStorageClass=rook-ceph-file
--set global.hclOneTestIngressDomain=onetest.{openshift-cluster-dns-name} \
--set global.hclFlexnetURL=https://hclsoftware.compliance.flexnetoperations.com \
--set global.hclFlexnetID={cloud-license-server-id} \
--set global.hclImagePullSecret=hclcr.io \
--set global.hclOneTestPasswordAutoGenSeed={password-seed}
```



**Note:** You must use the `--set global.hclCertSecretOptional=false` parameter in the `helm install` command only if you performed step 7.a.iv on page 42 to create an ingress Secret to store the CA.

- ii. Run the following command to install the server software and to enable OpenShift Service Mesh service virtualization, a Tech Preview feature:

```
helm install {my-ots} ./hcl-onetest-server -n test-system \
-f hcl-onetest-server/values-openshift.yaml \
--set global.persistence.rwxStorageClass=rook-ceph-file
--set global.hclOneTestIngressDomain=onetest.{openshift-cluster-dns-name} \
--set global.hclFlexnetURL=https://hclsoftware.compliance.flexnetoperations.com \
--set global.hclFlexnetID={cloud-license-server-id} \
--set global.hclImagePullSecret=hclcr.io \
--set global.hclOneTestPasswordAutoGenSeed={password-seed} \
-f hcl-onetest-server/values-openshift-demo.yaml
```

- iii. Run the following command to install the server software, to enable OpenShift Service Mesh service virtualization, a Tech Preview feature, and to enable Jaeger for performance and Web UI tests logs:

```
helm install {my-ots} ./hcl-onetest-server -n test-system \
-f hcl-onetest-server/values-openshift.yaml \
--set global.persistence.rwxStorageClass=rook-ceph-file
--set global.hclOneTestIngressDomain=onetest.{openshift-cluster-dns-name} \
--set global.hclFlexnetURL=https://hclsoftware.compliance.flexnetoperations.com \
--set global.hclFlexnetID={cloud-license-server-id} \
--set global.hclImagePullSecret=hclcr.io \
--set global.hclOneTestPasswordAutoGenSeed={password-seed} \
-f hcl-onetest-server/values-openshift-demo.yaml \
--set-string execution.annotations.sidecar\\.jaegertracing\\.io/inject=true \
--set global.jaegerAgent.internalHostName=localhost \
--set global.jaegerDashboard.externalURL={my-jaeger-dashboard-url}
```

- iv. Run the following command to enable service virtualization through Istio, a Tech Preview feature in the specific namespace:

```
oc create rolebinding istio-virtualization-enabled -n
bookinfo --clusterrole={my-ots}-execution-istio-test-system
--serviceaccount=test-system:{my-ots}-execution
```



Where, {my-ots} is the name of the release that you provided during the installation of the server software.



**Note:** When you uninstall the chart, the manually created role bindings are not deleted from the namespace. You can run the following command to delete the role bindings:

```
oc delete rolebinding istio-virtualization-enabled -n bookinfo
```

- v. Run the following command to install the server software and to enable Jaeger for performance and Web UI tests logs:

```
helm install {my-ots} ./hcl-onetest-server -n test-system \
-f hcl-onetest-server/values-openshift.yaml \
--set global.persistence.rwxStorageClass=rook-ceph-file \
--set global.hclOneTestIngressDomain=onetest.{openshift-cluster-dns-name} \
--set global.hclFlexnetURL=https://hclsoftware.compliance.flexnetoperations.com \
--set global.hclFlexnetID={cloud-license-server-id} \
--set global.hclImagePullSecret=hclcr.io \
--set global.hclOneTestPasswordAutoGenSeed={password-seed} \
--set-string execution.annotations.sidecar\\.jaegertracing\\.io/inject=true \
--set global.jaegerAgent.internalHostName=localhost \
--set global.jaegerDashboard.externalURL={my-jaeger-dashboard-url}
```



**Note:** You must use the `--set global.hclCertSecretOptional=false` parameter in the `helm install` command only if you performed step 7.a.iv on page 42 to create an Ingress Secret to store the CA.

You must substitute the value of the variables in the `helm install` command with the actual value:

- `{my-ots}` with the release name of your choice.



**Note:** The release name must consist of alphanumeric characters that are in lowercase or “-” (hyphen). The release name must also start with an alphabetic character and end with an alphanumeric character. For example, *my-org* or *abc-123*.

- `{openshift-cluster-dns-name}` with the Ingress DNS name that you selected for the server.



**Remember:** You must provide the value that consists of alphanumeric characters that are in lowercase, “-” (hyphen) or “.” (period). The value must also start and end with an alphanumeric character.



**Note:** You can run the following command to obtain the default value of `openshift-cluster-dns-name`:



```
oc get --namespace=openshift-ingress-operator ingresscontroller/default
-ojsonpath='{.status.domain}'
```

- `{password-seed}` with a value of your choice.



**Important:** This password seed is used to create several default passwords for the server. You must store the password seed securely. When you install the server software by using the backup of the user data, you can reuse the password seed. You can use this seed to restore the backed-up files either on the current or later versions of the server software.

- **Optional:** `{cloud-license-server-id}` with the ID of the License Server for the initial team space, if you want to set the license for the first time.



**Important:** If you want to upgrade the product from the previous version, you must configure the value of **License Server ID** from the **Team Space License Configuration** page when the installation of the server is complete.

- `{my-jaeger-dashboard-url}` with the URL of the Jaeger server.

9. **Optional:** Run the following command to remove a job that is used to initialize the PostgreSQL database during the installation of the server software:

```
oc delete job {my-ots}-postgresql-init -n test-system
```

10. **Optional:** Perform the following steps to migrate data into HCL OneTest™ Server, if you upgraded the product from the previous version (V10.1.0, V10.1.1, or V10.1.2):

- a. Run the following script from the `hcl-onetest-server/files` directory to create a directory that contains metadata related to the Persistent Volume Claims and their Persistent Volumes:

```
migrate.sh create-pvcs -n test-system {my-ots}
```

- b. Run the following script from the `hcl-onetest-server/files` directory to merge the data into the server:

```
migrate.sh merge-dbs -n test-system {my-ots}
```

- c. Run the following command to remove the resources that were created during the migration process:

```
migrate.sh delete-temp-resources -n test-system {my-ots}
```

where `{my-ots}` is the name of the release that you provided during the installation of the server software.

11. Run the following command to verify and test the installed server software:

```
$ helm test {my-ots} -n test-system
```

where `{my-ots}` is the name of the release that was provided during the installation of the server software.

## Results

You have installed the server software. The command line displays the following information:

- Keycloak URL to manage and authenticate users.
- A URL to access the HCL OneTest™ Server UI.

## What to do next

You can perform certain tasks as a *Server Administrator*. See [Configuration of the server software on page 104](#).

---

Related information

[Troubleshooting Guide on page 987](#)

## Installation of the server software in an air-gapped environment

You can find information about the tasks that you can perform to install HCL OneTest™ Server on the Red Hat OpenShift platform in an air-gapped environment.

If your OpenShift Container Platform (OCP) cluster is within a restricted network with no access to the internet, then you can install the product on such OCP clusters by using an air-gapped environment. To install the product, unlike online installations, air-gapped installations require an internal registry to mimic a typical online installation by using images from your internal registry.

In an air-gapped environment, you can mirror the images of HCL OneTest™ Server from IBM Entitled Registry to the internal Docker registry that has access to the OCP cluster. With the help of a Bastion host that can connect to the internet and your OCP cluster, you can install HCL OneTest™ Server on the OCP cluster.

## Prerequisites for installing the server software in an air-gapped environment

You must complete certain tasks before you install HCL OneTest™ Server in an air-gapped environment on the Red Hat OpenShift platform.

The following sections describe each prerequisite in detail:

- [Entitlement key on page 48](#)
- [Red Hat OpenShift Container Platform on page 48](#)
- [Bastion host on page 48](#)
- [Docker registry on page 48](#)
- [Mandatory software on page 49](#)

- [Service virtualization through Istio on page 49](#)
- [Service virtualization usage metrics on page 49](#)

## Entitlement key

You must copy the **Entitlement key** from the [Container software library](#) to pull the images that are used by the server software. For more information, refer to [Creating an API key](#).

## Red Hat OpenShift Container Platform

You must install Red Hat OpenShift Container Platform (OCP) and have cluster administrator privileges. For more information, refer to the Installing section of [Red Hat OpenShift](#) documentation.



### Note:

For more information about specific versions of software requirements, see [System Requirements on page 13](#).

You can run the following command on the OpenShift command-line interface to gain cluster administrator access:

```
oc login -u kubeadmin -p {password} https://api.{openshift-cluster-dns-name}:6443
```



**Note:** You can run the following command to obtain the default value of `openshift-cluster-dns-name`:

```
oc get --namespace=openshift-ingress-operator ingresscontroller/default  
-ojsonpath='{.status.domain}'
```

## Bastion host

You must set up a Bastion host that has Linux installed. You must also have internet access.

A bastion host is a server that is provisioned with a public IP address and must be accessible through remote access Secure Shell (SSH).



**Note:** You must ensure that the Bastion host has access to your OCP cluster.

For more information refer to the [IBM Cloud Docs](#).

## Docker registry

You must set up a local Docker registry to store the images of HCL OneTest™ Server and ensure that it has access to the OCP cluster and the Bastion host.

For more information refer to the [Red Hat OpenShift](#) documentation.

## Mandatory software

You must install the following software on Red Hat OpenShift:

- OpenShift command-line interface (CLI) on the Bastion host. For more information, refer to the Getting started with the CLI section in the [Red Hat OpenShift](#) documentation.
- IBM Cloud Pak CLI (cloudctl) on the Bastion host to manage Container Application Software for Enterprises (CASE). For more information, refer to the [cloud-pak-cli](#) repository.
- Installed Helm V3.5.2. For more information, refer to the Installing a Helm chart on an OpenShift Container Platform cluster section in the [Red Hat OpenShift](#) documentation.

## Service virtualization through Istio



**Disclaimer:** This release contains access to the Istio virtualized services feature in HCL OneTest™ Server as a Tech Preview. The Tech Preview is intended for you to view the capabilities of virtualized services offered by HCL OneTest™ Server, and to provide your feedback to the product team. You are permitted to use the information only for evaluation purposes and not for use in a production environment. HCL provides the information without obligation of support and "as is" without warranty of any kind.

To provide your feedback, you must perform the following tasks:

1. Sign up or Log in to the [Ideas portal](#).
2. Click **Add a new idea** to provide your feedback on the Tech Preview features.
3. Select **OneTest TechPreview Programs** in the **Choose a workspace for this idea** field.
4. Provide all the necessary details, and then click **Share Idea**.

The default configuration does not enable service virtualization through Istio. To use the service virtualization feature through Istio, you must configure it appropriately. As a Tech Preview feature, you can virtualize the *bookinfo* sample application.

You must install the *bookinfo* application in the `bookinfo` namespace because HCL OneTest™ Server supports service virtualization through Istio only for the *bookinfo* sample application.

For more information about the *bookinfo* application, refer to the [Istio](#) documentation.

## Service virtualization usage metrics

You must perform the following tasks so that HCL OneTest™ Server shows the correct usage of service virtualization metrics:

- Create the `cluster-monitoring-config` ConfigMap object to configure the OpenShift Container Platform (OCP) monitoring stack, if the ConfigMap does not exist.

For more information about creating a cluster monitoring config map, based on the version of OCP that you use, refer to the following documentation:

- [Creating a cluster monitoring config map in OCP V4.5](#)
- [Creating a cluster monitoring config map in OCP V4.6](#)

- Enable the user workload monitoring services in the OpenShift cluster to show the service virtualization usage metrics in HCL OneTest™ Server.

For more information about the enabling monitoring of your services, based on the version of OCP that you use, refer to the following documentation:

- [Enabling monitoring of your services in OCP V4.5](#)
- [Enabling monitoring of your services in OCP V4.6](#)

#### Related information

[OpenShift Container Platform 4.5 Documentation](#)

[OpenShift Container Platform 4.6 Documentation](#)

[IBM Cloud Pak foundational services](#)

[System Requirements on page 13](#)

## Configuring the Bastion host

You must configure the Bastion host to mirror the images of HCL OneTest™ Server from IBM Entitled Registry to the OpenShift Container Platform (OCP) cluster.

### Before you begin

You must have performed the following tasks:

- Completed the tasks provided in the Prerequisites section. See [Prerequisites for installing the server software in an air-gapped environment on page 47](#).
- Copied the **Entitlement key** from the [Container software library](#).

1. Log in to your OCP cluster as a cluster administrator by running the `oc login` command.
2. Run the following command to confirm that the container registry is accessible from the Bastion host:

```
oc get routes -n openshift-image-registry
```

3. **Optional:** Create a route to provide external access to the container registry if no resources are found:

```
oc patch configs.imageregistry.operator.openshift.io/cluster \
--patch '{"spec":{"defaultRoute":true}}' --type=merge
```

4. Create a namespace for the internal registry to host the mirror images:

```
oc new-project cp
```



**Note:** The `cp` is the name of the namespace and it is required to pull the images from IBM Entitled Registry.

5. Create the following environment variables with the image name and download the images of HCL OneTest™ Server to the offline store by running the following commands:

Before mirroring images of HCL OneTest™ Server, you must set the environment variables on your Bastion host, and connect to the internet so that you can download the corresponding CASE files from IBM Entitled Registry.

```
export CASE_NAME=ibm-rtas-case
export CASE_VERSION=5.1020.0
export CASE_ARCHIVE=${CASE_NAME}-${CASE_VERSION}.tgz
export CASE_REMOTE_PATH=https://github.com/IBM/cloud-pak/raw/master/repo/case/${CASE_ARCHIVE}
export OFFLINEDIR=$HOME/offline

cloudctl case save \
--case $CASE_REMOTE_PATH \
--outputdir $OFFLINEDIR
```

6. Run the following commands to store credentials to authenticate IBM Entitled Registry:

The images of HCL OneTest™ Server are within IBM Entitled Registry. You must log in to IBM Entitled Registry to access source images and log in to your internal registry that acts as a local registry when you copy the images.

```
export ENTITLEMENT_KEY=YOUR_ENTITLEMENT_KEY
export CASE_INVENTORY_SETUP=ibmRtasProd

cloudctl case launch \
--case $OFFLINEDIR/$CASE_ARCHIVE \
--inventory $CASE_INVENTORY_SETUP \
--action configure-creds-airgap \
--args "--registry cp.icr.io --user cp --pass $ENTITLEMENT_KEY"
```



**Note:** You must replace `YOUR_ENTITLEMENT_KEY` with the key that you copied from the [Container software library](#).

7. Run the following commands to store credentials to authenticate the local Docker registry:

```
export LOCAL_DOCKER_REGISTRY=$(oc get route default-route -n openshift-image-registry -o
jsonpath='{.spec.host}')
export LOCAL_DOCKER_USER=UNUSED
export LOCAL_DOCKER_PASSWORD=$(oc whoami -t)

cloudctl case launch \
```

```
--case $OFFLINEDIR/$CASE_ARCHIVE \
--inventory $CASE_INVENTORY_SETUP \
--action configure-creds-airgap \
--args "--registry $LOCAL_DOCKER_REGISTRY --user $LOCAL_DOCKER_USER --pass
$LOCAL_DOCKER_PASSWORD"
```

8. Run the following command to mirror the images from IBM Entitled Registry to the local Docker registry:

```
cloudctl case launch \
--case $OFFLINEDIR/$CASE_ARCHIVE \
--inventory $CASE_INVENTORY_SETUP \
--action mirror-images \
--args "--registry $LOCAL_DOCKER_REGISTRY --inputDir $OFFLINEDIR"
```

9. Run the following commands to configure the OCP cluster where HCL OneTest™ Server needs to be installed with a `global pull secret` for the local Docker registry:

By using the `global pull secret`, the OCP cluster can pull the images of HCL OneTest™ Server from your local Docker registry instead of IBM Entitled Registry.

```
cloudctl case launch \
--case $OFFLINEDIR/$CASE_ARCHIVE \
--inventory $CASE_INVENTORY_SETUP \
--action configure-cluster-airgap \
--namespace default \
--args "--registry $LOCAL_DOCKER_REGISTRY --inputDir $OFFLINEDIR"
```



#### Notes:

- After you perform step 9 on page 52, the cluster restarts all the nodes.
- The `namespace` argument is required for legacy reasons, its value can be any namespace that exists.

10. Run the following command to extract the mirrored images:

```
tar xf $OFFLINEDIR/charts/ibm-rtas-prod-5.1020.0.tgz
```

## Results

You have configured the Bastion host and mirrored the images of HCL OneTest™ Server.

## What to do next

You can install the server software on your restricted OCP. See [Installing the server software in an air-gapped environment on page 53](#).

---

### Related information

[Using image pull secrets](#)

[Configuring image registry repository mirroring](#)



[IBM Cloud Pak foundational services](#)

[IBM Cloud Pak CLI \(cloudctl\) CASE commands](#)

## Installing the server software in an air-gapped environment

You might have an environment that has a high level of security and is isolated from the internet. In such a scenario, you can install HCL OneTest™ Server on the Red Hat OpenShift platform in an air-gapped environment.

### Before you begin

You must have performed the following tasks:

- Completed the tasks provided in the Prerequisites section. See [Prerequisites for installing the server software in an air-gapped environment on page 47](#).
  - Configured the Bastion host. See [Configuring the Bastion host on page 50](#).
  - Copied the **Entitlement key** from the [Container software library](#).
1. Log in to your OCP cluster as a cluster administrator by running the `oc login` command.
  2. Create a namespace in which you want to install the server software by running the following command:

```
oc new-project test-system
```



**Remember:** The `test-system` is the name of the namespace. If you created a namespace by using a different value, then you must use that value in place of `test-system` in all the instances in this procedure.

3. Perform one of the following steps to enable certificates as trusted certificates:

You must go to [Step 3.b on page 54](#) if you use OpenShift Service Mesh service virtualization, a Tech Preview feature.

- a. Perform the following steps to add the Certificate Authority (CA) into a Secret:

- i. Run the following command to verify whether an additional CA is required:

```
curl -sw'{http_code}' -o/dev/null \
"https://wildcard.${(oc get -n openshift-ingress-operator ingresscontroller default
-ojsonpath='{.status.domain}')"
```

If the result of the command is displayed as `503`, the CA is already trusted. You must continue with [4.b.i on page 55](#).

If the result of the command is displayed as `000`, then CA must be added into a Secret. You must continue with [step 3.a.ii on page 53](#).

- ii. Run the following command to get the default CA in a PEM format:

```
oc get -n openshift-ingress-operator secret router-ca -ojsonpath='{.data.tls\.crt}'
| base64 --decode > ca.crt
```

- iii. Run the following command to validate that the CA used to sign the certificate is the same for ingress:

```
curl -sw'%{http_code}' -o/dev/null --cacert ca.crt \
"https://wildcard.$(oc get -n openshift-ingress-operator ingresscontroller default
-ojsonpath='{.status.domain}')
```

If the result of the command is displayed as 503, then you must continue with the next step.

If the result of the command is displayed as 000, then the configuration of the certificate has been customized. You must find the signer of the certificate to continue with the next step.

- iv. Run the following command to create an ingress Secret to store the CA:

```
oc create secret generic -n test-system ingress --from-file=ca.crt=ca.crt
```

- b. Perform the following steps to add the CA into a Secret if you use OpenShift Service Mesh service virtualization, a Tech Preview feature:

- i. Run the following script from the `hcl-onetest-server/files` directory:

```
./files/certificate.sh -n istio-system -s istio-ingressgateway-certs
{openshift-cluster-dns-name}
```

You must replace the `{openshift-cluster-dns-name}` with the ingress DNS name that you selected for the server. You can run the following command to obtain the default value of

`openshift-cluster-dns-name`:

```
oc get --namespace=openshift-ingress-operator ingresscontroller/default
-ojsonpath='{.status.domain}'
```

- ii. Run the following command to create an ingress Secret:

```
oc create secret generic -n test-system ingress \
"--from-literal=ca.crt=$(oc get -n istio-system secret istio-ingressgateway-certs
-ojsonpath='{.data.ca\.crt}' | base64 --decode)"
```

- iii. Run the following commands to enable an OpenShift route that the product creates for the Istio gateway:

```
cat <<EOF | oc apply -n istio-system -f - >/dev/null
apiVersion: maistra.io/v1
kind: ServiceMeshMemberRoll
metadata:
  name: default
spec:
  members:
    - test-system
EOF
```

When some components such as static agents or Docker agents want to communicate with HCL OneTest™ Server, the component presents its certificate to the server to verify its identity. HCL OneTest™ Server trusts the component only if it is signed by a recognized and trusted CA. Therefore, you must add the signed CA into a trust by placing it in a Secret to enable certificates as trusted certificates.

4. Perform the following steps to install the server software:

- a. Run the following commands to update the runAsUser and fsGroup to match the Security Context Constraints (SCCs):

```
sed -i -e "s/runAsUser: 1001/runAsUser: $(oc get project test-system -oyaml \
| sed -r -n 's# *openshift.io/sa.scc.uid-range: *([0-9]*)/.*#\1#p')/g;
s/fsGroup: 1001/fsGroup: $(oc get project test-system -oyaml \
| sed -r -n 's# *openshift.io/sa.scc.supplemental-groups: *([0-9]*)/.*#\1#p')/g"
hcl-onetest-server/values-openshift.yaml
```

HCL OneTest™ Server is compatible with the restricted SCC. You must run this command to ensure that the runAsUser and fsGroup strategies match with the SCC policy.

- b. Perform one of the steps described in the following table to install the server software based on your requirement:

Step description	Step no
To install the server software	Perform <a href="#">4.b.i on page 55</a>
To install the server software and enable the OpenShift Service Mesh service virtualization through Istio, a Tech Preview feature	Perform <a href="#">4.b.ii on page 56</a> and <a href="#">4.b.iv on page 56</a>
To install the server software, enable the OpenShift Service Mesh service virtualization through Istio, a Tech Preview feature, and enable Jaeger for performance and Web UI tests logs	Perform <a href="#">4.b.iii on page 56</a> and <a href="#">4.b.iv on page 56</a>
To install the server software and enable Jaeger for performance and Web UI tests logs	Perform <a href="#">4.b.v on page 56</a>

- i. Run the following command to install the server software:

```
helm install {my-rtas} ./ibm-rtas-prod -n test-system \
--set license=true \
-f ibm-rtas-prod/values-openshift.yaml \
--set global.persistence.rwxStorageClass=ibmc-file-gold \
--set global.ibmRtasIngressDomain=rtas.{openshift-cluster-dns-name} \
--set global.ibmRtasPasswordAutoGenSeed={password-seed} \
--set global.rationalLicenseKeyServer=@{rlks-ip-address}
```



**Note:** You must use the `--set global.ibmRtasCertSecretOptional=false` parameter in the `helm install` command only if you performed step 3.a.iv on page 54 to create an ingress Secret to store the CA.

- ii. Run the following command to install the server software and to enable OpenShift Service Mesh service virtualization, a Tech Preview feature:

```
helm install {my-rtas} ./ibm-rtas-prod -n test-system \
--set license=true \
-f ibm-rtas-prod/values-openshift.yaml \
-f ibm-rtas-prod/values-openshift-demo.yaml \
--set global.persistence.rwxStorageClass=ibmc-file-gold \
--set global.ibmRtasIngressDomain=rtas.{openshift-cluster-dns-name} \
--set global.ibmRtasPasswordAutoGenSeed={password-seed} \
--set global.rationalLicenseKeyServer=@{rlks-ip-address}
```

- iii. Run the following command to install the server software, to enable OpenShift Service Mesh service virtualization, a Tech Preview feature, and to enable Jaeger for performance and Web UI tests logs:

- iv. Run the following command to enable service virtualization through Istio, a Tech Preview feature in the specific namespace:

```
oc create rolebinding istio-virtualization-enabled -n
bookinfo --clusterrole={my-ots}-execution-istio-test-system
--serviceaccount=test-system:{my-ots}-execution
```

Where, {my-ots} is the name of the release that you provided during the installation of the server software.



**Note:** When you uninstall the chart, the manually created role bindings are not deleted from the namespace. You can run the following command to delete the role bindings:

```
oc delete rolebinding istio-virtualization-enabled -n bookinfo
```

- v. Run the following command to install the server software and to enable Jaeger for performance and Web UI tests logs:



**Note:** You must use the `--set global.ibmRtasCertSecretOptional=false` parameter in the `helm install` command only if you performed step 3.a.iv on page 54 to create an ingress Secret to store the CA.

You must replace the value of the variables in the helm install command with the actual value:

- `{my-ots}` with the release name of your choice.



**Note:** The release name must consist of alphanumeric characters that are in lowercase or “-” (hyphen). The release name must also start with an alphabetic character and end with an alphanumeric character. For example, *my-org* or *abc-123*.

- `{openshift-cluster-dns-name}` with the ingress DNS name that you selected for the server.



**Remember:** You must provide the value that consists of alphanumeric characters that are in lowercase, “-” (hyphen) or “.” (period). The value must also start and end with an alphanumeric character.



**Note:** You can run the following command to obtain the default value of `openshift-cluster-dns-name`:

```
oc get --namespace=openshift-ingress-operator ingresscontroller/default
-ojsonpath='{.status.domain}'
```

- `{password-seed}` with a value of your choice.



**Important:** This password seed is used to create several default passwords for the server. You must store the password seed securely. When you install the server software by using the backup of the user data, you can reuse the password seed. You can use this seed to restore the backed-up files either on the current or later versions of the server software.

- `{my-jaeger-dashboard-url}` with the URL of the Jaeger server.

5. **Optional:** Run the following command to remove a job that is used to initialize the PostgreSQL database during the installation of the server software:

```
oc delete job {my-ots}-postgresql-init -n test-system
```

6. Run the following command to verify and test the installed server software:

```
$ helm test {my-ots} -n test-system
```

where `{my-ots}` is the name of the release that was provided during the installation of the server software.

## Results

You have installed the server software. The command line displays the following information:

- Keycloak URL to manage and authenticate users.
- A URL to access the HCL OneTest™ Server UI.

## What to do next

You can perform certain tasks as a *Server Administrator*. See [Configuration of the server software on page 104](#).

---

Related information

[Troubleshooting Guide on page 987](#)

## Installation of the server software on Ubuntu

You can find information about the tasks that you can perform to install HCL OneTest™ Server software on the Ubuntu platform.

## Prerequisites for installing the server software on Ubuntu

You must complete certain tasks before you install HCL OneTest™ Server on the Ubuntu platform.

The following sections describe each prerequisite in detail:

- [Internet access on page 58](#)
- [Harbor repository credentials on page 58](#)
- [Ubuntu Server on page 59](#)
- [Backup of user data on page 59](#)
- [Mandatory software on page 59](#)
- [k3s environment on page 59](#)
- [Service virtualization through Istio on page 60](#)
- [Collection of usage metrics on page 60](#)

### Internet access

You must have access to the internet to install HCL OneTest™ Server.

### Harbor repository credentials

- **New customers:**

You must create a [support ticket](#) to get the credentials that are required to access the product binaries from the Harbor repository. For more information, refer to [How to create an HCL Support case](#).

- **Existing customers:**

You must use your existing Harbor repository credentials to access the product binaries.

## Ubuntu Server

You must install an Ubuntu Server and have a working Domain Name Server (DNS) that resolves the host name into a machine-readable IP address.

For more information about specific versions of software requirements, see [System Requirements on page 13](#).



**Note:** Depending on your testing workload, HCL OneTest™ Server might require more resources. You must use the entire disk space and set up Logical Volume Manager (LVM) by using the `ext4` file system. If your organization requires application data to be stored in a separate partition, then you can create a mount point at `/var/lib/rancher/k3s/storage/` with at least 128 GB capacity.

## Backup of user data

If you want to upgrade the product from a previous version, then you must perform one of the following tasks based on the existing version of the server software:

- Back up the data from V10.0.2, Fix Pack 1 or earlier. See [Backing up the user data from a previous release](#).
- Back up the data from V10.1.0 or later. See [Backing up the server data on Ubuntu on page 92](#).



**Note:** You must select the existing version of HCL OneTest™ Server from the drop-down list to view the procedure for backing up and restoring the user data for your version because the instructions differ for different versions.

## Mandatory software

You must install the following mandatory software:

- Helm V3.5.2. For more information, refer to the [Helm](#) documentation.



**Note:** The Helm command must be in one of the directories in your `PATH` environment variable.

- OpenSSH Server. For more information, refer to the Installation section in the [OpenSSH Server](#) documentation.



**Note:** If you use ssh to log in to the computer, then you can skip this step because OpenSSH is already installed on your computer.

## k3s environment

You must set up the Kubernetes environment (k3s) along with other configurations such as firewall, Jaeger, and Prometheus server. See [Setting up a Kubernetes environment \(k3s\) on Ubuntu on page 60](#).

## Service virtualization through Istio



**Disclaimer:** This release contains access to the Istio virtualized services feature in HCL OneTest™ Server as a Tech Preview. The Tech Preview is intended for you to view the capabilities of virtualized services offered by HCL OneTest™ Server, and to provide your feedback to the product team. You are permitted to use the information only for evaluation purposes and not for use in a production environment. HCL provides the information without obligation of support and "as is" without warranty of any kind.

To provide your feedback, you must perform the following tasks:

1. Sign up or Log in to the [Ideas portal](#).
2. Click **Add a new idea** to provide your feedback on the Tech Preview features.
3. Select **OneTest TechPreview Programs** in the **Choose a workspace for this idea** field.
4. Provide all the necessary details, and then click **Share Idea**.

The default configuration does not enable service virtualization through Istio. To use the service virtualization feature, you must configure it appropriately. You can virtualize the *bookinfo* sample application. You must install this application in the `bookinfo` namespace. For more information about the *bookinfo* application, refer to the [Istio](#) documentation.

For more information about how to enable service virtualization through Istio, see [Setting up a Kubernetes environment \(k3s\) on Ubuntu on page 60](#).

### Collection of usage metrics

If you want HCL OneTest™ Server to display the usage statistics of the virtual service instances, you must install the Prometheus server at the time of server installation.

## Setting up a Kubernetes environment (k3s) on Ubuntu

You must set up a Kubernetes environment (k3s) so that you can install the server software on an Ubuntu server. You can use the script that is provided with HCL OneTest™ Server to set up a Kubernetes environment (k3s).

### Before you begin

You must have completed the following tasks:

- Installed the following software:
  - OpenSSH server
  - Helm



- Ensured that your computer has a Domain Name Server (DNS) resolvable host name to resolve the host name into a machine-readable IP address.
- Copied a **Secret key** from the Harbor repository.

### About this task

As part of the Kubernetes environment (k3s) set up, you can install the following software:

- **Jaeger:** By using this software, you can trace test logs and Jaeger-based reports when you run tests.
- **Prometheus server:** By using this software, you can monitor your system resources by using metrics data that are collected by a Prometheus server.

1. Log in to the Ubuntu server using an SSH session.
2. Add the software registry to Helm by running the following command:

```
helm repo add hclsoftware https://hclcr.io/chartrepo/ot --username {okta-email-address}
--password {harbor-cli-secret}
```



**Note:** You must replace `{okta-email-address}` with the user name of the Harbor repository and replace `{harbor-cli-secret}` with the secret key that you copied from the Harbor repository.

If the user name contains any special characters, such as \$, you must enclose it within single quotes.

3. Run the following command to get the latest updates from the repository:

```
helm repo update
```

4. Run the following commands to fetch the scripts that are used to install Kubernetes:

```
helm pull --untar hclsoftware/hcl-onetest-base --version 5.1020.0
chmod +x hcl-onetest-base/*.sh
```

5. Perform one of the steps described in the following table to install the k3s Kubernetes environment based on your requirement:

Step description	Step no
To install the Kubernetes environment (k3s)	Perform <a href="#">5.a on page 62</a>
To install the Kubernetes environment (k3s) by overriding the default name	Perform <a href="#">5.b on page 62</a>
To install the Kubernetes environment (k3s) and enable the service virtualization through Istio, a Tech Preview feature	Perform <a href="#">5.c on page 62</a>

Step description	Step no
To install the Kubernetes environment (k3s) by overriding the default name and enable the service virtualization through Istio, a Tech Preview feature	Perform <a href="#">5.d</a> on <a href="#">page 62</a>

- a. Run the following commands to install the Kubernetes with a default domain name that is either based on IP address or fully qualified hostname:

```
#Run the following commands if you are on Ubuntu 18.04
$ cd hcl-onetest-base
$ sudo ./ubuntu-init.sh

#Run the following commands if you are on Ubuntu 20.04
$ cd hcl-onetest-base
$ sudo HOME=$HOME ./ubuntu-init.sh
```

- b. Run the following commands to install the Kubernetes environment (k3s) by overriding the default name:

```
$ cd hcl-onetest-base
$ sudo INGRESS_DOMAIN={onetest}.myorg.com HOME=$HOME ./ubuntu-init.sh
```

- c. Run the following command to install the Kubernetes environment (k3s) and to enable the service virtualization through Istio, a Tech Preview feature:

```
#Run the following commands if you are on Ubuntu 18.04
$ cd hcl-onetest-base
$ sudo ./ubuntu-init.sh --demo

#Run the following commands if you are on Ubuntu 20.04
$ cd hcl-onetest-base
$ sudo HOME=$HOME ./ubuntu-init.sh --demo
```

- d. Run the following command to install the Kubernetes environment (k3s) by overriding the default name and to enable the service virtualization through Istio, a Tech Preview feature:

```
$ cd hcl-onetest-base
$ sudo INGRESS_DOMAIN={onetest}.myorg.com HOME=$HOME ./ubuntu-init.sh --demo
```

where:

- `{onetest.}` is a sub-domain name that you specified for the server. For example, `testenv.`



**Note:** The sub-domain must consist of lowercase alphanumeric characters, “-” (hyphen) or “.” (period). Also, the value must start and end with an alphanumeric character.

- `myorg.com` is the domain name of your organization. For example, `hcl.com`

You can access the product through a web browser by using any of the following methods:

- **Fully Qualified Hostname:** When the server is configured, you can use `hostname -f` command to get the fully qualified host name defined in the DNS to access HCL OneTest™ Server. For example, `{onetest}.myorg.com`
- **IP address:** You can use the IP address to access HCL OneTest™ Server when you cannot create a specific DNS record for the server. For example, `ip-address.nip.io`

## Result

On completion of the `ubuntu-init.sh` script, a namespace with the name `test-system` is created to install the server software and the output displays the following information on the command-line interface:

- The `INGRESS_DOMAIN` that is in use. This is the URL from where you can access HCL OneTest™ Server. You must use this value for the **global.hclOneTestIngressDomain** parameter in step 4 on page 65 in the server installation topic.
- The DNS information that the Kubernetes cluster uses to resolve names.
- Certificate Authority (CA) that you must import into the browser to prevent certificate errors.

You can run the following command to get the certificate from the system:

```
kubectl get secret ingress -n test-system -o jsonpath={.data.ca\\.crt} | base64 -d
```

- Instructions to confirm whether the Kubernetes environment has started.

You can refer to the **Results** section for more details on how to verify that the Kubernetes environment has started.

6. Perform one of the following options to configure a firewall:

### Choose from:

- Run the following script to configure the firewall that allows traffic on `cnio` and port 443:

```
#Run the following command if you are on Ubuntu 18.04
$ sudo ./ubuntu-firewall.sh

#Run the following command if you are on Ubuntu 20.04
$ sudo HOME=$HOME ./ubuntu-firewall.sh
```



**Note:** You must consult your network administrator before you run this script, and confirm whether the firewall is compatible with your corporate policy.

- Update the firewall that allows traffic on `cnio` and port 443, if your Ubuntu server is already configured with the firewall.

7. **Optional:** Run the following command to enable the Jaeger traces for performance and Web UI tests:

```
./service.sh expose jaeger
```



**Note:** If you do not enable Jaeger, HCL OneTest™ Server produces text output in a microservice log file instead of the Jaeger traces when you run performance and Web UI test assets.



**Important:** The Jaeger traces are not protected, thus, any information logged into the Jaeger server might be accessible by anyone who has or discovers the `<server-url>/jaeger` URL.

8. **Optional:** Run the following command to enable the Prometheus server to monitor your system resources by using metrics data:

```
./service.sh expose prometheus
```



**Important:** The Prometheus metrics are not protected, thus, any information logged into the Prometheus server might be accessible by anyone who has or discovers the `<server-url>/prometheus` URL.

## Results

You have set up the Kubernetes environment on Ubuntu.

## What to do next

- You must log in to the server host system again after the installation process is completed so that changes to the group membership are applied.



**Note:** You can run the `kubectl get pods -A` command to verify that the Kubernetes environment is working. After a while, the status of the pods display a `Running` or `Complete` state.

- You can install the server software. See [Installing the server software on Ubuntu by using k3s on page 64](#).

---

### Related information

[Jaeger Operator](#)

[Prometheus server](#)

[Troubleshooting Guide on page 987](#)

## Installing the server software on Ubuntu by using k3s

You can install HCL OneTest™ Server on the Ubuntu server that has a Kubernetes environment to run functional, integration, and performance tests. HCL OneTest™ Server combines test data, test environments, and test runs and reports into a single, web-based browser for testers and non-testers.

### Before you begin

You must have performed the following tasks:

- Completed the tasks provided in the Prerequisites section. See [Prerequisites for installing the server software on Ubuntu on page 58](#).
- Set up the Kubernetes environment (k3s). See [Setting up a Kubernetes environment \(k3s\) on Ubuntu on page 60](#).
- Logged in to the server host system again after you completed the k3s environment setup.
- Copied the following values to be used during the installation process:
  - The **Secret key** from the Harbor repository.
  - The INGRESS\_DOMAIN value that is displayed on completion of the `ubuntu-init.sh` script.

1. Log in to the Ubuntu server using an SSH session.
2. Run the following command to get the latest updates from the repository:

```
helm repo update
```

3. Create a Secret to pull the images that are used by HCL OneTest™ Server by running the following command:

```
kubectl create secret docker-registry hclcr.io \
-n test-system \
--docker-server=hclcr.io \
--docker-username={okta-email-address} \
--docker-password={harbor-cli-secret} \
--docker-email=example@abc.com
```



#### Notes:

- You must replace `{okta-email-address}` with the user name of the Harbor repository and replace `{harbor-cli-secret}` with the secret key that you copied from the Harbor repository.

If the user name contains any special characters, such as \$, you must enclose it within single quotes.

- You can replace `example@abc.com` with the email address of the administrator if required.

4. Perform the following steps to install the server software:

- a. Run the following command to get the latest updates from the repository:

```
helm repo update
```


- b. Run the following command to retrieve the charts required to install the server software:

```
helm pull --untar hclsoftware/hcl-onetest-server --version 5.1020.0
```

- c. Run the following command to provide the execution permission for the scripts that are available in the `files` directory:

```
chmod +x hcl-onetest-server/files/*.sh
```

- d. Perform one of the steps described in the following table to install the server software based on your requirement:

Step description	Step no
To install the server software	Perform <a href="#">4.d.i on page 66</a>
To install the server software and enable the service virtualization through Istio, a Tech Preview feature	Perform <a href="#">4.d.ii on page 66</a> and <a href="#">4.d.iv on page 67</a>
To install the server software, enable the service virtualization through Istio, a Tech Preview feature, and to use another instance of the Jaeger UI	Perform <a href="#">4.d.iii on page 67</a> and <a href="#">4.d.iv on page 67</a>
To install the server software and to use another instance of the Jaeger UI   <b>Note:</b> You must perform this step only if you want to use another instance of Jaeger than the one you installed during the setting up of the Kubernetes k3s environment.	Perform <a href="#">4.d.v on page 67</a>
To install the server software and enable performance test runs on the static agents	Perform <a href="#">4.d.vi on page 67</a>

- i. Run the following command to install the server software:

```
helm install {my-ots} ./hcl-onetest-server -n test-system \
-f hcl-onetest-server/values-k3s.yaml \
--set global.hclOneTestIngressDomain={my-ingress-dns-name} \
--set global.hclFlexnetURL=https://hclsoftware.compliance.flexnetoperations.com \
--set global.hclFlexnetID={cloud-license-server-id} \
--set global.hclImagePullSecret=hclcr.io \
--set global.hclOneTestPasswordAutoGenSeed={password-seed}
```

- ii. Run the following command to install the server software and to enable Istio service virtualization, a Tech Preview feature:

```
helm install {my-ots} ./hcl-onetest-server -n test-system \
-f hcl-onetest-server/values-k3s.yaml \
--set global.hclOneTestIngressDomain={my-ingress-dns-name} \
--set global.hclFlexnetURL=https://hclsoftware.compliance.flexnetoperations.com \
--set global.hclFlexnetID={cloud-license-server-id} \
--set global.hclImagePullSecret=hclcr.io \
```

```
--set global.hclOneTestPasswordAutoGenSeed={password-seed} \
-f hcl-onetest-server/values-k3s-demo.yaml
```

- iii. Run the following command to install the server software, to enable Istio service virtualization, a Tech Preview feature, to use another instance of Jaeger than the one you installed during the setting up of the Kubernetes k3s environment:

```
helm install {my-ots} ./hcl-onetest-server -n test-system \
-f hcl-onetest-server/values-k3s.yaml \
--set global.hclOneTestIngressDomain={my-ingress-dns-name} \
--set global.hclFlexnetURL=https://hclsoftware.compliance.flexnetoperations.com \
--set global.hclFlexnetID={cloud-license-server-id} \
--set global.hclImagePullSecret=hclcr.io \
--set global.hclOneTestPasswordAutoGenSeed={password-seed} \
-f hcl-onetest-server/values-k3s-demo.yaml
--set global.jaegerDashboard.externalURL={my-jaeger-dashboard-url} \
--set global.jaegerAgent.internalHostName=localhost
```

- iv. Run the following command to enable service virtualization through Istio, a Tech Preview feature in the specific namespace:

```
kubectl create rolebinding istio-virtualization-enabled -n
bookinfo --clusterrole={my-ots}-execution-istio-test-system
--serviceaccount=test-system:{my-ots}-execution
```

Where, {my-ots} is the name of the release that is provided during the installation of the server software.



**Note:** When you uninstall the chart, the manually created role bindings are not deleted from the namespace. You can run the following command to delete the role bindings:

```
kubectl delete rolebinding istio-virtualization-enabled -n bookinfo
```

- v. Run the following command to install the server software and to use another instance of Jaeger than the one you installed during the setting up of the Kubernetes k3s environment:

```
helm install {my-ots} ./hcl-onetest-server -n test-system \
-f hcl-onetest-server/values-k3s.yaml \
--set global.hclOneTestIngressDomain={my-ingress-dns-name} \
--set global.hclFlexnetURL=https://hclsoftware.compliance.flexnetoperations.com \
--set global.hclFlexnetID={cloud-license-server-id} \
--set global.hclImagePullSecret=hclcr.io \
--set global.hclOneTestPasswordAutoGenSeed={password-seed} \
--set global.jaegerDashboard.externalURL={my-jaeger-dashboard-url} \
--set global.jaegerAgent.internalHostName=localhost
```

- vi. Run the following command to install the server software and to enable performance test runs on the static agents:

If you have performance tests that distribute workload across different workstations, then you can add HCL OneTest™ Performance Agents to your projects to run those tests on remote workstations. To run performance test assets such as VU schedule or Rate schedule on a

static agent, you must run the following command to enable the performance test runs on the static agents.

```
helm install {my-ots} ./hcl-onetest-server -n test-system \
-f hcl-onetest-server/values-k3s.yaml \
--set global.hclOneTestIngressDomain={my-ingress-dns-name} \
--set global.hclFlexnetURL=https://hclsoftware.compliance.flexnetoperations.com \
--set global.hclFlexnetID={cloud-license-server-id} \
--set global.hclImagePullSecret=hclcr.io \
--set networkPolicy.enabled=false \
--set global.hclOneTestPasswordAutoGenSeed={password-seed} \
--set networkPolicy.enabled=false
```

You must substitute the value of the following variables with the actual value in the command:

- `{my-ots}` with the release name of your choice.



**Note:** The release name must consist of alphanumeric characters that are in lowercase or “-” (hyphen). The release name must also start with an alphabetic character and end with an alphanumeric character. For example, *my-org* or *abc-123*.

- `{my-ingress-dns-name}` with the `INGRESS_DOMAIN` value that is displayed on completion of the `ubuntu-init.sh` script.
- **Optional:** `{cloud-license-server-id}` with the ID of the License Server for the initial team space, if you want to set the license for the first time.



**Important:** If you want to upgrade the product from the previous version, you must configure the value of **License Server ID** from the **Team Space License Configuration** page when the installation of the server is complete.

- `{password-seed}` with a value of your choice.



**Important:** This password seed is used to create several default passwords for the server. You must store the password seed securely. When you install the server software by using the backup of the user data, you can reuse the password seed. You can use this seed to restore the backed-up files either on the current or later versions of the server software.

- `{my-jaeger-dashboard-url}` with the URL of the Jaeger server.

5. **Optional:** Run the following command to remove a job that is used to initialize the PostgreSQL database during the installation of the server software:

```
kubectl delete job {my-ots}-postgresql-init -n test-system
```

6. **Optional:** Perform the following steps to migrate data into HCL OneTest™ Server, if you upgraded the server software from the previous version (V10.1.0, V10.1.1, or V10.1.2):



- a. Run the following script from the `hcl-onetest-server/files` directory to create a directory that contains metadata related to the Persistent Volume Claims and their Persistent Volumes:

```
migrate.sh create-pvcs -n test-system {my-ots}
```

- b. Run the following script from the `hcl-onetest-base` directory to back up the data:

```
sudo backup.sh create-pvc-links -v ~/migration-pvc-links
```

- c. Run the following command to stop the cluster and HCL OneTest™ Server:

```
k3s-killall.sh
```

- d. Run the following script from the `hcl-onetest-base` directory to restore the backed-up data:

```
sudo backup.sh restore -v ~/migration-pvc-links <backup-file-name> --release
```



**Note:** You must replace `<backup-file-name>` with the name of the backed-up file that you saved.

- e. Run the following command to restart Kubernetes and to start HCL OneTest™ Server:

```
sudo systemctl start k3s
```

- f. Run the following script from the `hcl-onetest-server/files` directory to merge the data into the server:

```
migrate.sh merge-dbs -n test-system {my-ots}
```

- g. Run the following command to remove the resources that were created during the migration process:

```
migrate.sh delete-temp-resources -n test-system {my-ots}
```

**7. Optional:** Perform the following steps to restore the backed-up user data from V10.1.3 to the latest version:

- a. Run the following script from the `hcl-onetest-base` directory to create a directory that contains metadata related to the Persistent Volume Claims and their Persistent Volumes:

```
sudo backup.sh create-pvc-links
```

- b. Stop the Kubernetes cluster by running the `k3s-killall.sh` command.

- c. Restore the backed-up user data by running the following commands:

```
sudo ./backup.sh restore [options] <backup-file-name>
```



**Note:** You must replace `<backup-file-name>` with the name of the backed-up file that you saved.

You can use the following parameters along with the restore command:

- `--namespace` or `-n`: Use this parameter to restore a specific namespace from the backup file. If you do not mention the namespace, then volumes from all the namespaces in the backup file are restored. You can map one namespace to another namespace by using colon (:). The syntax is:

```
--namespace <name of the namespace> [:<target-namespace>]
```

- `--release` or `-r`: Use this parameter if the Helm release name of the server to which the backup is being restored is different than the Helm release name of the server where the backup was taken. The syntax is:

```
--release <backup-release>:<target-release>
```

- `--volumes` or `-v`: Use this parameter to specify the directory path of the Volumes. The syntax is:

```
--volumes <path-of-the-directory>
```

- `-k` or `--confirm`: Use this parameter to skip the confirmation step.

d. Restart the Kubernetes cluster by running the `systemctl start k3s` command to start HCL OneTest™ Server.

e. Run the following script from the `hcl-onetest-server/files` directory to create all the missing databases:

```
migrate.sh create-missing-dbs {my-ots}
```



**Note:** You can expect that the output of the command to contain a number of errors pertaining to objects that already exist. They do not indicate a problem with the execution of the script.

8. Run the following command to verify and test the installed server software:

```
$ helm test {my-ots} -n test-system
```

where `{my-ots}` is the name of the release that was provided during the installation of the server software.

## Results

On completion of the installation of server software, the output displays the following information on the command-line interface:

- Instructions to access Keycloak to manage and authenticate users.

The user name can be `keycloak` and the password can be retrieved by running the following command:

```
kubectrl get secret -n namespace onetest-keycloak-postgresql -o jsonpath="{.data.password}" | base64 --decode; echo
```

where:

- *onetest* is a sub-domain name that you selected for the server.
- *namespace* is the name of the namespace that you created.
- The URL to access the HCL OneTest™ Server UI.

### What to do next

You can perform certain tasks as a *Server Administrator*. See [Configuration of the server software on page 104](#).

---

Related information

[Helm documentation](#)

[Troubleshooting Guide on page 987](#)

## Installation of the server software on Azure Kubernetes Service

You can find information about the tasks that you can perform to install HCL OneTest™ Server on the Azure Kubernetes Service (AKS) platform.

### Prerequisites for installing the server software on Azure Kubernetes Service

You must complete certain tasks before you install HCL OneTest™ Server on the Azure Kubernetes Service (AKS) platform.

The following sections describe each prerequisite in detail:

- [Internet access on page 71](#)
- [Harbor repository credentials on page 72](#)
- [Azure subscription on page 72](#)
- [Azure Kubernetes Service on page 72](#)
- [Mandatory software on page 72](#)
- [Azure resource provider on page 73](#)

#### Internet access

You must have access to the internet to install HCL OneTest™ Server.

## Harbor repository credentials

- **New customers:**

You must create a [support ticket](#) to get the credentials that are required to access the product binaries from the Harbor repository. For more information, refer to [How to create an HCL Support case](#).

- **Existing customers:**

You must use your existing Harbor repository credentials to access the product binaries.

## Azure subscription

You must have an active Azure subscription along with an **Owner**, or **Contributor**, and **User-Access Administrator** role on the Azure subscription.



**Note:** You can create a subscription from [Microsoft Azure](#) if you do not have a valid subscription.

## Azure Kubernetes Service

You must set up the AKS cluster with Kubernetes V1.19. For more information about Azure Kubernetes Service and deployment options, refer to [Azure Kubernetes Service](#) documentation.

## Mandatory software

You must install the following mandatory software:

- Git Bash. For more information, refer to the [Git](#) documentation.
- Helm V3.5.2. For more information, refer to the [Helm](#) documentation.



**Note:** The Helm command must be in one of the directories in your **PATH** environment variable.

- Azure command-line interface (CLI) V2.0.64. For more information, refer to the [Azure CLI](#) documentation.



**Note:** You can run the `az version` command to find the version and dependent libraries that are installed.

- kubectl tool V1.19. For more information, refer to the [Kubernetes](#) documentation.
- NGINX Ingress Controller V3.23.0. See [Installing NGINX Ingress Controller on page 74](#).
- **Optional:** Jaeger operator V1.21 to trace test logs and Jaeger-based reports when you run tests. For more information, refer to [Jaeger](#) documentation.

## Azure resource provider

Register an Azure resource provider so that the AKS cluster can provision to use Persistent Volumes with Azure Files for the ReadWriteOnceMany (RWX) storage class. You can run the following command to enable the storage for the RWX storage class:

```
az provider register -n Microsoft.Storage
```

For more information about Azure resource providers and types, refer to [Azure Kubernetes Service](#) documentation.

---

Related information

[NGINX Ingress Controller](#)

## Cluster autoscaler on Azure Kubernetes Service

You can enable the cluster autoscaler feature on Azure Kubernetes Service (AKS) to manage the cost of the cluster effectively.

You can use the cluster autoscaler with multiple node pools that are enabled. Depending on your AKS setup, when the nodes are not in use, the nodes of the AKS cluster are automatically shut down to minimize the runtime costs of the AKS cluster without risking the performance of HCL OneTest™ Server.

When you create an AKS cluster, it contains a single node pool as the default node pool. You can use the `az aks node pool` command to add a node pool to your existing AKS cluster. You can then use the `az aks update` command to enable and configure the cluster autoscaler on the node pool for the existing AKS cluster.

You can use the following commands to add a node pool and enable the cluster autoscaler on the node pool for the existing AKS cluster:

```
az aks nodepool add -g <resource_group> -n wrk0 \
--cluster-name <aks_cluster_name> \
--kubernetes-version <k8s_version> \
--node-vm-size Standard_D2ds_v4 \
--enable-cluster-autoscaler \
--node-count 0 --min-count 0 --max-count 2 \
--labels purpose=test-execution \
--node-taints purpose=test-execution:NoSchedule

az aks update \
-g <resource_group> -n <aks_cluster_name> \
--cluster-autoscaler-profile scale-down-unneeded-time=1m scale-down-delay-after-add=1m
```

 **Important:**



- You must replace `<resource_group>` and `<aks_cluster_name>` with the name of the resource group and the name of the AKS cluster in the command.
- You must add the following command in Step 8.a on page 80 or 8.b on page 80 when you install the server software to run test assets on the created node pool:

```
-f hcl-onetest-server/values-dedicated-nodes.yaml
```

After you run the commands, you have completed the following actions:

- Added a user node pool called `wrk0`
- Enabled the cluster autoscaler feature
- Updated the default values of **scale-down-unnneeded-time** and **scale-down-delay-after-add** in the cluster-wide autoscaler profile to `1m`

When you install the server software by enabling the cluster autoscaler feature on your AKS cluster, a new container is created only to run test assets. You can run the `kubectl get pods` command to see the status of the execution pod, which is displayed as `Pending`.

You can describe the pod by running the `kubectl describe pod <pod_name> -n <namespace>` command to see a scale event when test assets are running from HCL OneTest™ Server.

When the test asset run is complete, the execution pods shut down automatically. Thus, you can reduce the cost of the cluster when the node is not in use. You can then run the `kubectl get nodes` command to verify the number of nodes that are currently in the `Ready` state.

---

#### Related information

[Automatically scale a cluster to meet application demands on Azure Kubernetes Service \(AKS\)](#)

[Create and manage multiple node pools for a cluster in Azure Kubernetes Service \(AKS\)](#)

## Installing NGINX Ingress Controller

You must install NGINX Ingress Controller to terminate Transport Layer Security (TLS) and route traffic to HCL OneTest™ Server. You can install NGINX Ingress Controller by pulling the images from Azure Container Registry (ACR) when you do not want to pull the images from the public registries.

### Before you begin

You must have completed the following tasks:

- Set up the Azure Kubernetes Service (AKS) cluster with Kubernetes V1.19. For more information, refer to [Azure Kubernetes Service](#) documentation.
- Installed the following software:
  - Git Bash. For more information, refer to the [Git](#) documentation.
  - Helm V3.5.2. For more information, refer to the [Helm](#) documentation.



**Note:** The Helm command must be in one of the directories in your **PATH** environment variable.

- kubectl tool V1.19. For more information, refer to the [Kubernetes](#) documentation.
- Azure command-line interface (CLI) V2.0.64. For more information, refer to the [Azure CLI](#) documentation.



**Note:** You can run the `az version` command to find the version and dependent libraries that are installed.

- Logged in to the AKS cluster. You can run the `az login` command to log in to AKS interactively.
- Set your subscription to active azure subscription. You can run the `az account set --subscription <subscription_name>` command to set your active subscription.
- Configured the location of your AKS cluster. You can run the `az configure --defaults location=<location_name>` command to set your location of your AKS cluster.



**Tip:** You can run the following command to get the list of all the available locations:

```
az account list-locations --query "[].{DisplayName:displayName, Name:name}" -o table
```

### About this task

You must replace the following variables with the actual value in this procedure:

- `<resource_group>` with the name of the resource group that you created during the creation of the AKS cluster.
- `<azure_cluster>` with the name of the Azure cluster that you created during the creation of the AKS cluster.
- `<azure_container_registry>` with the name of ACR that you created during the AKS setup.

- `<ip_address>` with the internal IP address to use with the ingress controller.



**Note:** You must ensure that the IP address is not already in use within your virtual network.

1. Run the following command to configure kubectl to connect to the AKS cluster:

```
az aks get-credentials -g <resource_group> -n <azure_cluster>
```



**Note:** You can verify the connection to the AKS cluster by running the kubectl get nodes command that displays a list of the cluster nodes.

2. Add the NGINX Ingress Controller repository to Helm by running the following command:

```
helm repo add ingress-nginx https://kubernetes.github.io/ingress-nginx --force-update
```

3. **Optional:** Perform the following steps if you want to install NGINX Ingress Controller from ACR:

- a. Run the following command to list the images to move into ACR:

```
for image in $(helm template ingress-nginx ingress-nginx/ingress-nginx \
  | sed -r -n 's/ *image: *("[^"]*)" / \1/p' | uniq)
do
  echo "az acr import -n <azure_container_registry> \\"
  echo "  --source ${image}/*?@/@" \\"
  tag=${image}/*/*
  echo "  --image ${tag}/*/*"
done
```

#### Result

The terminal displays a block of commands.

- b. Run the commands that are displayed on the terminal after you perform Step 3.a on page 76 to import an image to ACR.

4. Create a namespace in which you want to install NGINX Ingress Controller by running the following command:

```
kubectl create namespace ingress-nginx
```



**Remember:** `ingress-nginx` is the name of the namespace. If you created a namespace by using a different value, then you must use that value in place of the `ingress-nginx` in all the instances in this procedure.

5. Run the following command to label the namespace so that network policies facilitate traffic to pods of HCL OneTest™ Server:

```
kubectl label namespace ingress-nginx app.kubernetes.io/name=ingress-nginx
```

6. Perform one of the following steps to install NGINX Ingress Controller based on your requirement:



- a. Run the following command to install the NGINX Ingress Controller by referencing the images from ACR:

```
helm install ingress-nginx ingress-nginx/ingress-nginx \
-n ingress-nginx \
--set
  controller.admissionWebhooks.patch.image.repository="<azure_container_registry>.azurecr.
io/jettech/kube-webhook-certgen" \
--set controller.admissionWebhooks.patch.nodeSelector."beta\.kubernetes\.io/os"=linux \
--set
  controller.image.repository="<azure_container_registry>.azurecr.io/ingress-nginx/controll
er" \
--set controller.nodeSelector."beta\.kubernetes\.io/os"=linux \
--set
  controller.service.annotations."service\.beta\.kubernetes\.io/azure-load-balancer-interna
l"=true \
--set controller.service.loadBalancerIP=<ip_address>
```

- b. Run the following command to install the NGINX Ingress Controller by referencing the images from the public registry:

```
helm install ingress-nginx ingress-nginx/ingress-nginx \
-n ingress-nginx \
--set controller.admissionWebhooks.patch.nodeSelector."beta\.kubernetes\.io/os"=linux \
--set controller.nodeSelector."beta\.kubernetes\.io/os"=linux \
--set
  controller.service.annotations."service\.beta\.kubernetes\.io/azure-load-balancer-interna
l"=true \
--set controller.service.loadBalancerIP=<ip_address>
```

7. Run the following command to verify that external-IP is assigned to NGINX Ingress Controller:

```
kubectl get svc -n ingress-nginx ingress-nginx-controller
```



**Note:** The status of EXTERNAL-IP displayed as `Pending` for few minutes. If the status does not change to `$INGRESS_IP`, then that indicates you have a permissions problem. You can run the following command to investigate the problem:

```
kubectl describe svc -n ingress-nginx ingress-nginx-controller
```

## Results

You have installed NGINX Ingress Controller.

## What to do next

You can install the server software. See [Installing the server software on Azure Kubernetes Service on page 78](#).

---

### Related information

[Azure Kubernetes Service documentation](#)

## Installing the server software on Azure Kubernetes Service

You can install HCL OneTest™ Server on Azure Kubernetes Service (AKS) that has a Kubernetes environment to run functional, integration, and performance tests. HCL OneTest™ Server combines all the capability into a single web-based browser for testers and non-testers. The capabilities can be test runs, test data, test environment, or test reports.

### Before you begin

You must have performed the following tasks:

- Completed the tasks provided in the Prerequisites section. See [Prerequisites for installing the server software on Azure Kubernetes Service on page 71](#).
- Copied the **Secret key** from the Harbor repository to be used during the installation process.
- Logged in to the AKS cluster. You can run the `az login` command to log in to AKS interactively.
- Set your subscription to active azure subscription. You can run the `az account set --subscription <subscription_name>` command to set your active subscription.
- Configured the location of your AKS cluster. You can run the `az configure --defaults location=<location_name>` command to set your location of your AKS cluster.



**Tip:** You can run the following command to get the list of all the available locations:

```
az account list-locations --query "[].{DisplayName:displayName, Name:name}" -o table
```

### About this task

You must replace the following variables with the actual value in this procedure:

- `<resource_group>` with the name of the resource group that you created during the creation of the AKS cluster.
- `<azure_cluster>` with the name of the Azure cluster that you created during the creation of the AKS cluster.
- `<azure_container_registry>` with the name of the Azure Container Registry (ACR) that you created during the AKS setup.
- `<okta-email-address>` with the user name of the Harbor repository.
- `<harbor-cli-secret>` with the secret key that you copied from the Harbor repository.
- `<my-ingress-dns-name>` with the INGRESS\_DOMAIN value that is you provided during the installation of NGINX Ingress Controller.

1. Add the software registry to Helm by running the following command:

```
helm repo add hclsoftware https://hclcr.io/chartrepo/ot --username {okta-email-address}
--password {harbor-cli-secret}
```



**Note:** You must replace `{okta-email-address}` with the user name of the Harbor repository and replace `{harbor-cli-secret}` with the secret key that you copied from the Harbor repository.

If the user name contains any special characters, such as `$`, you must enclose it within single quotes.

2. Run the following command to get the latest updates from the repository:

```
helm repo update
```

3. Run the following commands to fetch the scripts that are used to install HCL OneTest™ Server:

```
helm pull --untar hclsoftware/hcl-onetest-server --version 5.1020.0
chmod +x hcl-onetest-server/files/*.sh
```

4. Run the following command to configure kubectl to connect to the AKS cluster:

```
az aks get-credentials -g <resource_group> -n <azure_cluster>
```



**Note:** You can verify the connection to the AKS cluster by running the `kubectl get nodes` command that displays a list of the cluster nodes.

5. Run the following script from the `hcl-onetest-server/files` directory to pull the images of HCL OneTest™ Server from the Harbor repository to ACR:

```
PULL_ARGUMENTS="-u <okta-email-address> -p <harbor-cli-secret>" \
bash hcl-onetest-server/files/move-images.sh <azure_container_registry>.azurecr.io/hcl-onetest
hclcr.io/ot/hcl-onetest
```



**Tip:** You can verify that multiple manifests are associated with the image in the ACR by running the following commands:

```
az acr repository list -n <azure_container_registry> -otsv
az acr repository show-manifests -n <azure_container_registry> --repository \
$(az acr repository list -n <azure_container_registry> -otsv --query [0])
```

6. Create a namespace in which you want to install the server software by running the following command:

```
kubectl create namespace test-system
```



**Remember:** `test-system` is the name of the namespace. If you created a namespace by using a different value, then you must use that value in place of the `test-system` in all the instances in this procedure.

7. Run the following script from the `hcl-onetest-server/files` directory to enable Certificate Authority (CA) as trusted certificates and to create an ingress Secret:

```
./files/certificate.sh -n test-system -s ingress <my-ingress-dns-name>
```

8. Perform one of the steps described in the following table to install the server software based on your requirement:

Step description	Step no
To install the server software	Perform <a href="#">8.a on page 80</a>
To install the server software and enable Jaeger for performance and Web UI tests logs	Perform <a href="#">8.b on page 80</a>

- a. Run the following command to install the server software:

```
helm install {my-ots} ./hcl-onetest-server -n test-system \
-f hcl-onetest-server/values-k8s.yaml \
--set global.persistence.rwxStorageClass=azurefile \
--set global.hclOneTestIngressDomain={my-ingress-dns-name} \
--set global.hclFlexnetURL=https://hclsoftware.compliance.flexnetoperations.com \
--set global.hclFlexnetID={cloud-license-server-id} \
--set global.hclOneTestRegistry=<azure_container_registry>.azurecr.io/hcl-onetest \
--set global.hclOneTestPasswordAutoGenSeed={password-seed}
```

- b. Run the following command to install the server software and to enable Jaeger for performance and Web UI tests logs:

```
helm install {my-ots} ./hcl-onetest-server -n test-system \
-f hcl-onetest-server/values-k8s.yaml \
--set global.persistence.rwxStorageClass=azurefile \
--set global.hclOneTestIngressDomain={my-ingress-dns-name} \
--set global.hclFlexnetURL=https://hclsoftware.compliance.flexnetoperations.com \
--set global.hclFlexnetID={cloud-license-server-id} \
--set global.hclOneTestRegistry=<azure_container_registry>.azurecr.io/hcl-onetest \
--set global.hclOneTestPasswordAutoGenSeed={password-seed} \
--set-string execution.annotations.sidecar\\.jaegertracing\\.io/inject=true \
--set global.jaegerAgent.internalHostName=localhost \
--set global.jaegerDashboard.externalURL={my-jaeger-dashboard-url}
```

You must substitute the value of the following variables with the actual value in the command:

- `{my-ots}` with the release name of your choice.



**Note:** The release name must consist of alphanumeric characters that are in lowercase or "-" (hyphen). The release name must also start with an alphabetic character and end with an alphanumeric character. For example, *my-org* or *abc-123*.

- `{my-ingress-dns-name}` with the INGRESS\_DOMAIN value that is you provided during the installation of NGINX Ingress Controller.
- **Optional:** `{cloud-license-server-id}` with the ID of the License Server for the initial team space, if you want to set the license for the first time.

**!** **Important:** If you want to upgrade the product from the previous version, you must configure the value of **License Server ID** from the **Team Space License Configuration** page when the installation of the server is complete.

- `{password-seed}` with a value of your choice.

**!** **Important:** This password seed is used to create several default passwords for the server. You must store the password seed securely. When you install the server software by using the backup of the user data, you can reuse the password seed. You can use this seed to restore the backed-up files either on the current or later versions of the server software.

- `{my-jaeger-dashboard-url}` with the URL of the Jaeger server.

9. Run the following command to verify and test the installed server software:

```
$ helm test {my-ots} -n test-system
```

where `{my-ots}` is the name of the release that was provided during the installation of the server software.

## Results

You have installed the server software. The terminal displays the following information:

- The URL to access Keycloak to manage and authenticate users.

The user name can be `keycloak` and the password can be retrieved by running the following command:

```
kubectl get secret -n test-system {my-ots}-keycloak-postgresql -o jsonpath="{.data.password}" | base64 --decode; echo
```

where:

- `{my-ots}` is the name of the release that was provided during the installation of the server software.
- `test-system` is the name of the namespace that you created during the installation of the server software.

- The URL to access the HCL OneTest™ Server UI.

## What to do next

You can perform certain tasks as a *Server Administrator*. See [Configuration of the server software on page 104](#).

---

Related information

[Helm documentation](#)

[Azure Kubernetes Service documentation](#)

## Server software upgrade methods

When you want to use the enhanced functionalities of HCL OneTest™ Server, you must upgrade the product to the latest version of the server software.

You can upgrade the server software by using two methods:

- **In-place upgrade:** In this method, you can upgrade the server software in the same location that contained the V10.1.3 of the server software by using the `helm upgrade` command.
- **Upgrade (by new install):** In this method, you can upgrade the server software by uninstalling the existing version of the product, and then by installing the latest version of the product.



**Note:** The upgrade procedures are different based on the existing version of the software.



**Important:** On the Ubuntu platform, the **in-place upgrade** method is not available. You must use the **upgrade (by new install)** method to upgrade the server software.

To learn more about server software upgrade methods on platforms such as Ubuntu and Red Hat OpenShift, refer to the following topics:

### Server software upgrade on Red Hat OpenShift

When you want to use the latest version of the server software on Red Hat OpenShift, you must upgrade HCL OneTest™ Server from the previous version of the software.

On the Red Hat OpenShift platform, you can either use **in-place upgrade** or **upgrade (by new install)** methods depending on existing version of the server software.

Existing version of the software	Method to use
V10.1.3	In-place upgrade
V10.1.2 or earlier	Upgrade (by new install)

### Upgrading the server software by using the in-place upgrade method

When you want to install the latest version of the server software in the same location that contained the V10.1.3, then you can upgrade HCL OneTest™ Server by using the `helm upgrade` command.

#### Before you begin

You must have completed the following tasks:

- Informed users that HCL OneTest™ Server is offline temporarily during the upgrade process.
- Installed Helm V3.5.2. For more information, refer to the Installing a Helm chart on an OpenShift Container Platform cluster section in the [Red Hat OpenShift](#) documentation.
- Completed all test executions that are running on the existing version of HCL OneTest™ Server.
- Stopped all stub executions that are running on the existing version of HCL OneTest™ Server.
- Canceled any scheduled test runs that have a future date or time.
- Copied the **Secret key** from the Harbor repository.

**Optional.** You can back up the user data from the previous version of the product. If the upgrade fails, then you can use that backup file to restore it on V10.2.0. See [Backing up the server data on Red Hat OpenShift on page 90](#).

1. Log in to your OCP cluster as a cluster administrator by running the `oc login` command.
2. Add the software registry to Helm to access the server install charts by running the following command:

```
helm repo add hclsoftware https://hclcr.io/chartrepo/ot --username {okta-email-address}
--password {harbor-cli-secret}
```



**Note:** You must replace `{okta-email-address}` with the user name of the Harbor repository and replace `{harbor-cli-secret}` with the secret key that you copied from the Harbor repository.

If the user name contains any special characters, such as `$`, you must enclose it within single quotes.

3. Run the following command to get the latest updates from the repository:

```
helm repo update
```

4. Run the following command to retrieve the charts:

```
helm pull --untar hclsoftware/hcl-onetest-server --version 5.1020.0
```

5. Create a Secret to pull images that are used by HCL OneTest™ Server by running the following commands:

```
oc create secret docker-registry hclcr.io \
-n test-system \
--docker-server=hclcr.io \
--docker-username={okta-email-address} \
--docker-password={harbor-cli-secret} \
--docker-email=example@abc.com
```



**Notes:**



- You must replace `{okta-email-address}` with the user name of the Harbor repository and replace `{harbor-cli-secret}` with the secret key that you copied from the Harbor repository.

If the user name contains any special characters, such as \$, you must enclose it within single quotes.

- You can replace `example@abc.com` with the email address of the administrator if required.



**Remember:** `test-system` is the name of the namespace. If you created a namespace by using a different value, then you must use that value in place of the `test-system` in all the instances in this procedure.

6. Perform the following steps to upgrade the server software:

- a. Run the following commands to update the `runAsUser` and `fsGroup` to match the Security Context Constraints (SCC):

```
sed -i -e "s/runAsUser: 1001/runAsUser: $(oc get project test-system -oyaml \
| sed -r -n 's# *openshift.io/sa.scc.uid-range: *([0-9]*)/.*\#1#p')/g;
s/fsGroup: 1001/fsGroup: $(oc get project test-system -oyaml \
| sed -r -n 's# *openshift.io/sa.scc.supplemental-groups: *([0-9]*)/.*\#1#p')/g"
hcl-onetest-server/values-openshift.yaml
```

HCL OneTest™ Server is compatible with the restricted Security Context Constraint. You must run this command to ensure that the `runAsUser` and `fsGroup` strategies match with the SCC policy.

- b. Perform one of the steps described in the following table to upgrade the server software based on your requirement:

Step description	Step no
To upgrade the server software	Perform <a href="#">6.b.i on page 84</a>
To upgrade the server software and enable the OpenShift Service Mesh service virtualization through Istio, a Tech Preview feature	Perform <a href="#">6.b.ii on page 85</a> and <a href="#">6.b.iii on page 85</a>
To upgrade the server software and enable Jaeger for performance and Web UI tests logs	Perform <a href="#">6.b.iv on page 85</a>

- i. Run the following command to upgrade the server software:

```
helm upgrade {my-ots} ./hcl-onetest-server -n test-system \
-f hcl-onetest-server/values-openshift.yaml \
```



```
--set global.persistence.rwxStorageClass=rook-ceph-file
--set global.hclOneTestIngressDomain=onetest.{openshift-cluster-dns-name} \
--set global.hclImagePullSecret=hclcr.io \
--set global.hclOneTestPasswordAutoGenSeed={password-seed}
```



**Note:** You must use the `--set global.hclCertSecretOptional=false` parameter in the `helm upgrade` command only if you created an ingress Secret to store the CA during the previous installation of the server software.

- ii. Run the following command to upgrade the server software and to enable the OpenShift Service Mesh service virtualization feature:

```
helm upgrade {my-ots} ./hcl-onetest-server -n test-system \
-f hcl-onetest-server/values-openshift.yaml \
--set global.persistence.rwxStorageClass=rook-ceph-file
--set global.hclOneTestIngressDomain=onetest.{openshift-cluster-dns-name} \
--set global.hclImagePullSecret=hclcr.io \
--set global.hclOneTestPasswordAutoGenSeed={password-seed} \
-f hcl-onetest-server/values-openshift-demo.yaml
```

- iii. Run the following command to enable service virtualization through Istio, a Tech Preview feature in the specific namespace:

```
oc create rolebinding istio-virtualization-enabled -n
bookinfo --clusterrole={my-ots}-execution-istio-test-system
--serviceaccount=test-system:{my-ots}-execution
```

Where, {my-ots} is the name of the release that you provided during the installation of the server software.



**Note:** When you uninstall the chart, the manually created role bindings are not deleted from the namespace. You can run the following command to delete the role bindings:

```
oc delete rolebinding istio-virtualization-enabled -n bookinfo
```

- iv. Run the following command to upgrade the server software and to enable Jaeger for performance and Web UI tests logs:

```
helm upgrade {my-ots} ./hcl-onetest-server -n test-system \
-f hcl-onetest-server/values-openshift.yaml \
--set global.persistence.rwxStorageClass=rook-ceph-file
--set global.hclOneTestIngressDomain=onetest.{openshift-cluster-dns-name} \
--set global.hclImagePullSecret=hclcr.io \
--set global.hclOneTestPasswordAutoGenSeed={password-seed} \
--set-string execution.annotations.sidecar\\.jaegertracing\\.io/inject=true \
```

```
--set global.jaegerAgent.internalHostName=localhost \
--set global.jaegerDashboard.externalURL={my-jaeger-dashboard-url}
```



**Note:** You must use the `--set global.hclCertSecretOptional=false` parameter in the `helm upgrade` command only if you created an ingress Secret to store the CA during the previous installation of the server software.

7. Run the following command to verify and test the upgraded server software:

```
$ helm test {my-ots} -n test-system
```

where `{my-ots}` is the name of the release that was provided during the installation of the server software.

## Results

On the successful upgrade of HCL OneTest™ Server, the output displays the following information:

- Keycloak URL to manage and authenticate users.
- A URL to access the HCL OneTest™ Server UI.

---

Related information

[Troubleshooting Guide on page 987](#)

## Server software upgrade by using upgrade (by new install) method

When you do not have any specific requirement to upgrade the server software, you can uninstall the existing version of the server software, and then install the latest version.

Before you upgrade the server software, you must ensure that you have completed the following tasks:

- Installed Helm V3.5.2. For more information, refer to the Installing a Helm chart on an OpenShift Container Platform cluster section in the [Red Hat OpenShift](#) documentation.
- Completed all test executions that are running on the existing version of HCL OneTest™ Server.
- Stopped all stub executions that are running on the existing version of HCL OneTest™ Server.
- Canceled any scheduled test runs that have a future date or time.
- Informed users that HCL OneTest™ Server is offline temporarily during the upgrade process.

In upgrade (by new install) method, depending on the existing version of the software, you can upgrade the server software by using any of the following options.

To upgrade HCL OneTest™ Server (V10.1.0 or later) to latest version, you must perform these tasks in sequence as listed in the following table:


Tasks	Task description	More information
1	Back up the user data from the previous version of HCL OneTest™ Server.	<a href="#">Backing up the server data on Red Hat OpenShift on page 90</a>
2	Uninstall the previous version of HCL OneTest™ Server.	<a href="#">Uninstalling the server software from Red Hat OpenShift on page 101</a>
3	Install the latest version of the server software and then migrate the data as part of the installation process.	<a href="#">Installing the server software on Red Hat OpenShift on page 40</a>

To upgrade HCL OneTest™ Server V10.0.2, Fix Pack 1 or earlier to latest version, you must perform these tasks in sequence as listed in the following table:

Tasks	Task description	More information
1	Upgrade HCL OneTest™ Server V10.1.2.	<a href="#">Installation of the server software on Red Hat OpenShift</a>
2	Back up the user data from HCL OneTest™ Server V10.1.2.	<a href="#">Backing up the server data on Red Hat OpenShift on page 90</a>
3	Uninstall HCL OneTest™ Server V10.1.2.	<a href="#">Uninstalling the server software</a>
4	Install the latest version of the server software and then migrate the data as part of the installation process.	<a href="#">Installing the server software on Red Hat OpenShift on page 40</a>

## Server software upgrade on Ubuntu

When you want to use the latest version of the server software on Ubuntu, you must upgrade HCL OneTest™ Server from the previous version of the software.

 **Important:** On the Ubuntu platform, the **in-place upgrade** method is not available. You must use the **upgrade (by new install)** method to upgrade the server software.

Before you upgrade the server software, you must ensure that you have completed the following tasks:

- Installed Helm V3.5.2. For more information, refer to the [Helm](#) documentation.
- Completed all test executions that are running on the existing version of HCL OneTest™ Server.
- Stopped all stub executions that are running on the existing version of HCL OneTest™ Server.
- Canceled any scheduled test runs that have a future date or time.
- Informed users that HCL OneTest™ Server is offline temporarily during the upgrade process.

In upgrade (by new install) method, depending on the existing version of the software, you can upgrade the server software by using any of the following options.

To upgrade HCL OneTest™ Server (V10.1.0 or later) to latest version, you must perform these tasks in sequence as listed in the following table:

Tasks	Task description	More information
1	Back up the user data from the previous version of HCL OneTest™ Server.	<ul style="list-style-type: none"> <li>• For V10.1.0 or V10.1.1, refer to <a href="#">Backing up user data</a>.</li> <li>• For V10.1.2 or later, refer to <a href="#">Backing up user data</a>.</li> </ul>
2	Uninstall the previous version of HCL OneTest™ Server.	<a href="#">Uninstalling the server software from Ubuntu on page 102</a>
3	Install the latest version of the server software and then migrate the data as part of the installation process.	<a href="#">Installing the server software on Ubuntu by using k3s on page 64</a>

If your existing version of server software is V10.0.2, Fix Pack 1, or earlier, then you cannot directly upgrade to the latest version of the server software due to the steps involved in the migration process. Therefore, you must perform the tasks in sequence as listed in the following table:

Tasks	Task description	More information
1	Upgrade HCL OneTest™ Server V10.1.2.	<a href="#">Installation of the server software on Ubuntu</a>
2	Back up the user data from HCL OneTest™ Server V10.1.2.	<a href="#">Backing up user data - V10.1.2</a>
3	Uninstall HCL OneTest™ Server V10.1.2.	<a href="#">Uninstalling the server software</a>
4	Install the latest version of the server software and then migrate the data as part of the installation process.	<a href="#">Installing the server software on Ubuntu by using k3s on page 64</a>

## Backup and restoration of the server data

You must back up all your server data before you uninstall the current version of the server software or upgrade to the latest version of the server software. You must back up the server data to avoid data loss or inaccessibility of data.

A backup is the process of creating a copy of the data on your system and storing it elsewhere, generally on secondary storage. You can then use that copy to recover if your original data is lost or becomes inaccessible.

A restore is a process of copying the backed-up data from the secondary storage and restoring it to the original location. You can restore the backed-up data when your original data is lost or becomes inaccessible.

You must back up and restore the data when you perform the following tasks:

- Move the existing environment to a new system.
- Change the name of the release or namespace that you used during the installation of the server software.
- Minimize the downtime of HCL OneTest™ Server during disaster recovery.
- Upgrade to the latest version of the server software.

To learn more about backup and restoration of the server data on platforms such as Ubuntu, Red Hat OpenShift, and Azure Kubernetes Service (AKS), refer to the following topics:

- [Backup and restoration of the server data on Ubuntu on page 91](#)
- [Backup and restoration of the server data on Red Hat OpenShift on page 89](#)
- [Backup and restoration of the server data on Azure Kubernetes Service on page 96](#)

## Backup and restoration of the server data on Red Hat OpenShift

To secure the data in HCL OneTest™ Server that is installed on the Red Hat OpenShift platform, you can back up the data. At any point in time, after you install HCL OneTest™ Server, you can restore the backed-up data.

### Preparing Red Hat OpenShift cluster to backup and restore the server data

You must prepare your OpenShift cluster before you back up or restore the data of HCL OneTest™ Server.

#### Before you begin

You must have installed OpenShift CLI. For more information, refer to the Getting started with the CLI section in the [Red Hat OpenShift](#) documentation.

#### About this task

Velero is one of the tools that are available to back up and restore the data of HCL OneTest™ Server that is installed on the Red Hat OpenShift platform. You must prepare your cluster before you back up or restore the data by using Velero. You can also use the other tools to back up and restore the data. If you use a different tool, then you must include the Persistent Volumes in the cluster.

1. Log in to the cluster by using `oc login`.
2. Install and configure Velero with the Restic Integration.
3. Set the name of the namespace in which HCL OneTest™ Server is installed by running the following command:

```
NS=<namespace>
```

4. Update stateful sets to apply the annotations required by Velero to back up the PVs used by the pods by running the following commands:

```
for sts in $(oc get sts -n "$NS" -o name); do \
  if ! oc -n "$NS" patch --type=json "$sts" -p \
    '[{"op": "add", "path": "/spec/template/metadata/annotations/backup.velero.io~1backup-volumes",
    "value": "data"}]' 2>/dev/null; then
    oc -n "$NS" patch --type=json "$sts" -p \
```

```
' [{"op": "add", "path": "/spec/template/metadata/annotations",
"value": {"backup.velero.io/backup-volumes": "data"}}] '
fi \
done
```

5. Change the security restriction of the pods by running the following commands:

```
for sts in $(oc get sts -n "$NS" -oname); do \
  if oc get -n "$NS" "$sts" -ojsonpath='{.spec.template.spec.securityContext}' | grep -q
  "runAsNonRoot:true"; then \
    echo $sts
    oc patch -n "$NS" "$sts" --type json \
      -p '[{"op": "replace", "path": "/spec/template/spec/securityContext", "value":
      {"runAsNonRoot": false}}]'
  fi \
done
```

## Results

You have prepared your Red Hat OpenShift cluster to back up the server data.

## What to do next

You can back up the server data. See [Backing up the server data on Red Hat OpenShift on page 90](#).

---

Related information

[Velero Documentation](#)

## Backing up the server data on Red Hat OpenShift

When you want to upgrade to a new version of the server software or to move your existing environments to new systems, you must back up the server data.

### Before you begin

You must have completed the following tasks:

- Been assigned the same role that was required to install and uninstall the server software.
- Installed the server software and all of the HCL OneTest™ Server pods are in `Running` state.
- Communicated to the users that HCL OneTest™ Server might be unavailable for some time until the process is complete.
- Prepared your cluster to back up and restore data by using Velero. See [Preparing Red Hat OpenShift cluster to backup and restore the server data on page 89](#).

1. Log in to the cluster by using `oc login`.
2. Start the backup process by running the following command:

```
velero backup create <backup_file_name> --include-namespaces=<name_of_the_namespace>
```



**Note:** You must replace `<backup_file_name>` and `<name_of_the_namespace>` with the actual value in the command.

## Results

You have backed up the data of HCL OneTest™ Server.

---

Related information

[Velero Documentation](#)

## Restoring the server data on Red Hat OpenShift

You can restore the data that is backed up in HCL OneTest™ Server at any point in time after you install latest version of server software.

### Before you begin

You must have completed the following tasks:

- Been assigned the same role that was required to install and uninstall the server software.
  - Communicated to the users that HCL OneTest™ Server might be unavailable for some time until the process is complete.
  - Backed up the data from HCL OneTest™ Server.
1. Log in to the cluster by using `oc login`.
  2. Start the restore process by entering the following command:

```
velero restore create --from-backup=<backup-name> --restore-volumes
```

## Results

You have restored the data of HCL OneTest™ Server.

---

Related information

[Velero Documentation](#)

## Backup and restoration of the server data on Ubuntu

To secure the data of HCL OneTest™ Server that is installed on the Ubuntu server, you can back up the data. At any point in time, after you install HCL OneTest™ Server, you can restore the backed-up data.

## Backing up the server data on Ubuntu

When you want to upgrade to a new version of the server software or to move your existing environments to new systems, you must back up the server data.

### Before you begin

You must have completed the following tasks:

- Been assigned the same role that was required to install and uninstall the server software.
- Been granted with the `sudo` access.
- Installed the server software and all of the HCL OneTest™ Server pods are in `Running` state.
- Communicated to the users that HCL OneTest™ Server might be unavailable for some time until the process is complete.

1. Log in to the Ubuntu server and open a terminal.
2. Add the software registry to Helm by running the following command:

```
helm repo add hclsoftware https://hclcr.io/chartrepo/ot --username {okta-email-address}
--password {harbor-cli-secret}
```



**Note:** You must replace `{okta-email-address}` with the user name of the Harbor repository and replace `{harbor-cli-secret}` with the secret key that you copied from the Harbor repository.

If the user name contains any special characters, such as `$`, you must enclose it within single quotes.

3. Run the following command to extract the base:

```
helm pull --untar hclsoftware/hcl-onetest-base --version 5.1020.0
```

4. Change to the `hcl-onetest-base` directory.

This directory contains the `backup.sh` script which is required to complete the backup operation.

5. Create a directory that contains metadata related to the Persistent Volume Claims and their Persistent Volumes by running the following command:

```
sudo ./backup.sh create-pvc-links
```

You can use the following optional parameter along with the **create-pvc-links** command:

`--volumes` or `-v`: Use this parameter to specify the directory path of the Volumes.

For example, `backup.sh create-pvc-links -v hcl-onetest-base/my-pvc-links` creates a directory called `my-pvc-links` that stores the metadata required for backup.



**Remember:**





- If you did not provide the volume parameter, by default, the command creates a sub-directory in the same directory where you have the backup script.
- You must ensure that the directory you create must be empty or not exist to avoid script failure.
- If you specify a directory by using the `-v` parameter, then you must use the same parameter value when you restore the data.

6. Run the following command to stop the cluster and HCL OneTest™ Server:

```
k3s-killall.sh
```

7. Run the following command to create a backup of the existing user data:

```
sudo ./backup.sh create [options] <backup-file-name>
```

After you run this command, a backup of the local Persistent Volumes is created. The backup is created as tar archives that is compressed by using gzip (.tar.gz). The **create** command archives the Persistent Volumes into the `<backup-file-name>`.

You can use the following parameters along with the **create** command:

- `--namespace` or `-n`: Use this parameter to back up the Persistent Volumes in the specified namespace. If you do not mention the namespace, then all the Volumes from all the namespaces are included in the backup. The syntax is:

```
--namespace <name of the namespace>
```

- `--volumes` or `-v`: Use this parameter to specify the directory path of the Volumes. The syntax is:

```
--volumes <path-of-the-directory>
```



#### Remember:

- If you did not provide the volume parameter, by default, the command creates a sub-directory in the same directory where you have the backup script.
- If you specify a directory by using the `-v` parameter, then you must use the same parameter value when you restore the data.

For example,

```
sudo ./backup.sh create --namespace test-system my-backup.tar.gz
```

This command creates a backup file named `my-backup.tar.gz` that contains all of the Persistent Volumes associated with pods available in the `test-system` namespace.

8. Run the following command to restart the cluster and HCL OneTest™ Server:

```
sudo systemctl start k3s
```

## Results

You have backed up the data of HCL OneTest™ Server.

## Restoring the server data on Ubuntu

You can restore the data that is backed up in HCL OneTest™ Server at any point in time after you install latest version of server software.

### Before you begin

You must have completed the following tasks:

- Been assigned the same role that was required to install and uninstall the server software.
- Been granted with the *sudo* access.
- Communicated to the users that HCL OneTest™ Server might be unavailable for some time until the process is complete.
- Backed up the data from HCL OneTest™ Server.

### About this task

The following procedure is for the restoration of the server data from the current release. If you are upgrading the server software, then you can restore the server data from the earlier version to the latest version during the installation of the server software. For more information, see the **Related information** section.

1. Log in to the Ubuntu server and open a terminal.
2. Add the software registry to Helm by running the following command:

```
helm repo add hclsoftware https://hclcr.io/chartrepo/ot --username {okta-email-address}
--password {harbor-cli-secret}
```



**Note:** You must replace `{okta-email-address}` with the user name of the Harbor repository and replace `{harbor-cli-secret}` with the secret key that you copied from the Harbor repository.

If the user name contains any special characters, such as \$, you must enclose it within single quotes.

3. Run the following command to extract the base:

```
helm pull --untar hclsoftware/hcl-onetest-base --version 5.1020.0
```

4. Change to the `hcl-onetest-base` directory.

This directory contains the `backup.sh` script which is required to complete the restore operation.

5. Create a directory that contains metadata related to the Persistent Volume Claims and their Persistent Volumes by running the following command if you have not done before:

```
sudo ./backup.sh create-pvc-links
```

You can use the following optional parameter along with the **create-pvc-links** command:

**--volumes** or **-v**: Use this parameter to specify the directory path of the Volumes.

For example, `backup.sh create-pvc-links -v hcl-onetest-base/my-pvc-links` creates a directory called `my-pvc-links` that stores the metadata required to restore your user data.



**Remember:**

- If you did not provide the volume parameter, by default, the command creates a sub-directory in the same directory where you have the backup script.
- You must ensure that the directory you create must be empty or not exist to avoid script failure.

6. Run the following command to stop the cluster and HCL OneTest™ Server:

```
k3s-killall.sh
```

7. Run the following command to restore the backed-up user data:

```
sudo ./backup.sh restore [options] <backup-file-name>
```

The **restore** command overwrites the existing Persistent Volumes with data from the *<backup-file-name>*.

You can use the following parameters along with the restore command:

- **--namespace** or **-n**: Use this parameter to restore a specific namespace from the backup file. If you do not mention the namespace, then volumes from all the namespaces in the backup file are restored.

You can map one namespace to another namespace by using colon (:). The syntax is:

```
--namespace <name of the namespace> [:<target-namespace>]
```

- **--release** or **-r**: Use this parameter if the Helm release name of the server to which the backup is being restored is different than the Helm release name of the server where the backup was taken. The syntax is:

```
--release <backup-release>:<target-release>
```

- **--volumes** or **-v**: Use this parameter to specify the directory path of the Volumes. The syntax is:

```
--volumes <path-of-the-directory>
```

- **-k** or **--confirm**: Use this parameter to skip the confirmation step.

For example,

```
sudo ./backup.sh restore --namespace test-system:new-test-system -r rel-1:rel-2 -k my-backup.tar.gz
```

This command restores the volumes that are backed up from the *rel-1* release in the *test-system* namespace to the *rel-2* release in the *new-test-system* namespace and skips the confirmation step.

8. Run the following command to restart the cluster and HCL OneTest™ Server:

```
sudo systemctl start k3s
```



**Note:**

- The cluster and HCL OneTest™ Server might take some time to restart and until that time, you cannot access the HCL OneTest™ Server URL.
- You can run the `kubectl get pods -A` command to verify that the Kubernetes environment is working. After a while, the status of the pods must be `Running` state.

## Results

You have restored the data of HCL OneTest™ Server.

## Backup and restoration of the server data on Azure Kubernetes Service

To secure the data in HCL OneTest™ Server that is installed on Azure Kubernetes Service (AKS), you can back up the data. At any point in time, after you install HCL OneTest™ Server, you can restore the backed-up data.



**Attention:** The instructions provided are an example of how an AKS cluster can be configured to use Velero for backing up and restoring the HCL OneTest™ Server data. The procedures to backup and restore the server data might be not valid in all the scenarios.

You must refer to the Velero and Azure Kubernetes Service documentation to identify how you can configure Velero in your AKS cluster based on your existing AKS setup to backup and restore the HCL OneTest™ Server data.

- [Velero Documentation](#)
- [AKS Documentation](#)

## Preparing the Azure Kubernetes Service cluster to back up the server data

When you want to back up the server data, you must prepare your Azure Kubernetes Service (AKS) cluster to store the backed-up data.

### Before you begin

You must have completed the following tasks:

- Created an Azure subscription along with an **Owner** or **Contributor** and **User-Access Administrator** role on the Azure subscription.

You can create a subscription from [Microsoft Azure](#) if you do not have a valid subscription.

- Downloaded the required version of the Velero plugin for Microsoft Azure from the [Velero repository](#).
- Copied the Velero executable file into one of the directories in the **PATH** environment variable.
- Installed Azure Command-Line Interface (CLI). For more information, refer to the [Azure CLI](#) documentation.

### About this task

Velero is one of the tools that is available to back up and restore the server data. You must prepare your AKS cluster where you installed HCL OneTest™ Server before you back up the data by using Velero.

1. Run the following command to log in to AKS interactively:

```
az login
```

#### Result

The command-line interface opens a browser and displays the Log-in page of AKS.

2. Sign in with your AKS account credentials.
3. Run the following command to set your active subscription:

```
az account set --subscription <subscription_name>
```

You must replace `<subscription_name>` with the name of your active Azure subscription.

4. Run the following command to set the location of your AKS cluster:

```
az configure --defaults location=<location_name>
```

For example, `az configure --defaults location=eastus2`



**Tip:** You can run the following command to get the list of all the available locations:

```
az account list-locations --query "[].{DisplayName:displayName, Name:name}" -o table
```

5. Run the following commands to create a storage account within the AKS cluster for the Velero to store the backup files:

```
AZURE_BACKUP_SUBSCRIPTION_NAME=<subscription_name>
AZURE_BACKUP_SUBSCRIPTION_ID=$(az account list
  --query="[?name=='$AZURE_BACKUP_SUBSCRIPTION_NAME'].id | [0]" -o tsv)
AZURE_BACKUP_RESOURCE_GROUP=Velero_Backups

az group create -n $AZURE_BACKUP_RESOURCE_GROUP --location <location_name>

AZURE_STORAGE_ACCOUNT_ID="velero$(date -u +%s)"
az storage account create \
  --name $AZURE_STORAGE_ACCOUNT_ID \
  --resource-group $AZURE_BACKUP_RESOURCE_GROUP \
  --sku Standard_GRS \
  --encryption-services blob \
  --https-only true \
  --kind BlobStorage \
  --access-tier Hot
```



**Note:** You must replace `<subscription_name>` and `<location_name>` with the name of your active Azure subscription and location of the AKS cluster.

6. Run the following command to create a container within the storage account to store the backup files:

```
AZURE_BLOB_CONTAINER=velero
az storage container create -n $BLOB_CONTAINER --public-access off --account-name
$AZURE_STORAGE_ACCOUNT_ID
```

7. Run the following commands to obtain a key of Azure storage account and to create a file that contains all the relevant environment variables:

```
AZURE_STORAGE_ACCOUNT_ACCESS_KEY=$(az storage account keys list \
--account-name $AZURE_STORAGE_ACCOUNT_ID \
--query "[?keyName == 'key1'].value" -o tsv)

cat << EOF > ./credentials-velero
AZURE_STORAGE_ACCOUNT_ACCESS_KEY=${AZURE_STORAGE_ACCOUNT_ACCESS_KEY}
AZURE_CLOUD_NAME=AzurePublicCloud
EOF
```

8. Run the following command to install Velero with the restic option by using the Azure storage account key:

```
velero install \
--provider azure \
--plugins velero/velero-plugin-for-microsoft-azure:v1.1.0 \
--bucket $AZURE_BLOB_CONTAINER \
--secret-file ./credentials-velero \
--backup-location-config
resourceGroup=$AZURE_BACKUP_RESOURCE_GROUP,storageAccount=$AZURE_STORAGE_ACCOUNT_ID,storageAccountKeyEnvVar=AZURE_STORAGE_ACCOUNT_ACCESS_KEY,subscriptionId=$AZURE_BACKUP_SUBSCRIPTION_ID \
--use-volume-snapshots=false \
--use-restic
```

## Results

You have prepared your cluster to back up the server data.

## What to do next

You can back up the server data. See [Backing up the server data on Azure Kubernetes Service on page 98](#)

---

### Related information

[Velero Documentation](#)

[AKS Documentation](#)

## Backing up the server data on Azure Kubernetes Service

You might want to back up the server data to protect against data corruption or malicious attacks. You might also want to back up the server data when you want to upgrade to a new version of the server software.

### Before you begin

You must have completed the following tasks:

- Been assigned the same role that was required to install and uninstall the server software.
- Installed the server software and all of the HCL OneTest™ Server pods are in `Running` state.
- Communicated to the users that HCL OneTest™ Server might be unavailable for some time until the process is complete.
- Prepared your cluster to store the backed-up data by using Velero. See [Preparing the Azure Kubernetes Service cluster to back up the server data on page 96](#).

### About this task

If you plan to back up the data immediately after you prepared your Azure Kubernetes Service (AKS) cluster, then you must go to [Step 3 on page 99](#) to start the backup process.

1. Run the following command to log in to AKS interactively:

```
az login
```

### Result

The command-line interface opens a browser and displays the Log-in page of AKS.

2. Sign in with your AKS account credentials.
3. Run the following command to start the backup process of the HCL OneTest™ Server data:

```
velero backup create <backup_file_name> --include-namespaces=test-system
--default-volumes-to-restic
```



**Note:** You must replace `<backup_file_name>` with the name that you want to provide for the backup file.



**Tip:** You can run the `velero backup describe <backup_file_name> --details` command to verify the progress of the backup operation.

### Results

You have backed up the data of HCL OneTest™ Server on AKS.

### What to do next

You can restore the server data. See [Restoring the server data on Azure Kubernetes Service on page 99](#).

---

Related information

[Velero Documentation](#)

## Restoring the server data on Azure Kubernetes Service

You can restore the server data that is backed up in HCL OneTest™ Server when required.

## Before you begin

You must have completed the following tasks:

- Been assigned the same role that was required to install and uninstall the server software.
- Communicated to the users that HCL OneTest™ Server might be unavailable for some time until the process is complete.
- Backed up the data from HCL OneTest™ Server.

## About this task

If you plan to restore the server data immediately after you backed up the data from HCL OneTest™ Server that you installed on Azure Kubernetes Service (AKS), then you must go to [Step 3 on page 100](#) to start the restoration process.

1. Run the following command to log in to AKS interactively:

```
az login
```

### Result

The command-line interface opens a browser and displays the Log-in page of AKS.

2. Sign in with your AKS account credentials.
3. Run the following command to start the restore process of the HCL OneTest™ Server data:

```
velero restore create --from-backup=<backup_file_name> --restore-volumes
```



**Tip:** You can run the `velero restore describe <backup_file_name> --details` command to verify the progress of the restore operation.



**Note:** You must replace `<backup_file_name>` with the name that you provided during the backing up of the server data.

## Results

You have restored the data of HCL OneTest™ Server on AKS.

---

Related information

[Velero Documentation](#)

## Uninstallation of the server software

When you no longer require HCL OneTest™ Server, you can uninstall the server software. You can uninstall the server software depending on the platform on which you installed the server software.



## Uninstalling the server software from Red Hat OpenShift

When you want to install a new version of server software or you want to reinstall the server software when an ongoing installation fails, you can uninstall the server software and its components from the Red Hat OpenShift platform.

### Before you begin

You must have completed the following tasks:

- Installed HCL OneTest™ Server software.
- Closed HCL OneTest™ Server, any open web browsers, and all other applications that are enabled by HCL OneTest™ Server.
- **Optional:** Backed-up data from the previous version of the product.

1. Open and log in to the terminal.
2. Run the following command to stop the workload that is running:

```
oc delete all,cm,secret --lexecution-marker --n test-system
```



**Remember:** `test-system` is the name of the namespace. If you created a namespace by using a different value, then you must use that value in place of the `test-system` in all the instances in this procedure.

3. Run the following command to uninstall the server software:

```
helm uninstall {my-ots} --n test-system
```

The PersistentVolumeClaims and PersistentVolumes that were created during the installation are not deleted automatically. If you reinstall the server software, the user data is reused unless you specifically delete those volumes.



**Note:** You must substitute `{my-ots}` with the same release name that you used during the installation of the server software.

4. Run the following command to delete all user data that is contained in PersistentVolumeClaims and PersistentVolumes:

```
oc delete project test-system
```



**Important:** If you want to migrate data from a previous version to the latest version of the product, then you must ensure to back up the data before you run this command.

### Results

You have uninstalled the server software from the Red Hat OpenShift platform.

## Uninstalling the server software from Ubuntu

When you want to install a new version of server software or you want to reinstall the server software when an ongoing installation fails, you can uninstall the server software and its components from the Ubuntu platform.

### Before you begin

You must have completed the following tasks:

- Installed HCL OneTest™ Server software.
- Closed HCL OneTest™ Server, any open web browsers, and all other applications that are enabled by HCL OneTest™ Server.
- **Optional:** Backed-up data from the previous version of the product.

1. Log in to the Ubuntu server using an SSH session.
2. Run the following command to stop the workload that is running:

```
kubectl delete all,cm,secret --lexecution-marker -n test-system
```



**Remember:** The `test-system` is the name of the namespace. If you created a namespace by using a different value, then you must use that value in place of `test-system` in all the instances in this procedure.

3. Run the following command to uninstall the server software:

```
helm uninstall {my-ots} -n test-system
```

The PersistentVolumeClaims and PersistentVolumes that were created during the installation are not deleted automatically. If you reinstall the server software, the user data is reused unless you specifically delete those volumes.



**Note:** You must substitute `{my-ots}` with the same release name that you used during the installation of the server software.

4. Run the following commands to delete the entire Kubernetes environment, including all user data:

```
#Run the following command if you are on Ubuntu 18.04
cd hcl-onetest-base
sudo ./ubuntu-wipe.sh --confirm

#Run the following command if you are on Ubuntu 20.04
cd hcl-onetest-base
sudo HOME=$HOME ./ubuntu-wipe.sh --confirm
```

After you run this command, the system returns to the same state as if `ubuntu-init.sh` was never run.



**Important:** If you want to migrate data from a previous version to the latest version of the product, then you must ensure to back up the data before you run this command.

## Results

You have uninstalled the server software from the Ubuntu platform.

## Uninstalling the server software from Azure Kubernetes Service

When you want to install a new version of server software or you want to reinstall the server software when an ongoing installation fails, you can uninstall the server software and its components from the Azure Kubernetes Service (AKS) cluster.

### Before you begin

You must have completed the following tasks:

- Installed HCL OneTest™ Server software.
- Closed HCL OneTest™ Server, any open web browsers, and all other applications that are enabled by HCL OneTest™ Server.
- **Optional:** Backed-up data from the previous version of the product.

1. Run the following command to log in to AKS interactively:

```
az login
```

### Result

The command-line interface opens a browser and displays the Log-in page of AKS.

2. Sign in with your AKS account credentials.
3. Run the following command to uninstall the server software:

```
helm uninstall {my-ots} -n test-system
```

The PersistentVolumeClaims and PersistentVolumes that were created during the installation are not deleted automatically. If you reinstall the server software, the user data is reused unless you specifically delete those volumes.



**Note:** You must substitute `{my-ots}` with the same release name that you used during the installation of the server software.

4. Run the following command to delete the resource group from the AKS cluster:

```
az group delete -g <resource_group>
```

You must replace `<resource_group>` with the name of the resource group that you created during the creation of the AKS cluster.

## Results

You have uninstalled the server software from the AKS cluster.

## Configuration of the server software

You can find information about the tasks that you must perform after you installed the server software to configure HCL OneTest™ Server.

The following table describes the tasks that you can perform as an administrator after you installed the server software:

Requirement	Tasks	More information
Required	Review the default user management to know how the server software manages the users and their authentication.	<a href="#">User administration on page 104</a>
Required	Import Certificate Authority (CA) into a browser to prevent certificate errors when you access the HCL OneTest™ Server UI.	Importing Certificate Authority into a browser
Optional	Copy the third-party application Jars to Kubernetes to run API Suites that uses a transport.	<a href="#">Copying third-party application Jars to Kubernetes on page 119</a>
Optional	Change the password seed to provide enhanced security to HCL OneTest™ Server.	<a href="#">Changing the password seed on page 122</a>
Optional	Disable the server extensions that are enabled during the installation of the server software if you do not want the server extensions to run on the cluster to save server resources.	<a href="#">Disabling server extensions on page 125</a>

## User administration

After you install HCL OneTest™ Server, you must consider how the software manages users and authentication.

You can review the default user management provided by the server and decide what additional controls you might want to add. If you manage users and authentication through an existing LDAP/AD server, you can review how to use that server to manage HCL OneTest™ Server users.

## Default user administration

You installed HCL OneTest™ Server and you want to know about how the software manages user access.

HCL OneTest™ Server uses Keycloak V12.0.4 (<https://www.keycloak.org/>) to manage and authenticate users.

If you manage and authenticate users by using an LDAP and Active Directory server, you can configure Keycloak to connect to that server. For more information, see [LDAP user administration on page 106](#).

Keycloak uses the concept of a realm to manage and authenticate users. When you install the server software, a realm called testserver is created for you in Keycloak. All server users belong to this realm and when they log in to the server, they log into that realm.

As an administrator, it is important to consider the following points about the server administration:

- By default, there is no administrator for HCL OneTest™ Server.

Such an administrator is required for accessing additional functions, which includes claiming ownership of server projects and unarchiving them. But you can assign administrative privileges to any user. You must do this by adding the admin role to the user in Keycloak.

- You must sign up a user that you want to be the administrator. You must go to the Login page at `https://<fully-qualified-dns-name>:443` and sign up.



**Note:** Do not use that admin user to perform non-administration tasks. Instead, sign up another user.

- After you sign up the user that you want to be the administrator for HCL OneTest™ Server, you must log in to the Keycloak Admin Console at `https://<fully-qualified-dns-name>:443/auth/admin/` to make that user the server administrator.

The default user name for the Keycloak administrator is `keycloak`. The password is randomly generated when the software is installed. You can see the password by using the following `kubectl` command:

```
kubectl get secret -n test-system {my-ots}-keycloak-postgresql -o jsonpath="{.data.password}" |
base64 --decode; echo
```



**Note:** You must substitute `{my-ots}` with the release name of your choice.

After you log in to the Keycloak Admin Console, from the Users page, you can search and select the user that you want to make an administrator. From the **Groups** tab, you can join the user to the **Admins** group.

For more information about assigning user roles, see [Groups](#) in the Keycloak documentation.

Now that you are the server administrator, it is important to consider the following points about the default user management and authentication:

- Minimum password length defaults to 8 characters
- Email verification of new users is turned off
- The Forgot Password feature is turned on by default but no instructions are sent to the user to reset their password
- Forgotten user passwords are changed by you if you do not enable Keycloak to send instructions to reset a password

You can review the following sections about changing the default authentication controls.

## Email settings

By default, the testserver realm sets the Forgot Password switch on. However, as an administrator, you must enable Keycloak to send an email to the user with instructions to reset their password. If you want to verify an email, you must also enable Keycloak to send an email to the user to verify their email address.

You must provide SMTP server settings for Keycloak to send an email. After you log in to the Keycloak Admin Console, see [Email Settings](#) in the Keycloak documentation.

Then, to set up the email verification, see [Forgot Password](#) in the Keycloak documentation.

## Password policy

By default, the testserver realm has a password policy where the minimum length of a password is 8. As an administrator, you can update password policies in Keycloak.

After you log in to the Keycloak Admin Console, see [Password Policies](#) in the Keycloak documentation.

## User password

If you did not enable Keycloak to send instructions to a user about how to reset a password, you must use the Keycloak Admin Console to change their password for them.

After you log in to the Keycloak Admin Console, see [User Credentials](#) in the Keycloak documentation.

## User deletion

When a user is inactive or no longer needs to access the server, you can delete that user.

After you log in to the Keycloak Admin Console, see [Deleting Users](#) in the Keycloak documentation.

## LDAP user administration

After you installed HCL OneTest™ Server, you can use a Lightweight Directory Access Protocol and Active Directory HTTP server to manage users and authentication. Some of the following best practices, can help you to use an LDAP/AD server to manage user access to HCL OneTest™ Server.

HCL OneTest™ Server uses Keycloak V12.0.4 (<https://www.keycloak.org/>) to manage and authenticate users.

Existing user databases hold user credentials. Keycloak federates these existing external user databases through the concept of storage providers. By default, Keycloak supports an LDAP and Active Directory storage provider. By adding a storage provider, you can map LDAP user attributes into Keycloak. You can also configure more mappings.

Before you configure Keycloak to use an existing LDAP/AD provider, you might consider the following best practices:

- Set up your LDAP/AD provider as a read-only repository so that HCL OneTest™ Server cannot change it.
- Add and remove users in LDAP/AD and not the Keycloak local user database.
- Import and synchronize your LDAP/AD users to your Keycloak local database.
- Map a login style name, for example, `user1@server.com`, by using the `UserPrincipalName` attribute in LDAP/AD to a `username` in Keycloak. If you want the full name of the user as your login style, use the `cn` attribute in LDAP/AD.



**Note:** The LDAP/AD user name attribute must match the LDAP/AD provider user name attribute (**Username LDAP attribute**) in Keycloak for the LDAP/AD provider to connect with Keycloak.

The following sections use these best practices to guide you to set up Keycloak to connect to your LDAP/AD HTTP server.

## LDAP provider selection in Keycloak

You can use the Keycloak Admin Console to add an LDAP/AD provider.

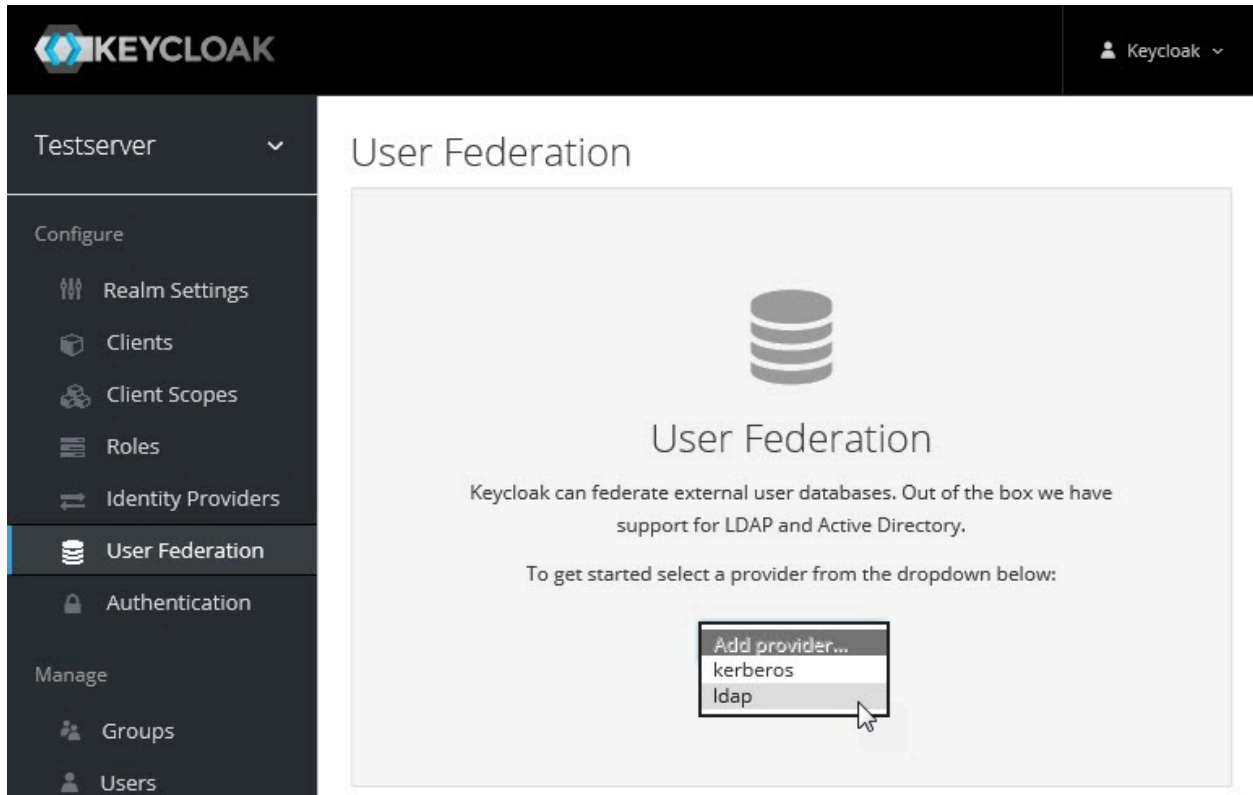
You can log in to the Keycloak Admin Console at `https://<fully-qualified-dns-nam>/auth/admin`. The default user name for the Keycloak administrator is `keycloak`. The password is randomly generated when you installed the HCL OneTest™ Server software. You can see the password by using the following `kubectl` command:

```
kubectl get secret -n test-system {my-ots}-keycloak-postgresql -o jsonpath="{.data.password}" | base64 --decode; echo
```



**Note:** You must substitute `{my-ots}` with the release name of your choice.

After you log in, you can go to the User Federation page to add your provider.



The screenshot shows the Keycloak administration interface. The top navigation bar includes the Keycloak logo and the user 'Keycloak'. The left sidebar is titled 'Testserver' and contains a 'Configure' section with options: Realm Settings, Clients, Client Scopes, Roles, Identity Providers, User Federation (selected), Authentication, and a 'Manage' section with Groups and Users. The main content area is titled 'User Federation' and features a database icon. Below the icon, the text reads: 'User Federation', 'Keycloak can federate external user databases. Out of the box we have support for LDAP and Active Directory.', and 'To get started select a provider from the dropdown below:'. A dropdown menu is open, showing the options 'Add provider...', 'kerberos', and 'ldap'. A mouse cursor is positioned over the 'ldap' option.

You want to select the provider named **ldap** from the list. From the Add user federation provider page, you can use a form to complete your LDAP/AD connection parameters.

### Required settings for a successful connection to your LDAP/AD provider


The form includes many fields and several fields include default values. Tables describe the important fields that you must complete to ensure a successful HTTP connection to your LDAP/AD provider.



## Required Settings

Enabled ?	<input checked="" type="checkbox"/> ON
Console Display Name ?	<input type="text" value="Test Server LDAP/AD Provider"/>
Priority ?	<input type="text" value="0"/>
Import Users ?	<input checked="" type="checkbox"/> ON
Edit Mode ?	<input type="text" value="READ_ONLY"/>
Sync Registrations ?	<input type="checkbox"/> OFF
* Vendor ?	<input type="text" value="Active Directory"/>

Field	Description
Console Display Name	You want to make your console name recognizable, for example, Test Server LDAP/AD Provider. This name is shown in the Keycloak Admin Console.
Priority	The priority must be set to 0 so that when a user logs in to HCL OneTest™ Server, Keycloak looks up the user first in the list of users managed in the LDAP/AD user database. If the user is not found in your LDAP/AD user database, Keycloak then looks up the user in the local Keycloak user database.
Import Users	<p>You want to leave import users <b>ON</b> so that users in your LDAP/AD user database are synchronized (imported) automatically into the Keycloak local user database. The import uses the settings that you are defining now.</p> <p>After the initial import, when you add a user to your LDAP/AD user database and that user logs in to HCL OneTest™ Server, the LDAP/AD provider imports the LDAP/AD user into the Keycloak user database and authenticates the user against the LDAP/AD password.</p> <p>Similarly, after the initial import, when you remove a user from your LDAP/AD user database that user cannot log in to HCL OneTest™ Server.</p>

Field	Description
	 <b>Note:</b> For consistent user management, continue to manage users in your LDAP/AD user database.
Edit Mode	<p>Make sure that edit mode is set to <b>READ_ONLY</b>.</p> <p>This setting means that your LDAP/AD user database is read-only. No user data defined through the mapping of attributes in Keycloak such as the <code>username</code> is written back from the Keycloak user database to your LDAP/AD user database.</p> <p>This read-only setting also means that a new user cannot sign up from HCL OneTest™ Server as no user data can be written to your LDAP/AD user database.</p>
Vendor	You want to select <b>Active Directory</b> from the list of vendors. Many fields complete with default values based on this selection.

\* Username  LDAP attribute ?

\* RDN LDAP attribute  ?

\* UUID LDAP attribute  ?

\* User Object Classes  ?

\* Connection URL  ? Test connection

\* Users DN  ?

Custom User LDAP Filter  ?

Search Scope  ? ▼

\* Bind Type  ? ▼

\* Bind DN  ?

\* Bind Credential  ? Test authentication

Field	Description
Username LDAP attribute	<p>You can use the default value <code>cn</code> for the <code>username</code>, which is a first name, last name, or you can use <code>userPrincipalName</code>, which is <code>username@domain</code>.</p> <p>You might want a login that matches your company style. For example, you might prefer <code>joetester@mycompany.com</code> instead of <code>Joe Tester</code>.</p> <p>If you do make this change the <code>username</code> that you specify now must match the <code>username</code> in the Mapper. Keycloak makes this change for you.</p>
User Object Classes	Because LDAP user records are found based on a user object class, you must set the User Object Classes to <code>User</code> .
Connection URL	<p>So that Keycloak can connect to your LDAP/AD user database, you must enter your LDAP/AD URL, for example, <code>ldap://&lt;hostname&gt;.&lt;domain&gt;</code></p> <p>Make sure that you test the connection and confirm that the connection is successful.</p>
Users DN	You must provide the directory where the LDAP users are listed, for example, <code>cn=Users,dc=OneTest,dc=COM</code> .
Custom User LDAP Filter	<p>You can filter your LDAP/AD users to import a subset of all your LDAP users.</p> <p>For example, you can set up a RTAS user group for your LDAP/AD user database such that only those users are imported to Keycloak.</p> <pre>(memberOf=cn=Testers,dc=OneTest,dc=COM)</pre>
Bind DN Bind Credential	<p>You must also provide the LDAP/AD user database administrator user ID for <b>BIND DN</b> and password for the <b>BIND Credential</b>. These credentials are used by Keycloak to access the LDAP/AD user database.</p> <p>Make sure that you test the authentication and confirm that the authentication is successful.</p>

You can save the settings, which create your LDAP/AD provider.

## Mappers

Keycloak uses mappers to map the user attributes defined in the Keycloak user model such as **username** and **email** to the corresponding user attributes in the LDAP/AD user database. By default, when you saved your settings and created your LDAP/AD provider, the following mappers were created.

## Test Server LDAP/AD Provider

Settings

Mappers

<input type="text" value="Search..."/> <input type="button" value="Q"/> <input type="button" value="Create"/>	
Name	Type
email	user-attribute-ldap-mapper
creation date	user-attribute-ldap-mapper
MSAD account controls	msad-user-account-control-mapper
username	user-attribute-ldap-mapper
last name	user-attribute-ldap-mapper
modify date	user-attribute-ldap-mapper
full name	full-name-ldap-mapper

The username attribute that you specified in the **Username LDAP attribute** must match the **username** attribute defined in the Keycloak mapper for the LDAP/AD user database to connect with Keycloak.

Because you changed the **Username LDAP attribute** from the default value `cn` to `userPrincipalName`, Keycloak made the same change in the mapper called **username** to match.

## Username

ID

Name \* 

Mapper Type 

User Model Attribute 

LDAP Attribute 

Read Only 
 ON
Always Read Value From LDAP 
 OFF
Is Mandatory In LDAP 
 ON
Is Binary Attribute 
 OFF



### User synchronization

You must import all users from your LDAP/AD user database by using the option to **Synchronize all users**. Users are imported based on your saved settings when you set up your LDAP/AD provider.

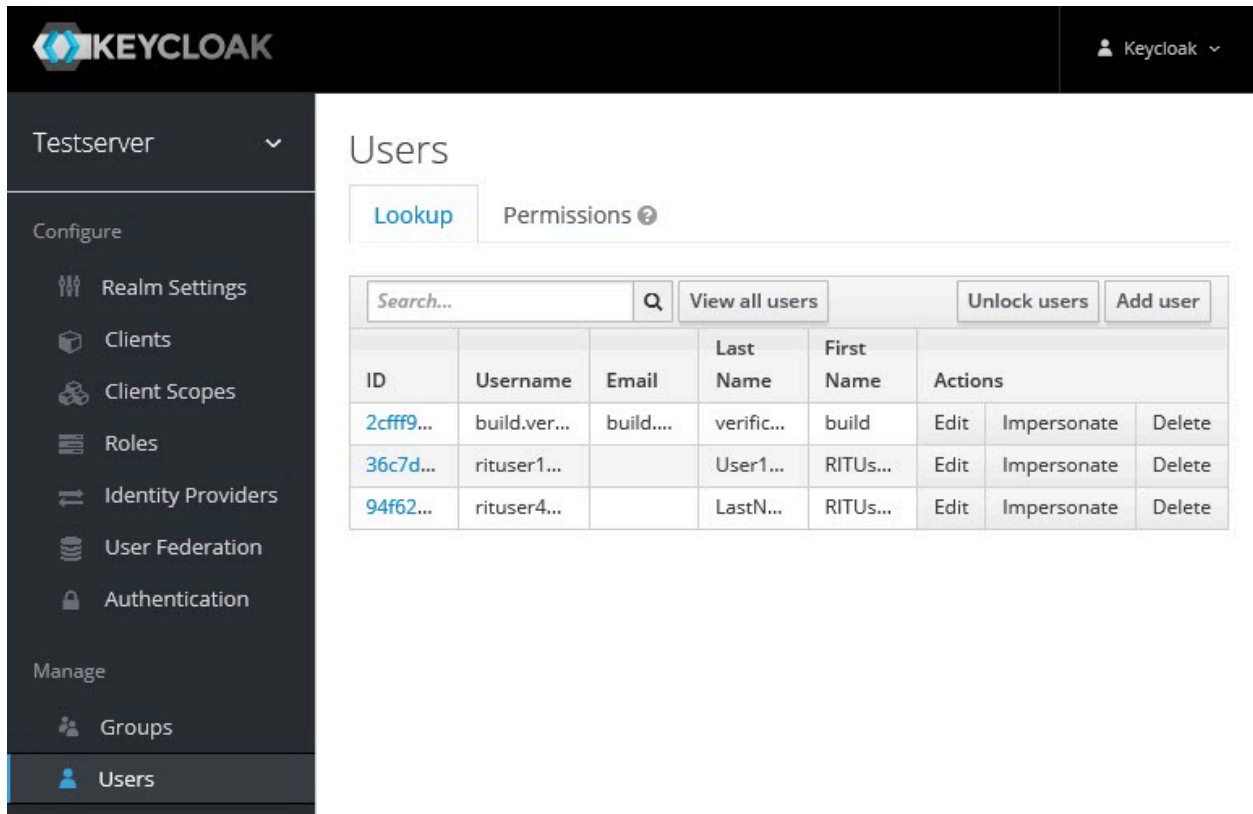






A successful import is followed by a success message with the number of users imported.

You can view all the LDAP/AD database users that were imported and authenticated from the Users page in the Keycloak Admin Console.



The screenshot shows the Keycloak Admin Console interface. The top navigation bar includes the Keycloak logo and a user profile dropdown. The left sidebar is titled 'Testserver' and contains a 'Configure' section with options like 'Realm Settings', 'Clients', 'Client Scopes', 'Roles', 'Identity Providers', 'User Federation', and 'Authentication', and a 'Manage' section with 'Groups' and 'Users'. The 'Users' page is active, showing a 'Lookup' tab and a 'Permissions' icon. Below this is a search bar and a 'View all users' button. A table lists users with columns for ID, Username, Email, Last Name, First Name, and Actions. The table contains three rows of user data.

ID	Username	Email	Last Name	First Name	Actions		
2cfff9...	build.ver...	build....	verific...	build	Edit	Impersonate	Delete
36c7d...	rituser1...		User1...	RITUs...	Edit	Impersonate	Delete
94f62...	rituser4...		LastN...	RITUs...	Edit	Impersonate	Delete

Users are listed with ID, Username, Email, Last Name, and First Name. The ID is generated by Keycloak. The value of the other attributes is fetched from the LDAP/AD user database by using mappers.

**!** **Important:** If you get a message of a failed import for an LDAP/AD user, then the failed import might be due to any one of the following reasons:

- The **User logon name** field under the **Account** tab of that LDAP/AD user is blank.
- The value of the **Custom User LDAP Filter** field to import only a subset of all LDAP users does not match with the LDAP set of users.

You can check the log file of the Keycloak pod to understand the reason for the failed import.

You can perform the following steps to view the log file of the Keycloak pod:

1. Run the following command to view the list of all pods:

```
kubectl get pods -n test-system
```

2. Identify the Keycloak pod `ssocloak-0`.
3. Run the following command to view the log of the Keycloak pod:



```
kubectl logs {my-ots}-ssocloak-0 -n test-system
```



**Note:** Where, {my-ots} is the name of the release that is provided during the installation of the server software.

The log file of the Keycloak pod is displayed.

A sample of a log file is as follows:

```
12:49:00,535 ERROR [org.keycloak.storage.ldap.LDAPStorageProviderFactory] (default task-40)
Failed during import user from LDAP: org.keycloak.models.ModelException: User returned from LDAP
has null username!
Check configuration of your LDAP mappings.
Mapped username LDAP attribute: userPrincipalName, user DN:
CN=DefaultAccount,CN=Users,DC=ONETEST,DC=COM,
attributes from LDAP: {whenChanged=[20190527143130.0Z], whenCreated=[20190527143130.0Z],
cn=[DefaultAccount],
userAccountControl=[66082], pwdLastSet=[0]}
```

## Managing account settings

You might want to change your password or other settings for your account.


### About this task

As an account owner, you can view your account settings, which include your user name, email, and first and last name. You can also change all of these settings except the user name. Changing your password is also possible from your account settings.



**Note:** Account settings cannot be changed if HCL OneTest™ Server was configured to use an LDAP/AD provider. For more information about server security, see the related links.


- Edit your account details by following these steps:

1. Click the **User** icon  from the menu bar, and select **Account Settings**.

The Edit account page is displayed.

2. Edit your user details and save the changes. You can also reset your changes. If you cancel your changes, you return to the Home page.

- Change your password by following these steps:

1. Click the **User** icon  from the menu bar, and select **Account Settings**.

The Edit account page is displayed.

2. Click **Change password**.

The Change password page is displayed.

3. Type your current password followed by your new password
4. Confirm the new password by typing it again and save the changes. You can also reset your changes. If you cancel your changes, you return to the Home page.

---

#### Related information

[Default user administration on page 105](#)

## Certificate authority: Importing and extending lists

When you want browsers and other applications to access HCL OneTest™ Server, you must enable browsers to trust the certificate authority (CA) of HCL OneTest™ Server. For some applications that integrate with HCL OneTest™ Server, you must also extend the CA lists that are trusted by the applications.

You can find the following tasks to import the certificate authority in the browser that you want to use:

- [Importing Certificate Authority into the Google Chrome browser on page 116](#)
- [Importing Certificate Authority into the Microsoft Edge browser on page 117](#)
- [Importing Certificate Authority into the Mozilla Firefox browser on page 118](#)

You can find the following tasks when you want to extend the CA lists:

- [Extending the trusted CA list on Linux on page 118](#)
- [Extending the trusted CA list on Windows systems on page 119](#)

## Importing Certificate Authority into the Google Chrome browser

You must import certificate authority into the Google Chrome browser to prevent certificate errors when accessing the HCL OneTest™ Server UI.

### Before you begin

You must have completed the following tasks:

- Saved a certificate authority (CA) during the installation of HCL OneTest™ Server.

You can run the following command to get the certificate from the system:

```
kubectl get secret ingress -n test-system -o jsonpath={.data.ca\\.crt} | base64 -d
```





**Remember:** *test-system* is the name of the namespace. If you created a namespace by using a different value, then you must use that value in place of the *test-system*.

- Installed the browser that you want to use to access HCL OneTest™ Server.
1. Open a Google Chrome browser.
  2. Navigate to **Customize and control Google Chrome > Settings**.
  3. Click **Security** from the **Privacy and security** pane.
  4. Click **Manage certificates**, and then **Import**.
  5. Click **Next** in the **Certificate Import Wizard** window.
  6. Click **Browse** and select the CA that you want to import, and then click **Next**.
  7. Select the **Place all certificates in the following store** option to store the CA securely.
  8. Click **Browse**, and then select **Trust Root Certification Authorities** as certificate store.
  9. Click **OK**, and then **Finish**.

### Results

You have imported the CA into the Google Chrome browser.

## Importing Certificate Authority into the Microsoft Edge browser

You must import Certificate Authority (CA) into the Microsoft Edge browser to prevent certificate errors when accessing the HCL OneTest™ Server UI.

### Before you begin

You must have completed the following tasks:

- Saved a certificate authority (CA) during the installation of HCL OneTest™ Server.

You can run the following command to get the certificate from the system:

```
kubectl get secret ingress -n test-system -o jsonpath={.data.ca\\.crt} | base64 -d
```



**Remember:** *test-system* is the name of the namespace. If you created a namespace by using a different value, then you must use that value in place of the *test-system*.

- Installed the browser that you want to use to access HCL OneTest™ Server.
1. Type `certmgr.msc` in the **Start** menu **Search** field, and then press **Enter**.
  2. Expand **Trusted Root Certification Authorities**, and then select **Certificate**.
  3. Right-click on the empty space, and then select **All tasks > Import**.
  4. Click **Next** in the **Certificate Import Wizard** window.
  5. Click **Browse** and select the CA that you want to import, and then click **Next**.
  6. Select the **Place all certificates in the following store** option to store the CA securely.

7. Click **Browse**, and then select **Trust Root Certification Authorities** as certificate store.
8. Click **Next**, and then **Finish** to import the certificate.

### Results

You have imported the CA into the Microsoft Edge browser.

## Importing Certificate Authority into the Mozilla Firefox browser

You must import Certificate Authority (CA) into the Mozilla Firefox browser to prevent certificate errors when accessing the HCL OneTest™ Server UI.

### Before you begin

You must have completed the following tasks:

- Saved a certificate authority (CA) during the installation of HCL OneTest™ Server.

You can run the following command to get the certificate from the system:

```
kubectl get secret ingress -n test-system -o jsonpath={.data.ca\\.crt} | base64 -d
```



**Remember:** *test-system* is the name of the namespace. If you created a namespace by using a different value, then you must use that value in place of the *test-system*.

- Installed the browser that you want to use to access HCL OneTest™ Server.
  1. Open a Mozilla Firefox browser.
  2. Enter "about:preferences#privacy" in the address bar, and then press **Enter**.
  3. Locate and click the **View Certificates** option.
  4. Select **Authorities** tab, and then click **Import**.
  5. Select the CA that you want to import, and then click **Open**.
  6. Select **Trust this CA to identify websites** in the **Downloading Certificate** window.
  7. Click **OK** to import the CA into the browser, and then close the window.
  8. Restart Firefox.

### Results

You have imported the CA into the Mozilla Firefox browser.

## Extending the trusted CA list on Linux

HCL OneTest™ Server trusts the certificates signed by a recognized Certified Authority (CA). When you have any additional CAs, you must extend the trusted certificates list by using an environment variable on Linux.

### Before you begin

You must have saved the CA assigned to HCL OneTest™ Server.



**Note:** You can contact the administrator of HCL OneTest™ Server if you do not have a copy of the CA.

1. Open the terminal.
2. Run the following command to create an environment variable on the Linux operating system:

```
export NODE_EXTRA_CA_CERTS= {path of the downloaded CA with the file extension}
```

For example,

```
export NODE_EXTRA_CA_CERTS=/opt/mycert/ca-master.crt
```

### Results

You have extended the trusted CA certificates list on Linux.

## Extending the trusted CA list on Windows systems


HCL OneTest™ Server trusts the certificates signed by a recognized Certified Authority (CA). When you have any additional CAs, you must extend the trusted certificates list by using an environment variable on Windows systems.

### Before you begin

You must have saved the CA assigned to HCL OneTest™ Server.



**Note:** You can contact the administrator of HCL OneTest™ Server if you do not have a copy of the CA.

1. Right-click the Windows icon  from the Windows taskbar, and then click **System**.
2. Click **Advanced system settings** from the **Related setting** section.
3. Click the **Advanced** tab, and then click **Environment Variables**.
4. Click **New** from the **System variables** section.
5. Enter `NODE_EXTRA_CA_CERTS` in the **Variable name** field.
6. Enter the path of the downloaded CA with the file extension in the **Variable value** field.

For example, if you stored the `ca-master.crt` CA in the `C:\Users\Desktop\cert` location, then enter the **Variable value** field as `C:\Users\Desktop\cert\ca-master.crt`

7. Click **OK**.

### Results

You have extended the trusted CA certificates list on Windows systems.

## Copying third-party application Jars to Kubernetes

You can run API Suites in a project on HCL OneTest™ Server. If the API Suite uses a transport and the transport requires third-party application `Jar` files for a successful run, you must ensure that the third-party application `Jar` files are available at the test run time. To achieve this, you must copy the third-party application `Jar` files to the computer where HCL OneTest™ Server is installed on Kubernetes.

**Before you begin**

If you want to copy the third-party application `Jar` files to the remote Docker host, see [Copying third-party application Jars to a remote Docker host on page 294](#).

You must have server administrator or cluster administrator privileges.

You must have completed the following tasks:

- Identified the third-party application `Jar` files that are required and copied the files. See [Test run considerations for API Suites on page 273](#).
- Copied the third-party application `Jar` files from HCL OneTest™ API to the directory or folder in Kubernetes from where you can run the `kubectl` commands.

**About this task**

You can copy the third-party application `Jar` files to the folder that is specific for the application under the `/data/<application_name>` folder. You need not extract the files.

You can perform this task any time after you have installed HCL OneTest™ Server and you plan to run an API Suite that meets any of the following conditions:

- The API Suite uses transports and the transports require third-party application `Jar` files to be available at the test run time.
- The API Suite in the project has a results database configured.

You can copy the JDK version `Jar` files that you have used to create the JUnit tests.

You can get help on the `kubectl` commands by running the command: `kubectl cp --help` from the `/kube` directory. For more information, refer to the [kubectl documentation](#).

1. Use the following table to find the name of the folder that corresponds to the specific third-party application for the transport used in the API Suite.

**Table 2. Name of the folder for the application**

Application	Name of the folder to use
Camel	Camel
CentraSite	CentraSite
CICS	CICS
Coherence	Coherence
Database	JDBC
IMS	IMS

Application	Name of the folder to use
Integra	Integra
JMS	JMS
SAP RFC	SAP
Software AG Universal Messaging	SoftwareAGUM
TIBCO EMS	TIBCO
TIBCO Rendezvous	
TIBCO SmartSockets	
WebSphere Application Server Service Integration Bus (SiBus)	WAS
WebLogic	WebLogicJMX
Software AG webMethods	webMethods
WebSphere MQ	WMQ

- Open the command prompt from the `/kube` directory and copy the `Jar` files to the folder that corresponds to the application by using the following command:

```
kubectl cp <compressed_files> test-system/{my-ots}-userlibs-0:/data/<application_name>/
```

For example, if you are copying the Database `Jar` files, then the name of the folder is `JDBC` and the command to use is:

```
kubectl cp mysql-connector-java.jar test-system/{my-ots}-userlibs-0:/data/JDBC/
```



**Note:** You must substitute `{my-ots}` with the release name of your choice.

- Verify whether the `Jar` files are copied by running the following command:

```
kubectl exec {my-ots}-userlibs-0 -n test-system -- ls /data/<folder_created>
```

For example, if you copied the Database `Jar` files, `mysql-connector-java.jar` to the folder `JDBC`, then the command to verify is:

```
kubectl exec {my-ots}-userlibs-0 -n test-system -- ls /data/JDBC
```

The following information is displayed for the `JDBC` folder:

```
mysql-connector-java.jar
```



**Note:** You must substitute `{my-ots}` with the release name of your choice.

## Results

You have successfully copied the third-party application `JAR` files to the folder in Kubernetes.

## What to do next

You can configure the API Suite run. See [Configuring an API Suite run on page 314](#).

## Changing the password seed

As an administrator, you can change the password seed that is used when you install the server software to provide enhanced security to HCL OneTest™ Server.

### Before you begin

You must have completed the following tasks:

- Installed HCL OneTest™ Server. See [Installation of the server software on page 36](#).
- Installed the JSON command-line tool, `jq`, and ensured that the `jq` is in your environment `PATH`. For information about `jq`, refer to [jq documentation](#).
- Installed the Curl command line tool. For more information refer to [curl documentation](#).

### About this task

When you install HCL OneTest™ Server, you supply a password seed when you run the `helm install` command. This password seed is used to generate several Kubernetes secrets. Kubernetes Secrets can contain the following information:

- The authentication credentials for micro-services.
- An encryption key for the user-created secrets collection or other secrets.

When you change the password seed for HCL OneTest™ Server, you must consider the following scenarios:

- HCL OneTest™ Server cannot communicate until you reconcile the passwords which are in Kubernetes Secrets by using the old and a new password seed.
- Users cannot read secret collections or other secrets that they have created in HCL OneTest™ Server until you re-encrypt them using a new password seed.



**Important:** You must provide an offline token and old password seed that you used during the installation of server software to re-encrypt user secrets.

1. Run the following command to change the password seed for HCL OneTest™ Server:

```
helm upgrade {my-ots} ./hcl-onetest-server -n test-system \
--reuse-values \
--set global.hclOneTestPasswordAutoGenSeed= {my-new-super-secret}
```



**Notes:** You must substitute the value of the following variables with the actual value in the command:

- `{my-ots}` with the release name that you used during the installation of the server software.
- `{my-new-super-secret}` with a new value of your choice as the password seed.
- You must run the following `helm upgrade` command from the same directory where the `helm install` command was run during the installation of the server software. Because the upgrade is dependent on the helm charts and `.yaml` file values used during the run time of the `helm install` command.

2. Run the following script to generate new server secrets from the updated password seed and to save them to the persistent storage:

```
./hcl-onetest-server/files/reconcile-secrets.sh -n test-system {my-ots}
```

3. Run the following command to restart all the pods:

```
kubectl delete pods -n test-system \
-lapp.kubernetes.io/instance={my-ots} \
-lapp.kubernetes.io/managed-by=Helm
```

4. Run the following commands to re-encrypt the user-created secrets collection or other secrets by providing the old password seed:

```
export ACCESS_TOKEN=$(curl -k -X POST {SERVER_URL}/rest/tokens/ \
-H "Content-Type: application/x-www-form-urlencoded" \
-H "accept: application/json" \
-d "refresh_token={OFFLINE_TOKEN}" | jq -r '.access_token')

curl -k -X POST {SERVER_URL}/rest/secrets/re-encrypt/ \
-H "Authorization: Bearer $ACCESS_TOKEN" \
-H "Content-Type: application/json" \
-d "{\"type\":\"helm\",\"password_auto_gen_seed\":\"{OLD_SEED}\"}"
```



**Note:** You must substitute the value of the following variables with the actual value in the following commands:

- `{SERVER_URL}` with the URL of your HCL OneTest™ Server UI.
- `{OFFLINE_TOKEN}` with the offline token that belongs to a user with the administrator role.
- `{OLD_SEED}` with the previous password seed that you used during the installation of the server software.

5. Run the following command to display the log file of the gateway pod:

```
kubectl logs {my-ots}-gateway-abcdefghij-abcde -n test-system
```



**Note:** You must substitute the value of the following variables with the actual value in the command:

- `{my-ots}` with the release name that you used during the installation of the server software.
- `abcdefghij-abcde` with an identifier that is assigned to the name of the gateway pod.

You can run the `kubectl get pods -n test-system` command to obtain the identifier that is assigned to the gateway pod.

## Result

The following message is displayed when re-encryption is completed:

```
reEncrypt complete. StringyReEncryptor [total=100, fixed=100, broken=0, noop=0]
```

## Results

You have successfully changed the password seed for HCL OneTest™ Server.

## Server extensions

You can run Postman, JMeter, or JUnit tests in HCL OneTest™ Server by enabling the server extensions, if they were disabled. The server extensions are enabled at the time of installation of the server software.

When the server extensions are enabled, you can perform the following tasks on HCL OneTest™ Server:

- Add repositories that contain the test assets for the Postman collections, JMeter tests, or JUnit tests.
- You can configure a test run for the following test types:
  - **Postman**
  - **JMeter**
  - **JUnit**



### Related information

[Configuring a JMeter test run on page 355](#)

[Configuring a Postman test run on page 364](#)

[Configuring a JUnit test run on page 359](#)

## Disabling server extensions

When you do not want the server extensions to run on the cluster to save server resources or when you no longer want to run any of the tests supported by the server extensions, you can disable them.

### Before you begin

You must have ensured that the server extension that you want to disable is enabled and running.



**Note:** You can verify the status by running the `kubectl get pods -n test-system` command. The status of an enabled server extension is displayed as `Running`.

### About this task

You can disable server extensions depending on the platform on which you have installed the server software:

- [Disabling server extensions on Ubuntu on page 125](#)
- [Disabling server extensions on OpenShift on page 126](#)

## Disabling server extensions on Ubuntu

1. Log in to the Ubuntu server using an SSH session.
2. Run the following command to disable all the three extensions:

```
helm upgrade {my-ots} -n test-system ./hcl-onetest-server \
--reuse-values \
--set global.hclOneTestPostmanEnabled=false \
--set global.hclOneTestJMeterEnabled=false \
--set global.hclOneTestJUnitEnabled=false
```



**Note:** You must substitute `{my-ots}` with the release name that you provided during the installation of the server software.

If you want to disable any specific extension, then you can set the respective helm parameter value to `false`. For example, to disable the JMeter extension alone, then you can run the following command:

```
helm upgrade {my-ots} -n test-system ./hcl-onetest-server \
--reuse-values \
--set global.hclOneTestJMeterEnabled=false
```

3. Run the following command to verify the status of the extensions:

```
kubectl get pods -n test-system
```



**Note:** The pod for each server extension that you disabled no longer displays.

## Disabling server extensions on OpenShift

1. Open and log in to the terminal.
2. Run the following command to disable all the three extensions:

```
helm upgrade {my-ots} -n test-system ./hcl-onetest-server \
--reuse-values \
--set global.hclOneTestPostmanEnabled=false \
--set global.hclOneTestJMeterEnabled=false \
--set global.hclOneTestJUnitEnabled=false
```



**Note:** You must substitute {my-ots} with the release name that you provided during the installation of the server software.

If you want to disable any specific extension, then you can set the respective helm parameter value to *false*. For example, to disable the JMeter extension alone, then you can run the following command:

```
helm upgrade {my-ots} -n test-system ./hcl-onetest-server \
--reuse-values \
--set global.hclOneTestJMeterEnabled=false
```

3. Run the following command to verify the status of the extensions:

```
oc get pods -n test-system
```



**Note:** The pod for each server extension that you disabled no longer displays.

### Results

You have disabled the required server extensions on HCL OneTest™ Server.

## Enabling server extensions

If you find that the server extension is disabled and you want to either add the test assets to the repository or run the tests supported by the server extension, you must enable the server extension.

### About this task

You can enable server extensions depending on the platform on which you have installed the server software:

- [Enabling server extensions on Ubuntu on page 127](#)
- [Enabling server extensions on OpenShift on page 127](#)

## Enabling server extensions on Ubuntu

1. Log in to the Ubuntu server using an SSH session.
2. Run the following command to enable all the three extensions:

```
helm upgrade {my-ots} -n test-system ./hcl-onetest-server \
--reuse-values \
--set global.hclOneTestPostmanEnabled=true \
--set global.hclOneTestJMeterEnabled=true \
--set global.hclOneTestJUnitEnabled=true
```



**Note:** You must substitute {my-ots} with the release name that you provided during the installation of the server software.

If you want to enable any specific extension, then you can set the respective helm parameter value to *true*. For example, to enable the Postman extension alone, then you can run the following command:

```
helm upgrade {my-ots} -n test-system ./hcl-onetest-server \
--reuse-values \
--set global.hclOneTestPostmanEnabled=true
```

3. Run the following command to verify that all the three or specific extensions are enabled:

```
kubectl get pods -n test-system
```

### Result

The status of the specific extension or all the three extensions in the pod is displayed as *Running*.

## Enabling server extensions on OpenShift

1. Open and log in to the terminal.
2. Run the following command to enable all the three extensions:

```
helm upgrade {my-ots} -n test-system ./hcl-onetest-server \
--reuse-values \
--set global.hclOneTestPostmanEnabled=true \
--set global.hclOneTestJMeterEnabled=true \
--set global.hclOneTestJUnitEnabled=true
```



**Note:** You must substitute {my-ots} with the release name that you provided during the installation of the server software.

If you want to enable any specific extension, then you can set the respective helm parameter value to *true*. For example, to enable the Postman extension alone, then you can run the following command:

```
helm upgrade {my-ots} -n test-system ./hcl-onetest-server \
--reuse-values \
--set global.hclOneTestPostmanEnabled=true
```

3. Run the following command to verify that all the three or specific extensions are enabled:

```
oc get pods -n test-system
```

### Result

The status of the specific extension or all the three extensions in the pod is displayed as `Running`.

### Results

You have enabled the required server extensions on HCL OneTest™ Server.

---

#### Related information

[Configuring a JMeter test run on page 355](#)

[Configuring a Postman test run on page 364](#)

[Configuring a JUnit test run on page 359](#)

## Team space administration

You can find information about team spaces including the initial team space on HCL OneTest™ Server along with the tasks that you can perform to administer a team space.

You can find the following information about team spaces:

- [Team space overview on page 128](#)
- [Creating a team space on page 131](#)
- [Tasks in a team space on page 132](#)
- [Tasks in projects in a team space on page 136](#)
- [License management on page 138](#)
- [Configuring email notifications](#)

### Team space overview

Team spaces are logical partitions of HCL OneTest™ Server. Team space is an exclusive, separate, and secure space on HCL OneTest™ Server that enables you to share a single installation among multiple teams.

You can find the following information about team spaces:

- [Initial team space on page 128](#)
- [Team spaces on page 129](#)
- [Migration from earlier releases of HCL OneTest Server on page 130](#)

### Initial team space

After you install HCL OneTest™ Server and log in for the first time, you enter into the **Initial Team Space**.

The initial team space is the default logical partition that is created on an instance of HCL OneTest™ Server.

As a server administrator, you must consider the following points:

- After you install HCL OneTest™ Server and log in for the first time, the **Projects** page of the initial team space is displayed.
- You must configure the license server for the initial team space.
- Add users in the team space and assign them different roles in a team space.
- You can assign the role of a *Team Space Owner* to any other member in the initial team space.
- Change the type of the initial team space from the default type of *permissive*.

The following table describes the type that you can assign to a team space:

Type	Description
Permissive	Specifies that any licensed user of HCL OneTest™ Server can access a team space and get assigned the default roles of a <i>Project Owner</i> and <i>Member</i> in the team space.
Restrictive	Specifies that all licensed users of HCL OneTest™ Server must request for access to a team space and must be provided access and assigned roles by the <i>Team Space Owner</i> .  The <b>Restrictive</b> type of team space can be either <b>Public</b> or <b>Private</b> . A <b>Public</b> team space is visible in the team spaces dashboard, while a <b>Private</b> team space is not visible in the team spaces dashboard.

- You can evaluate the requirement and create team spaces other than the initial team space.

As a licensed user, you must consider the following points:

- When you log in for the first time immediately after the installation of HCL OneTest™ Server, the **Projects** page of the initial team space is displayed.
- After you log in to the initial team space, you are assigned the default roles of *Project Creator* and *Member* of the initial team space.
- On subsequent log ins, the **My Projects** page in the initial team space is displayed as the default page.

## Team spaces

You can either work with the initial team space or you can create multiple team spaces on HCL OneTest™ Server. The team space facilitates users of the same team space to work together.

Apart from being a container for projects, a team space facilitates the registration of remote Docker hosts and Resource Monitoring agents, designing the system model, and viewing the registered agents, Dockers, or intercepts.

After you create a team space as a **server administrator**, you must consider the following actions:



**Note:** You are assigned the default role of a *Team Space Owner* for team spaces that you create.

- Add users in the team space and assign them different roles of a team space.
- Assign the role of a *Team Space Owner* to other team space members, if you want other members to own the team space.



**Note:** As a *Team Space Owner*, you can remove any member of your team space from the assigned role. The member who is removed from a team space, can send a request and become a member of that team space after the request is accepted.

- If the *Team Space Owner* is no longer a member of a team space, then you as the server administrator can access that team space with the default role of a *Team Space Owner*. You can then assign the role of a *Team Space Owner* to any other member.
- Delete the initial team space or any other team space that you created, if required.



**Note:** If you delete the initial team space, the users of the initial team space are redirected to the **Team Space Dashboard** page.

You as a server administrator or the *Team Space Owner* must perform the following tasks in a team space before other members can start using the team space:

- Configure the license server.
- Configure the team space type as either *Restrictive* or *Permissive*.
- Configure the SMTP server and add members who can receive notifications about the operations performed in the team space.

As a member of a team space, you must consider the following points:

- When you log in for the first time immediately after the installation of HCL OneTest™ Server, the **Projects** page of the team space is displayed.
- After you log in to the team space, you are assigned the default roles of a *Project Creator*, and *Member* of the team space.
- You can request to become a member of any other team space. After your request is approved, you are assigned the default role of a *Member* of that team space.
- You can become a member of either a **Permissive** or **Restrictive** type of team space.
- On subsequent log ins, the **Team Space Dashboard** page is displayed as the default page.

## Migration from earlier releases of HCL OneTest™ Server

The initial team space is the only logical partition in HCL OneTest™ Server V10.1.3. If you migrate from an earlier release of HCL OneTest™ Server to V10.2, then all the assets and resources that you created and associated with the initial team space are migrated to associate with the initial team space in HCL OneTest™ Server V10.2.

The following are the results of migration:

- All projects are contained in the initial team space.
- If you created a repository in a team space, then you can find the repository in the initial team space in V10.2.
- All Dockers, agents, and intercepts that you registered with the server are then registered with the initial team space and you can view from the **Agents and Intercepts** page of the initial team space.
- If you configured a resource monitoring agent in any project, then the details of the configured agent are available in the project.
- You can find the details of SMTP configuration on the server. Only you as a server administrator can view the details of SMTP configuration.
- The license server that is configured for the server for earlier release of HCL OneTest™ Server is then configured for the initial team space.
- All users are assigned the role of a *Project Creator* and *Architect* of the initial team space.
- All users with server administrator roles are assigned the default role of a *Team Space Owner* of the initial team space.



**Note:** All users of earlier releases are migrated to HCL OneTest™ Server V10.2. You can view the list of all migrated users in the application only when they log into the application at least once.

---

Related information

[Team space administration on page 128](#)

## Creating a team space

To manage projects in the logical containers of HCL OneTest™ Server, you must create team spaces.

### Before you begin

You must have been assigned the role of a server administrator of HCL OneTest™ Server.

### About this task

As a server administrator, you can create multiple team spaces on HCL OneTest™ Server.

1. Log in to HCL OneTest™ Server.

#### Result

The **Projects** page of the initial team space is displayed.

2. Click **Dashboard**.

#### Result

The **Team Space Dashboard** page is displayed.

3. Click **New team space**.

#### Result

The **New Team Space** dialog box is displayed.

4. Provide the name to the team space.
5. Enter a name as a URL alias for the team space.
6. Select an option to set the permissions to access the team space. You can select any one of the following options:
  - **Permissive** – When a team space is set as **Permissive**, the user gets the permissions automatically on sending the request to become a member of that team space.
  - **Restrictive** – When a team space is set as **Restrictive**, the *Team Space Owner* provides permission to the user to become a member of that team space.

You can set the visibility mode of the **Restrictive** type team space as **Public** or **Private**.

- The **Public** team space is visible to all users in the **Team Space Dashboard**.
- The **Private** team space is visible only to the members of that team space in the **Team Space Dashboard**.



**Note:** If you select the **Restrictive** type team space, then the default setting of the team space becomes **Public**.

7. Provide a brief description of the team space.
8. Select the owner of the team space from the list of users who accessed the application.

The selected user is assigned the role of a *Team Space Owner*.



**Note:** You can assign the *Team Space Owner* role to only one user while you create a team space.

9. Click **Create**.

#### **Result**

The dashboard of the team space is displayed.

### **Results**

You have created a team space.

If you are assigned the role of a *Team Space Owner*, then you can view the name of the team space that you created in the navigation pane.

If you assigned any other licensed user with the role of a *Team Space Owner*, then you can view the team space under the **Other Team Spaces** panel.

### **What to do next**

You must configure the license for your team space. See [Configuring licenses for team spaces on page 140](#).

## Tasks in a team space

You can find information about the tasks that you can perform as a *Member*, *Project Owner*, *Architect*, or *Team Space Owner* in a team space on HCL OneTest™ Server.



The roles assigned in a team space for users are as follows:

Persona	Initial team space		Any team space	
	Role	Notes	Role	Notes
Server administrator	<i>Team Space Owner</i>	Default role assigned	<i>Team Space Owner</i>	Default role assigned
Licensed user	<i>Team Space Owner</i>	Role assigned by Server administrator	<i>Team Space Owner</i>	Role assigned by Server administrator
	<i>Architect</i>	Role assigned by <i>Team Space Owner</i>	<i>Architect</i>	Role assigned by <i>Team Space Owner</i>
	<i>Project Creator</i>	Default role assigned	<i>Project Creator</i>	<ul style="list-style-type: none"> <li>• Default role assigned in a <i>Permissive</i> team space.</li> <li>• Role assigned by <i>Team Space Owner</i> in a <i>Restrictive</i> team space.</li> </ul>
	<i>Member</i>	Default role assigned	<i>Member</i>	Default role assigned in either a <i>Permissive</i> or <i>Restrictive</i> team space.


The following table shows the tasks that can be performed by licensed users with their assigned roles in a team space:

Tasks	Team Space Owner	Architect	Project Creator	Member
Configuring licenses	✓			
Configuring a Simple Mail Transfer Protocol (SMTP) server to send email notifications.	✓			
Changing the license server configuration	✓			
Adding members in a team space	✓			
Assigning different roles to other team space members	✓			
Modifying configuration details of a team space	✓			

Tasks	Team Space Owner	Architect	Project Creator	Member
Deleting a team space	✓			
Renaming a team space	✓			
Adding or removing the team space repository from a team space	✓	✓		
Designing a system model	✓	✓		
Creating projects in a team space	✓		✓	
Adding, removing, or modifying remote Docker hosts	✓	✓	✓	✓
Adding agents	✓	✓	✓	✓
Adding, modifying, or removing resource monitoring sources	✓			
Connecting resource monitoring agents	✓			
Adding resource monitoring labels	✓			
Working with projects in a team space	✓	✓	✓	✓

The following table provides the links to the information about the tasks that can be performed by users with different roles in a team space:

Tasks	Roles	More information
Configuring a team space license server	<i>Team Space Owner</i>	<a href="#">Configuring licenses for team spaces on page 140</a>
Configuring the Named Users	<i>Team Space Owner</i>	<a href="#">Adding users to the Named Users list on page 143</a>
Modifying the configuration of a team space	<i>Team Space Owner</i>	<a href="#">Modifying the configuration of a team space on page 571</a>
Viewing the configuration of a team space	All members of a team space	<a href="#">Viewing the configuration of a team space on page 573</a>
Reassigning the role of team space members	<i>Team Space Owner</i>	<a href="#">Managing members and their roles in a team space on page 576</a>

Tasks	Roles	More information
Becoming a team space member	<ul style="list-style-type: none"> <li>• <i>Team Space Owner</i></li> <li>• All users can send a request</li> </ul>	<a href="#">Becoming a team space member on page 575</a>
Viewing team spaces	<p>All members of a team space</p> <p> <b>Notes:</b></p> <ul style="list-style-type: none"> <li>• The <b>Permissive</b> and public <b>Restrictive</b> team spaces are visible to all the users.</li> <li>• The private <b>Restrictive</b> team spaces are only visible to its members.</li> </ul>	<a href="#">Viewing team spaces on page 567</a>
Deleting a team space	<i>Team Space Owner</i>	<a href="#">Deleting a team space on page 579</a>
Adding a project in a team space	<i>Project Creator</i>	<a href="#">Adding a project on page 582</a>
Viewing existing projects	All members of a team space	<a href="#">Viewing projects on page 586</a>
Viewing details about the users or license server	All members of a team space	<a href="#">Viewing license details of the team space on page 145</a>
Installing remote agents and configuring them to register with HCL OneTest™ Server	All members of a team space	<a href="#">Management of agents on page 282</a>
Registering a remote Docker host with HCL OneTest™ Server	All members of a team space	<a href="#">Registering a remote Docker host on page 296</a>
Editing configurations of a remote Docker host	All members of a team space	<a href="#">Editing configurations of a remote Docker host on page 300</a>
Unregistering a remote Docker host	All members of a team space	<a href="#">Unregistering a remote Docker host from HCL OneTest Server on page 304</a>
Viewing remote agents, remote Docker hosts, or intercepts that are registered with HCL OneTest™ Server in the initial team space	All members of a team space	<ul style="list-style-type: none"> <li>• <a href="#">Viewing remote Docker hosts that are registered with HCL OneTest Server on page 298</a></li> </ul>

Tasks	Roles	More information
		<ul style="list-style-type: none"> <li>• Viewing agents that are registered with HCL OneTest Server on page 283</li> <li>• Viewing intercepts that are registered with a team space on HCL OneTest Server on page 397</li> </ul>
Adding resource monitoring sources and agents	<i>Team Space Owner</i>	<a href="#">Resource monitoring service on page 446</a>
Adding resource monitoring labels to the sources	<i>Team Space Owner</i>	<a href="#">Controlling resource monitoring sources in a schedule on page 474</a>
Viewing the performance metrics	All members of a team space	Viewing the performance metrics
Creating a model of the system under test	<i>Architect</i>	System modeling
Adding a repository to a team space	<ul style="list-style-type: none"> <li>• <i>Team Space Owner</i></li> <li>• <i>Architect</i></li> </ul>	Adding a repository to a team space
Viewing an existing system model	All members of a team space	Viewing the system model

---

#### Related information

[Team space administration on page 128](#)

## Tasks in projects in a team space


You can find information about the roles that you can assign in a project for members of a team space and the tasks that you can perform in projects in a team space.

Members of a team space with the role of a *Project Creator* can create projects in the team space and become the *Owner* of a project. The owners of projects can add *Members* of the team space to a project and assign them the following roles:

- *Owner*
- *Tester*
- *Viewer*

For the list of actions that members assigned any of the roles can perform, see [Managing access to server projects on page 588](#).

All projects are created in a team space. When you open a project, the options that are displayed in the navigation pane are specific to the projects within a team space.

When you are working on a project in a team space and you want to return to the options available to work in the team space, you can click the **Home** icon  in the navigation pane.



**Note:** You must be assigned the role of an *Owner*, a *Tester*, or a *Viewer* of a project.

The following table lists the tasks that members of a team space can perform while working with projects in a team space:

Tasks	More information...
Viewing projects in a team space.	<a href="#">Viewing projects on page 586</a>
Adding a project.	<a href="#">Adding a project on page 582</a>
Adding a repository that contains test assets and resources that you want to run on the server.	<a href="#">Adding repositories to a server project on page 583</a>
Adding secrets to a secrets collection.	<a href="#">Secrets configuration on page 585</a>
Encrypting the data set resources.	Data security
Configuring a change management system.	Change management system
Adding members to a project and configuring their roles.	<a href="#">Adding users to a project on page 585</a>
Selecting the global branch in a project to view the test assets and resources.	<a href="#">Selecting the global branch in a project on page 597</a>
Adding agents to your project, if you are using remote agents as a location to run certain tests.	Adding an agent to a project
Adding remote Dockers, if you are using a remote Docker host as a location to run tests.	<a href="#">Adding a remote Docker host to the project for running tests on page 299</a>
Reading the considerations that you must take into account before you configure a test run.	<a href="#">Prerequisites to running tests on page 271</a>
Reading the considerations that you must take into account before you configure a run or start a virtual service.	<a href="#">Prerequisites for running virtual services on page 392</a>
Configuring a run of the test assets.	<a href="#">Test run configurations on page 305</a>
Configuring a run of the virtual service resource.	<a href="#">Configuring a run of a virtual service on page 401</a>
Viewing the progress of a test run.	<a href="#">Viewing the progress of running test assets on page 381</a>

Tasks	More information...
Managing a running test.	<a href="#">Management of running tests on page 378</a>
Viewing the results of a test run after the run is completed.	<a href="#">Test results on page 435</a>
Reviewing the test results and creating defects for test runs from the <b>Results</b> page.	<a href="#">Creating Jira defects on page 484</a>

## HCL license portal

Licensing for HCL OneTest™ Server is administered through [HCL Software License & Download portal](#). This portal is a FlexNet-based web application that helps to manage software entitlements and licenses.

When a software order is placed and acknowledged, a software entitlement is created. You must then follow the instructions in the Software Order Acknowledgment document that you receive to activate your entitlement, create devices, and download the software from the portal.

The license portal provides both software distribution and management of your software entitlements that are purchased from HCL Software. The portal provides control and flexibility on how to consume your licenses. An organization identifies one of its resources as a License Manager (also called Tech or Portal Admin) who is familiar with the language of licenses.

For more information about the [HCL Software License & Download portal](#), you can refer to the following knowledge articles:

- [What is the HCL Software License & Download portal \(FlexNet portal\)?](#)
- [How to find HCL Product Releases in HCL Software License & Download portal](#)
- [Managing Users on the HCL Software License & Download portal](#)

## License management

As a *Team Space Owner*, you can configure and manage licenses for your team spaces in HCL OneTest™ Server. As a member of any team space, you can only view the details of the licenses that are configured for the team space. If you are not a member of any team space, you cannot view or perform any operations in any of the team spaces on HCL OneTest™ Server.

When you log in to HCL OneTest™ Server, you can configure the details of the license by clicking **Manage > Licenses** for the initial team space or any other team space that you create.

Depending on your privileges, you can perform the following tasks in HCL OneTest™ Server:

Privileges	Tasks	More information
<i>Team Space Owner</i>	Read about licensing on HCL OneTest™ Server.	<a href="#">Server software licensing information on page 139</a>
	Configure licenses.	<a href="#">Configuring licenses for team spaces on page 140</a>
	Configure named users.	<a href="#">Adding users to the Named Users list on page 143</a>
	Remove or replace named users.	<a href="#">Removing or replacing named users on page 143</a>
Licensed user	Perform tasks or operations in a team space.	<a href="#">Tasks in a team space on page 132</a>
	Viewing details about licenses.	<a href="#">Viewing license details of the team space on page 145</a>

## Server software licensing information

You can find information about the licensing on HCL OneTest™ Server.

To use the features offered by HCL OneTest™ Server, you must purchase the following licenses:

- Named user
- Concurrent Virtual Service

### Named user licenses

As a *Team Space Owner*, you can configure the License Server and add users to the **Named Users** list for the team spaces.

The **Named user** licenses are automatically self-acquired on a first-come-first-serve basis. The licenses are always available for the users who are added to the **Named Users** list. Also, those users can perform the project tasks in their team space even though the **Disable automatic self-acquisition of named user licenses** checkbox is cleared in the **Team Space License Configuration** page. However, the other users receive an error notification about the unavailability of the license when they try to perform the project tasks.



**Remember:** A single **Named user** license is used from the total number of **Named user** licenses when you perform most of the operations in HCL OneTest™ Server.



**Important:** You can perform only the following operations in team space without using a license:



- Configure the License Server
- Modify the user roles in a team space
- Delete the team space



**Remember:** The **Named user** license is returned to the license pool only under any of the following conditions:

- When the **Named user** is no longer a member of any team space.
- When the **Named user** is removed as a user of HCL OneTest™ Server.



**Important:** The **Named user** license can take a few minutes to return to the license pool.

## Concurrent Virtual Service licenses

HCL OneTest™ Server requires an additional license to use the service virtualization feature. You must have sufficient **Concurrent Virtual Service** licenses to run instances of virtual services that are contained in API Suites on HCL OneTest™ Server.

A single **Concurrent Virtual Service** is used from the total number of **Concurrent Virtual Service** licenses when any of the following events occur:

- An instance of a virtual service is started by running a virtual service resource. A license is consumed for every running instance of the virtual service.
- An API suite is run that has a scenario that references a local virtual service and the scenario begins

The **Concurrent Virtual Service** license is returned to the license pool when the virtual service instance stops.

For example, consider you decided to have three team spaces including the initial team space, then you must configure the licenses required for each team space on the License Server based on your requirement. You must then use the URL and ID of those License Servers to configure the licenses for the team spaces.

---

Related information

[Configuring licenses for team spaces on page 140](#)

## Configuring licenses for team spaces

As a *Team Space Owner*, you can configure the License Server for the team spaces on the **Team Space License Configuration** page so that members of the team space can perform various operations such as creating a project, running tests, and so on.



## Before you begin

You must have completed the following tasks:

- Configured the licenses that you purchased on the Licensing Server.
- Been assigned the role of a *Team Space Owner* of the team space.
- Installed the product and signed up as a user on HCL OneTest™ Server.
- Optional: Created a team space. See [Creating a team space on page 131](#).

## About this task

The team spaces are logical partitions of HCL OneTest™ Server. The team space facilitates users of the same team space to work together. As a *Server Administrator*, you have access to the initial team space. You can also create multiple team spaces, if required. As a *Server Administrator*, you can assign the role of *Team Space Owner* to any other user when you create the team space on HCL OneTest™ Server. If you are the *Server Administrator*, you can add yourself as a *Team Space Owner*.

You must have a valid license to work with the team spaces that you create on HCL OneTest™ Server. You must provide a separate URL and ID of the License Server for the team spaces when you configure the License Server. If you provided the URL and ID of the License Server that is already configured for any other team space, then the configuration of the license for the team space fails with an error message. If you delete the team space, then the URL and ID of the License Server used for that team space is released. You can then use that for another team space.



**Note:** When you upgrade the server software from the previous version to the latest version, then the license configuration of the previous release applies to the initial team space. You can verify the information from the **Team Space License Configuration** page of the initial team space.

As a *Server Administrator*, you can assign the role of a *Server Administrator* to any user. For information about how to change the role, see Default user administration in **Related information**.



**Remember:** When your role is changed from a user to a *Server Administrator*, you must log out and log in again so that HCL OneTest™ Server can apply the change in your role.

1. Log in to HCL OneTest™ Server.

The **Projects** page of the initial team space is displayed.

2. Click **Dashboard**.

### Result

The **Team Space Dashboard** page is displayed.

3. Click a team space from **My Team Spaces**.
4. Click **Manage > Licenses**.

**Result**

The **Team Space License Configuration** page is displayed.

- Enter the values in the **License Server URL** and **License Server ID** fields to configure the licenses for the team space.

**Result**

The following information is displayed:

- The **License usage** section displays the number of **Named user** licenses that are currently in use.
  - The **License configuration** section displays the number of **Named user** licenses and the number of **Concurrent Virtual Service** licenses that are configured on the License Server.
- Select the **Disable automatic self-acquisition of named user licenses** option if you do not want to self-acquire **Named users** license on a first-come-first-serve basis for your team space.

The following table describes the functionality of the **Disable automatic self-acquisition of named user licenses** option:

Option	When	Description
<b>Disable automatic self-acquisition of named user licenses</b>	Cleared <input type="checkbox"/>	Any member of a team space can get a license when there are sufficient licenses are available for the team space.
	Select- ed <input checked="" type="checkbox"/>	Only such users who are added to the <b>Named Users</b> list get a license to perform the tasks in their team space. If other users try to perform any project operations in any of the team spaces, they cannot get a license even if sufficient licenses are available.

- Click **Configure** to apply the license configuration.

**Results**

You have configured the license for the team space.

**What to do next**

You can perform any of the following tasks:

- Add users to the **Named Users** list. See [Adding users to the Named Users list on page 143](#).
- Add members to your team space. See [Adding members to a team space on page 568](#).

---

 Related information

[Default user administration on page 105](#)

## Adding users to the Named Users list

As a *Team Space Owner*, you can add users to the **Named Users** list so that the licenses are always available for those users.

### Before you begin

You must have completed the following tasks:

- Been assigned the role of a *Team Space Owner* of the team space.
- Optional: Created a team space. See [Creating a team space on page 131](#).
- Configured the license for the team space. See [Configuring licenses for team spaces on page 140](#)

1. Log in to HCL OneTest™ Server and open the team space for which you want to add users.
2. Click **Manage > Licenses**.

#### Result

The **Team Space License Configuration** page is displayed.

3. Enter the name or the email ID of the user in the **Add named users** field, and then select the name from the list that displays to add members to the **Named Users** list.



**Note:** You can press **Ctrl** to select multiple users simultaneously.

4. Click **Add** to make selected users as **Named users**.

### Results

You have added users to the **Named Users** list.

## Removing or replacing named users

When you want to remove a user from the list of named users or you want to replace an existing user with another named user, you as a *Team Space Owner* can perform these changes from the **Team Space License Configuration** page.

### Before you begin

You must have completed the following tasks:



- Been assigned the role of a *Team Space Owner* for the team space that you want to modify the users from the **Named Users** list.
- Added users to the list of named users for the team space on HCL OneTest™ Server.




1. Log in to HCL OneTest™ Server and open the team space for which you want to remove or replace the users from the **Named users** list.
2. Click **Manage > Licenses**.

**Result**

The **Team Space License Configuration** page is displayed.

3. Perform the actions described in the following table to remove or replace users from the **Named Users** list:

Tasks	Actions
<p>Remove a user from the list of named users.</p>	<p>Perform the following steps:</p> <ol style="list-style-type: none"> <li>a. Locate the user that you want to remove from the <b>Named Users</b> list.</li> <li>b. Click the <b>Menu</b> icon  next to the name of the user, and then click <b>Remove</b>.</li> <li>c. Click <b>Remove</b> on the <b>Remove named user</b> dialog box to complete the action.</li> </ol> <p> <b>Remember:</b></p> <ul style="list-style-type: none"> <li>◦ When the <b>Disable automatic self-acquisition of named user licenses</b> option is cleared, and if the user who is removed from the <b>Named Users</b> list tries to perform any project operation, then the name of the user is added back to the list. The name of the user is added only if sufficient licenses are available.</li> <li>◦ When the <b>Disable automatic self-acquisition of named user licenses</b> option is selected and you remove the user from the <b>Named Users</b> list, then the following operations are impacted by the user who is removed from the list:</li> </ul>

Tasks	Actions
	 <ul style="list-style-type: none"> <li>▪ The test runs scheduled by the user might fail to run.</li> <li>▪ The test results which are in progress by the user might fail to record complete details.</li> <li>▪ The user cannot perform any of the project operations that include the generation of offline tokens that are used for long-run tests.</li> </ul>  <p><b>Note:</b> If the user is currently on HCL OneTest™ Server, then the license can be used while the current session remains active.</p>
Replace a user in the list of named users.	<p>Perform the following steps:</p> <ol style="list-style-type: none"> <li>a. Locate the user from the <b>Named Users</b> list that you want to replace with another user.</li> <li>b. Click the <b>Menu</b> icon  next to the name of the user, and then click <b>Replace</b>.</li> <li>c. Enter the email ID or name of another user that you want to add to the <b>Named Users</b> list.</li> <li>d. Click <b>Replace</b> to complete the action.</li> </ol>

**Results**

You have removed or replaced a user from the **Named Users** list.

**Viewing license details of the team space**

You can find information about the number of licenses configured, licensed users, or details of the license server on the **Team Space License Configuration** page.

**Before you begin**

The *Team Space Owner* must have configured the licenses for the team spaces on HCL OneTest™ Server.

**About this task**

You as a licensed user can create projects or use the other features in HCL OneTest™ Server only after the *Team Space Owner* has configured the licenses for the team space. If you are not a licensed user, you cannot view or perform any operations on HCL OneTest™ Server.

1. Log in to HCL OneTest™ Server and open the team space for which you want to view the license details.
2. Click **Manage > Licenses**.

**Result**

The **Team Space License Configuration** page is displayed with the following license information:

- The number of **Named user** licenses that are currently in use
- The number of **Named user** licenses and number of **Concurrent Virtual Service** licenses that are configured on the License Server
- The URL and ID of License Server
- Whether the self-acquisition of named user licenses feature is enabled or disabled
- The name and email ID of the users that are configured as named users by the *Team Space Owner*

**Results**

You have viewed the details about the licenses for the team space.

# Chapter 5. Test Author Guide

This guide describes how to create test assets in HCL OneTest™ Server and publish test assets to the Git repository that you have configured. This guide is intended for testers and test managers.


## Datasets overview

A dataset provides tests with variable data during a run. The test that uses a dataset at run time replaces a value in the recorded test with variable test data that is stored in the dataset.

In HCL OneTest™ Server, you can create a dataset and use it to replace the dataset values with original values during run time, when you want to run test assets that contain the dataset.

From the **Datasets** page, you can perform the following actions:

**Table 3. Options available in Datasets page**

Ac-tions	Descriptions
View	You can view the contents of a dataset. See <a href="#">Viewing a dataset on page 158</a> .
Cre-ate	You can create a dataset to use it during a test or schedule run. See <a href="#">Creating a dataset on page 148</a> .
Edit	You can edit a dataset when you want to run a test asset with different dataset values. See <a href="#">Editing a dataset on page 149</a> .
Con-figure	You can configure the string values in a dataset that contains variable data for tests to use when they run. See <a href="#">Editing a dataset on page 149</a> .
Delete	You can delete the dataset when it is not required in your test environment. See <a href="#">Deleting a dataset on page 160</a> .
Save As	You can save a copy of a dataset in a different folder or make a copy with a different name by clicking the <b>Menu</b> icon  and then by selecting the <b>Save As</b> option.
Branch selec-tion	You can view the dataset listed in the other branches of the repository by selecting the name of the branch from the <b>Branch</b> list. When you access the <b>Datasets</b> page for the first time after adding the repository, the global branch is set to the default branch of the repository.  When multiple repositories are added to the same project, the following events occur:

**Table 3. Options available in Datasets page (continued)**

Ac-tions	Descriptions
	<ul style="list-style-type: none"> <li>• The datasets stored in the selected branch of all the repositories are displayed.</li> <li>• All the branches in all the repositories are listed in the drop-down list.</li> <li>• If the selected branch is not present in a repository, then datasets from that repository are not displayed.</li> </ul> <p>To differentiate a common branch across multiple repositories added to the project, the tooltip is not displayed in front of the common branch name in the list.</p>

**Notes:**

- Only a project **Owner** or a project member whose role is a **Tester** can view and edit the dataset.
- To view the content of an encrypted dataset, you must provide an encryption key that you set while encrypting the dataset column.
- As a **Viewer**, when you try to access the **Datasets** page, an error message is displayed because you do not have the permission to view this page.

## Creating a dataset

You can create datasets in HCL OneTest™ Server to replace the existing dataset values with new dataset values during a test or schedule run.

### Before you begin

You must be a member of the project with the **Owner** or **Tester** role and must have completed the following tasks:

- Created a project in HCL OneTest™ Server. See [Adding a project on page 582](#).
- Configured the Git repository in your project. See [Adding repositories to a server project on page 583](#).

### About this task

When you create a dataset in HCL OneTest™ Server, it always creates 1 Row, 2 Column (1R X 2C) dataset. Later, you can edit the dataset by adding some rows and columns that you want, to add the data in it.

1. Go to the HCL OneTest™ Server URL.
2. Enter your user name and password, and then click **Login**.
3. Open your project from the HCL OneTest™ Server UI.
4. Go to the **Datasets** page, and then click **Create dataset**.



5. Enter a name for the dataset in the **Asset name** field, and then select the place where you want to save the dataset in the **Location** drop-down list.
6. Click **Create**.

#### Result

The new dataset opens in a CSV Editor in a web browser. The dataset created is listed on the **Datasets** page.

#### Results

You have created a dataset in your project.

#### What to do next

- You can add, modify, or remove data in the dataset. See [Editing a dataset on page 149](#).
- You can publish the dataset to the Git repository so that other members of the project can use the dataset. See [Publishing a dataset on page 157](#).

## Editing a dataset

You can add, modify, remove, import, or export data from a dataset by using the CSV Editor. The working principle of the CSV Editor is similar to that of a spreadsheet.

#### Before you begin

You must have created a dataset or configured a repository that contains the dataset.

#### About this task

If you are a project **Owner** or **Tester** in HCL OneTest™ Server V10.1.0 or later, you can perform basic tasks in the CSV Editor by right-clicking any row, column, or cell in the dataset to organize your data in a better way. For example, you can perform tasks such as updating data in a cell, inserting or deleting rows and columns, or renaming column names.

When you edit the dataset in the CSV Editor, you can use the following keyboard shortcuts to control the cursor selection in the CSV Editor:

- **Tab** - To move the cursor control to the next available option.
- **Shift-Tab** – To move the cursor control to the previous option.
- **Shift+F10** – To open the context menu from the dataset cell.

After you edit the dataset, you can save the changes made to the dataset, and then you can publish the dataset to the Git repository. If you save and close the edited dataset, the **Changes** page lists the edited dataset and later you can publish to the Git repository for other members to use.

When you or other members of the project edit the same dataset, you can see an icon with the initials of the member next to the name of the dataset on the **Datasets** and **Changes** pages. If you do not see the icon, you must refresh the

**Datasets** page to view the icons. The **Changes** page denotes conflicting edits when another member publishes their edited dataset first.

The screenshot shows two parts of the HCL OneTest interface. The top part is a table listing datasets:

Dataset name	Table matrix	Actions
> AdditionalLocations LogData/Logical/Locations/AdditionalLocations <small>AS JD John Doe</small>	2 Col × 7 Row	
> Locations LogData/Logical/Locations/Locations.sit	3 Col × 3 Row	

The bottom part is a 'Changes' table:

Asset name	Change	Last changed by	Last updated	Actions
<input type="checkbox"/> AdditionalLocations LogData/Logical/Locations/AdditionalLocations	<b>Update</b> <small>This asset has conflicting edits</small>	Amy Smith	7 minutes ago	

When a member with conflicting edits tries to publish the edited dataset, an error about conflicting changes is displayed. However, the member can use the **Save As** option to save and publish a copy of the dataset edits under a new asset name. The member must discard the edits that were made to the original dataset.

For example, consider a scenario when Amy Smith and John Doe edit the same dataset, and Amy Smith edits and publishes the dataset. On the **Changes** page, a message that indicates a conflict is displayed when John Doe tries to publish the same dataset.

**Note:** You cannot resize the width of rows in the CSV Editor. When you have a large amount of data in a cell, you can right-click the cell and select **Copy** (or Ctrl+C), and then paste it into a text-editing program to view the content. Alternatively, you can hover the mouse over the cell to view the content.

When you have a CSV file that has data separated from a character, then you can import that CSV file into the dataset. You can select any of the following separator characters from the **Configure Dataset** window, and the selection can be the separator character that you used in the CSV file:

- Comma
- Semicolon
- Space
- Tab
- Other

Consider that you have the data in the CSV file in the following format:

Name;CCNum
John;1234 5678 1234 5678
Bob;1122 3344 5566 7788
Amy;2233 4455 6677 8899


When you import the CSV file in the dataset, and then select the separator value as **Semicolon**, the data in the dataset is displayed as follows:

	Name	CCNum
1	John	1234 5678 1234 5678
2	Bob	1122 3344 5566 7788
3	Amy	2233 4455 6677 8899

If you want the data in its original format, that is, a semicolon (;) character to separate the data, then you can choose any other separator value from the **Configure Dataset** window.





**Note:** The default separator value is **Comma**.








1. Go to the **Datasets** page and find the dataset which you want to edit.
2. Click the **Edit** icon  from the **Actions** column of the dataset.





#### Result



The dataset opens in the CSV Editor in a web browser.


3. Perform the following actions to use the options available in the CSV Editor:


Options	Actions
Find and Replace 	<p>To find:</p> <ol style="list-style-type: none"> <li>a. Click the <b>Find and Replace</b> icon .</li> <li>b. Enter the content that you want to search in the <b>Find</b> field.</li> <li>c. Select any or all the following options to find the search content more effectively: <ul style="list-style-type: none"> <li>▪ Select the <b>Case sensitive</b> checkbox to search the content that is the exact letter case of the content entered in the <b>Find</b> field.</li> <li>▪ Select the <b>Match entire cell contents</b> checkbox to search for cells that contain only the characters that you have entered in the <b>Find</b> field.</li> <li>▪ Select the <b>Search using regular expression</b> checkbox to search the pattern that matches strings.</li> </ul> <p>For example, to search a cell that contains any number between 0 to 9, do the following:</p> <ol style="list-style-type: none"> <li>i. Enter <code>\d</code> in the <b>Find</b> field.</li> <li>ii. Select the <b>Search using regular expression</b> checkbox.</li> <li>iii. Click <b>Find</b>.</li> </ol> </li> </ol>

Options	Actions
	<p>d. Click <b>Find</b>. If the text is found, the cell containing that text is selected.</p> <p>e. Click <b>Find</b> again to find further instances of the search text.</p> <p>To find and replace:</p> <p>a. Click the <b>Find and Replace</b> icon .</p> <p>b. Enter the content that you want to search in the <b>Find</b> field.</p> <p>c. Enter the content that you want to replace in the <b>Replace</b> field.</p> <p>d. Select any or all the following options to find and replace the content more effectively:</p> <ul style="list-style-type: none"> <li>▪ Select the <b>Case sensitive</b> checkbox to find the content that is the exact letter case of the content entered in the <b>Find</b> field.</li> <li>▪ Select the <b>Match entire cell contents</b> checkbox to find and replace for cells that contain only the characters that you have entered in the <b>Find</b> and <b>Replace</b> fields.</li> <li>▪ Select the <b>Search using regular expression</b> checkbox to find and replace the pattern that matches strings.</li> </ul> <p>e. Click <b>Replace</b> to replace the individual instances.</p> <p>f. Click <b>Replace All</b> to replace every instance of the content throughout the dataset.</p>
Undo 	<p>a. Click the <b>Undo</b> icon .</p> <p>b. Select the recent changes from the list that you want to undo, and then click the list.</p> <p>The <b>Undo</b> option undoes anything you do in the dataset. The CSV Editor saves the unlimited undo-able action. You can perform the undo action even after you save your changes made to the dataset.</p>
Redo 	<p>a. Click the <b>Redo</b> icon .</p> <p>b. Select the recent changes from the list that you want to redo, and then click the list.</p> <p>The CSV Editor saves the unlimited redo action.</p>
Import 	<p>When you have a large amount of data stored in a CSV file, you can import that into a dataset instead of creating a new dataset. You must have a .csv file that contains variable data to import into a dataset.</p> <p>a. Click the <b>Import</b> icon .</p> <p>b. Click <b>Choose File</b> and select the CSV file that you want to import in the <b>Import File</b> dialog box.</p> <p>c. Select one of the following options to append or overwrite data in the dataset:</p> <ul style="list-style-type: none"> <li>▪ Click <b>Overwrite</b> to add the rows and columns from the selected CSV file from the beginning of the dataset.</li> <li>▪ Click <b>Append</b> to add rows and columns from the selected CSV file to the end of the dataset.</li> </ul> <p>d. Select the <b>First row contains headers</b> checkbox if your CSV file contains the header.</p>

Options	Actions
Export 	<p>You can export variable data from the dataset into a CSV file to reuse in future tests when required. You must have a dataset that you want to export.</p> <p>Click the <b>Export</b> icon  to download the dataset as a CSV file.</p>
Set as current row	<p>During the test run, if you want variable data to be selected from a current row instead of the first row in a dataset, right-click any cell in a row and select <b>Set as current row</b>.</p> <p>Also, you can set the current row from the <b>Datasets</b> page by clicking <b>Menu</b>, and then the <b>Configure</b> option.</p> <p><b>When rows are deleted:</b></p> <p>If you delete any row between row 1 to current row, the current row data is taken from the next row. For example, when you set the current row as 6, and then you delete any row between row 1 to row 6, the current row remains at row 6, but the content of row 7 is moved to row 6.</p> <p><b>When rows are inserted:</b></p> <p>If you insert any new row between row 1 to the current row, the current row data is taken from the previous row. For example, when you set the current row as 6, and then you insert any row between row 1 to row 6, the current row remains at row 6, but the content of row 5 is moved to row 6.</p>
Dataset configuration settings 	<p>In the <b>Configure Dataset</b> window, you can change the row and column settings and configure the string values in the dataset that contains variable data for tests to use when they run.</p> <ol style="list-style-type: none"> <li>Click the <b>Menu</b> icon , and then select the <b>Configure</b> option.</li> <li>Select any of the separator values that you used in the CSV file.</li> </ol> <p>The available options are <b>Comma</b>, <b>Semicolon</b>, <b>Space</b>, <b>Tab</b>, and <b>Other</b>. In the CSV file, if you have any other separator characters other than the available options, then you can select the <b>Other</b> option, and then can specify a value.</p> <p>For example, if the data in the CSV file is separated by a character #, then select the <b>Other</b> option and enter # in the field.</p> <ol style="list-style-type: none"> <li>Configure the following options to change the row and column settings:</li> </ol>

Options	Actions
	<ul style="list-style-type: none"> <li>▪ <b>Column header</b> - Use an up-down control button to increment or decrement the value of the column header.</li> <li>▪ <b>Data start point</b> - Use an up-down control button to increment or decrement the value of the data starting pointer.</li> <li>▪ <b>Current row</b> - Use an up-down control button to increment or decrement the value of the current row.</li> </ul> <p>d. Configure the following options to change the string values in the dataset:</p> <ul style="list-style-type: none"> <li>▪ <b>Treat as null</b> - Enter a string value that is to be treated as null when running the test.</li> <li>▪ <b>Treat as empty</b> - Enter a string value that is to be treated as empty when running the test.</li> </ul> <p>For example, when you run the test and the data 123 in the dataset to be treated as empty, then you can specify 123 in the <b>Treat as empty</b> field.</p> <ul style="list-style-type: none"> <li>▪ <b>Treat empty text as null</b> - Select this field when you want the dataset that contains any blank cells, and the value of those blank cells to be interpreted as null.</li> </ul> <p>e. Click <b>Update</b> to apply the changes.</p>
Discard 	Click the <b>Menu</b> icon  and select <b>Discard</b> to discard the changes made to the dataset.

4. Click the **Save** icon  to save the changes made to the dataset.

5. Click the **Publish** icon  to publish the dataset to the Git repository, and then close the CSV editor.

## Results

You have edited the dataset.

## Dataset encryption

Encrypted datasets are useful when you want to run tests that contain confidential information such as a set of passwords or account numbers.

When you run a test that uses an encrypted dataset, then you must provide an encryption key for decrypting the encrypted data in columns so that the data can be used in the test. If the test uses data from the multiple encrypted dataset columns, you must enter the same encryption key for every encrypted dataset column that the test uses.

When you run the test that uses the dataset with the encrypted column, the value of the column is decrypted at a run time. The data in the column is sent as a clear-text string in requests to the server. The actual values of the encrypted dataset variables are not displayed in the test log. The test log displays asterisks for the encrypted dataset variables.

You can use only one encryption key to encrypt data in the columns in any dataset.

### Important:

The encryption keys that you use to encrypt data in a dataset are not stored on the server nor can be retrieved from the server. Therefore, you must remember to store the encryption keys in a secure location. You must use the same encryption keys to perform the following operations:


- View the encrypted values
- Decrypt data
- Enable the use of the encrypted dataset during test runs

## Encrypting a dataset column

To secure test data, you must encrypt datasets. You can encrypt data in the columns of a dataset by using an encryption key. When you run a test that utilizes a dataset with encrypted variables, you must enter the encryption key for the encrypted column that the test uses.

### Before you begin

You must have created a dataset. See [Creating a dataset on page 148](#).

1. Go to the HCL OneTest™ Server URL.
2. Enter your user name and password, and then click **Login**.
3. Open your project from the HCL OneTest™ Server UI.
4. Go to the **Datasets** page and find the dataset that you want to encrypt.
5. Click the **Edit** icon  from the **Actions** column of the dataset.

#### Result

The dataset opens in the CSV Editor in a web browser.

6. Right-click any cell in a column that you want to encrypt and select **Encrypt column data**.

#### Result

The **Encrypt Column** window is displayed.

7. Enter an encryption key in the **Encryption Key** field to encrypt the data in the column.



**Remember:** When you have already encrypted other columns in the dataset, you must enter the same encryption key that you used previously. You can use only one encryption key to encrypt columns in a dataset.



**Important:**

The encryption keys that you use to encrypt data in a dataset are not stored on the server nor can be retrieved from the server. Therefore, you must remember to store the encryption keys in a secure location. You must use the same encryption keys to perform the following operations:

- View the encrypted values
- Decrypt data
- Enable the use of the encrypted dataset during test runs

8. Click **Encrypt Column**.

**Result**

Asterisks are displayed instead of actual data for the encrypted column.

**Results**

You have encrypted the dataset column in your project.

**What to do next**


You can publish the dataset to the Git repository so that other members of the project can use the dataset. See [Publishing a dataset on page 157](#).

## Decrypting a dataset column

To view the content of an encrypted dataset, you can decrypt the dataset column. Removing encryption from a dataset revokes the protection offered to the test data.

**Before you begin**

You must have created at least one dataset and encrypted the dataset with an encryption key. See [Creating a dataset on page 148](#) and [Encrypting a dataset column on page 155](#).

1. Go to the HCL OneTest™ Server URL.
2. Enter your user name and password, and then click **Login**.
3. Open your project from the HCL OneTest™ Server UI.
4. Go to the **Datasets** page and find the dataset that you want to decrypt.
5. Click the **Edit** icon  from the **Actions** column of the dataset.

**Result**

The dataset opens in the CSV Editor in a web browser.



- Right-click encrypted cells that display the contents with asterisks, and then select **Decrypt column data**.

**Result**

The **Decrypt Column** window is displayed.

- Enter the encryption key that you used to encrypt the data in the column in the **Encryption Key** field.
- Click **Decrypt Column**.

**Result**

Asterisks are replaced with the actual data in the decrypted column.

**Results**

The encryption is removed from the selected column in the dataset. When you run a test that uses a dataset that contains decrypted data, the variable data is substituted for the original data in the recorded test without prompting for the encryption key.

**What to do next**

You can publish the dataset to the Git repository so that other members of the project can use the dataset. See [Publishing a dataset on page 157](#).

## Publishing a dataset

When you create or edit any datasets in HCL OneTest™ Server, you can publish your changes to the Git repository. Therefore, when you publish a dataset, other members in the project can use your dataset in their test assets run, if required.

**Before you begin**

You must have created, edited, or deleted dataset assets in HCL OneTest™ Server.


**About this task**

The **Changes** page lists dataset assets that are modified. You can publish all datasets or a single dataset listed in the **Changes** page to the Git repository by selecting the appropriate checkboxes.

<input type="checkbox"/>	Asset name ▲	Change	Last changed by	Last updated	Actions
<input type="checkbox"/>	ds_server Project4/ds_server.csv	Deletion ▼	User1	a few seconds ago	⋮
<input type="checkbox"/>	DS1 Project1/DS1.csv	Update ▼	User1	a few seconds ago	⋮

HCL OneTest™ Server processes one publish request at a time. Therefore, when multiple users attempt to publish the same dataset, the request that reaches first from the users is processed. The other users receive an error message, and they are unable to publish.

When you modify the dataset and publish it to the Git repository, the other members who have access to that dataset can view the updated dataset.

You can also perform the following actions from the Changes page by clicking the **Menu** icon .

- **Edit** - If you want to make any last-minute updates to the dataset before publishing.
- **Discard** - To remove the changes that you made to the dataset asset.
- **Save As** - To save a copy of dataset.

1. Go to the HCL OneTest™ Server URL.
2. Enter your user name and password, and then click **Login**.
3. Open your project from the HCL OneTest™ Server UI.
4. Go to the **Changes** page, find the dataset that you want to publish.
5. Select the checkbox that precedes the **Dataset name** to publish to the Git repository, and then click **Publish**.
6. Enter a description for the changes made to the dataset, and then click **Publish**.

#### **Result**

A message is displayed for successful pushing of changes to the Git repository.

### **Results**

You have published the modified dataset into the Git repository.

## Viewing a dataset


When test assets includes datasets, you can view the contents of a dataset from HCL OneTest™ Server. The datasets residing in the Git repository are listed in the **Datasets** page.

### **Before you begin**

You must be a member of the project with the **Owner** or **Tester** role and must have completed the following tasks:

- Created a project in HCL OneTest™ Server. See [Adding a project on page 582](#).
- Configured the Git repository in your project. See [Adding repositories to a server project on page 583](#).
- Created at least one dataset and encrypted the dataset with an encryption key. See [Creating a dataset on page 148](#) and [Encrypting a dataset column on page 155](#).
- Created a classification for an encrypted dataset. See [Creating a classification on page 605](#).

1. Go to the HCL OneTest™ Server URL.
2. Enter your user name and password, and then click **Login**.
3. Open your project from the HCL OneTest™ Server UI.

- Go to the **Datasets** page, find the dataset that you are interested in, and then expand the dataset by clicking the **Expand** icon .





**Note:** You can click the **Dataset Name** field to sort the datasets by name in alphabetical order. Alternatively, you can use the **Search** field to search the dataset by name.

### Results

You have viewed the contents of the dataset.

## Viewing an encrypted dataset

You can use the **Dataset** page to view the contents of an encrypted dataset from HCL OneTest™ Server.

- Go to the **Datasets** page, find the encrypted dataset that you are interested in, and then expand the dataset by clicking the **Expand** icon .
- Perform the following steps, if you have created a classification but yet to add the encrypted dataset to it:
  - Find the encrypted dataset that you are interested in, and then click the **Lock** icon  from the **Actions** column of the dataset.
  - In the **Change the classification for the dataset** window, select a classification from the list and enter the encryption key set for the dataset.
  - Click **Save** to save the classification details.
- Right-click the encrypted column and click **Show encrypted data**.

You can right-click the decrypted column and select **Hide encrypted data** to encrypt the data again.



**Important:** The **Show encrypted data** option is available only when you have added the encrypted dataset to the respective classification.

### Results


You have viewed the contents of an encrypted dataset.

## Changing classification for an encrypted dataset

To move the encrypted dataset from one classification to another, you can change the classification for an encrypted dataset from the **Dataset** page.

### Before you begin

You must have created at least two or more classifications. See [Creating a classification on page 605](#).

- Go to the **Datasets** page and find an encrypted dataset for which you want to change the classification.
- Click the **Lock** icon  from the **Actions** column of a dataset.
- In the **Change the classification for the Dataset** window:

- a. Choose the classification from the list.
- b. Enter the encryption key for the dataset that was set while encrypting the dataset column.
- c. **Save** the classification details.



**Remember:** You can perform this task only if you know the encryption key set for the dataset while encrypting the dataset column.

## Results

You have changed the classification for an encrypted dataset.

---

Related information

[Managing an encrypted dataset on page 605](#)

[Creating a classification on page 605](#)

## Deleting a dataset

You can delete the dataset when it is not required in your test environment.

### Before you begin

You must have at least one dataset asset in your Git repository that you have configured.

### About this task


You can delete datasets in HCL OneTest™ Server in the following scenarios:

- Datasets that you created in the desktop clients and that are cloned to the repository in your project on HCL OneTest™ Server.
- Datasets that you created in HCL OneTest™ Server and that are published to the repository in your project.
- Datasets that are in the `.csv` file format.

You cannot delete the following datasets:

- Datasets that you created or edited in HCL OneTest™ Server and that are not published to the repository in your project.
- Datasets that are in the `.sit` file format.

1. Go to the HCL OneTest™ Server URL.
2. Enter your user name and password, and then click **Login**.
3. Open your project from the HCL OneTest™ Server UI.
4. Go to the **Datasets** page, find the dataset that you want to delete.

5. Click the **Menu** icon  from the **Actions** column of a dataset, and then click **Delete**.
6. Clear the **Publish delete to <branch name of the configured Git repository>** checkbox to delete the dataset from your project.

When you clear this option, the deletion of the dataset does not reflect in the configured Git repository. Therefore, the other members of the project can still use the dataset that you have deleted. If you want, later you can publish the dataset to the Git repository from the **Changes** page.



**Note:** By default, **Publish delete to <branch name of the configured Git repository>** field is selected.

7. Enter a description for deleting the dataset in the **Description of change** field.
8. Perform the following action:

Option	When	Action
<b>Publish delete to &lt;branch name of the configured Git repository&gt;</b>	Selected <input checked="" type="checkbox"/>	Click <b>Delete and Publish</b>
	Cleared <input type="checkbox"/>	Click <b>Delete</b>

#### Result

A message is displayed for the successful deletion of the dataset.

## Generation of test data

To test an application, you use data that is passed to get real-time results. Since creating test data manually is time-consuming, you can use HCL® OneTest™ Data, an automated tool to generate random test data. You can generate the test data in various file formats.

Test data is a core element in the process of testing any application. A sufficient amount of data is necessary to test all the possible scenarios of any application. HCL® OneTest™ Data requires a schema to generate the test data.

When you want to generate the test data, then you must fabricate a schema or generate a schema by using any external resource. After you fabricated a schema, you must define types of the schema and set properties for each type. You can then generate the test data for one or multiple schemas at a time. You must generate multiple schemas in HCL® OneTest™ Data by using the schemas created in the JDBC supported databases.

## Schema fabrication

A schema is a graphical structure to represent different types of data. You can fabricate a schema for the generation of test data.

HCL® OneTest™ Data supports the following ways to fabricate the schema:

- Import a schema from local file system into HCL® OneTest™ Data.
- Generate a schema in external resource by using HCL® OneTest™ Data.
- Create a schema by using HCL® OneTest™ Data.

You can fabricate a schema by using various methods. You can also manage the schema for your project to generate the test data.

## Schema import

HCL® OneTest™ Data requires a schema to generate the test data. If you have a schema in your local file system, then you can import the same schema into your project.

HCL OneTest Data supports the import of the schema, which is either in JavaScript Object Notation (JSON) or XML Schema Definition (XSD) file format.

After you import the schema, you can perform all the following tasks to the imported schema:

- Defining types
- Setting type properties
- Setting restrictions
- Applying functions

 **Restriction:** You cannot modify the properties of the types in an XSD or a JSON schema.

## Importing XSD or JSON files from the local file system

When you have an XSD or a JSON format file in your local file system, you can import the same files from the local file system into your HCL® OneTest™ Data project.

### Before you begin

- You must have created a project.
- You must have an XSD or a JSON format file in your local file system.

### About this task

XSD or JSON files are in schema format. To create a schema, you can import an existing schema that is in the XSD or JSON format from your local file system.

 **Restriction:** You cannot modify all the properties of the types in an XSD or a JSON schema.

The following table shows the list of properties supported by JSON or XSD schema:

Properties	Read-only JSON schema	Editable JSON schema	XSD schema
JavaScript rules	Yes	Yes	No
Regular expression	Yes	Yes	No
Restrictions	No	Yes	No
Default values	No	Yes	No

**Note:**

If a schema that you import resembles the JSON structure, then that schema is represented as a *read-only* JSON schema. However, when a schema that you import resembles the JSON data, then that schema is represented as an *editable* JSON schema.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

**Result**

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

**Result**

The **Data Fabrication** page is displayed.

4. Click the **Schemas** tab.
5. Click **XSD/JSON** from the **Schemas menu**.

**Result**

The **Import XSD/JSON** dialog box is displayed.

6. Select the JSON or XSD file that you want to import.
7. Provide a schema name with a brief description and keywords as **Tags**.
8. Click **OK** and save the changes.

**Results**

You have successfully imported XSD or JSON files from the local file system.

**What to do next**

You must define the types in the schema. See [Defining Types on page 171](#).

You can also perform the following actions:

- View the imported file listed under the **Schemas** tab.
- Use the **Search** field to search for any schema from the existing schemas.
- Sort the list of schemas by name, folder, and recently used schemas.

---

Related information

[JSON Schemas](#)

## Schema generation

When you want to generate the test data in any external resource, you must generate the schema by using that specific resource.

To generate the schema, you must establish a connection between HCL® OneTest™ Data and the external resource.

HCL® OneTest™ Data supports the establishment of connections with the following external resources:

- Java Database Connectivity (JDBC)
- Systems, Applications, and Products in Data Processing (SAP)
- MongoDB
- Excel



**Note:** After you generate schema by using any external resource, you can perform all the following tasks to the generated schema in HCL® OneTest™ Data:

- Defining types
- Setting type properties
- Setting restrictions
- Applying functions

The following topics show how HCL® OneTest™ Data connects with an external resource and generates the test data.

Tasks	More information
Generate test data in JDBC-supported databases	<a href="#">HCL OneTest Data support to write the generated test data in JDBC-supported databases on page 191</a>
Generate test data for SAP IDOC or DXOB	<a href="#">HCL OneTest Data support to generate test data for an SAP IDoc or DXOB file on page 198</a>
Generate test data for SAP BAPI	<a href="#">HCL OneTest Data support to generate test data for SAP BAPI on page 201</a>



Tasks	More information
Generate test data in MongoDB	<a href="#">HCL OneTest Data support to write the generated test data in MongoDB on page 205</a>
Generate test data for an Excel file	<a href="#">HCL OneTest Data support to generate test data for an Excel file on page 209</a>

## Schema creation

When you do not have an existing schema or you do not want to generate a schema in any database, then you can create a schema by using HCL® OneTest™ Data.

To create a schema, you need to design a schema by defining the types, setting the properties for each type of schema, and designing a structural view of schema. You can also set restrictions or apply functions to the defined types.

## Composing a schema

To generate the test data, you require a schema. You can compose a new schema if you do not have an existing schema or you do not want to generate the schema by using any external resource.

### Before you begin

You must have a project.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click **New** from the **Schemas**. Otherwise, select **Create a schema** from **Quick Links**.

#### Result

The **New Schema** dialog box is displayed.

5. Provide a schema name.
6. Select the default project folder or enter any project folder name.
7. Enter keywords as tags which can help for quick search.
8. Provide a brief description of the schema and click **OK**.

#### Result

The schema opens in the schema designer as Root as the data dictionary. You can view the schema name as a tab in the workspace.

### Results

The schema is displayed in the schema designer.

### What to do next

You must define the types in your schema. For information about defining types, see [Defining types on page 171](#).

## Importing data from local file system

If you have data or a sample file, you can import it into your project and can use that data to design a schema.

### Before you begin

- You must have created a project.
- You must have a data file in your local file system in CSV or Excel files.

### About this task

The **Files** tab on the **Data Fabrication** page is one of the design components of a project. By clicking this tab, you can create a file. This tab helps you to import an existing file from the local file system, for which you want to generate test data.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click the **Files** tab.
5. Click **New** from the **Files** to create a file.

#### Result

The **New File** dialog box is displayed.

6. Select a file that you want to import from your local file system.
7. Provide a name to a file, folder details, keyword as **Tag** for quick search, and a brief description of the file.
8. Click **OK** to import and create a file.

### Results

You achieved the following results:

- View the list of all the files imported for this project in the **Files** tab.
- Use the **Search** field to search any file from the existing files.
- Sort the list of files by name, folder, and recently used files.

### What to do next

After you have created a file into your project, you can import a file into schema designer to create a schema.

---

Related information

[Integrating sample files into a schema on page 179](#)

## Importing files into schema designer

To create a schema by using the data you imported, you must import the data file into schema designer.

### Before you begin

You must have imported a data file into your project from the local file system.

### About this task

When you want to create a schema, you require data. You can use the data which you have imported from your local file system.

1. Click **Import** from the **Schemas**.

#### Result

The **Import** dialog box is displayed.

2. Select the file type as CSV.
3. Select the file that you want to import from the **Sample File Path** and define the import properties of the data.
4. Provide a schema name for this data and click **Import**.

#### Result

The schema appears in the schema designer.

5. Save the changes.

### Results

You can view the imported file listed under the **Schemas** tab.

### What to do next

You must define the types in the schema. For information about defining types, see [Defining types on page 171](#).

## Copying artifacts of a schema to another schema

When a schema in your project includes artifacts, and you want to reuse those artifacts in another schema in the same project, then you can copy the artifacts of a schema to another schema by using HCL® OneTest™ Data.

### Before you begin

You must have a schema in your HCL® OneTest™ Data project.

### About this task

While you copy the artifacts of the schema, you can either create a schema or select an existing schema.



**Restriction:** You cannot copy an XSD or a JSON schema to another schema.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

**Result**

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

**Result**

The **Data Fabrication** page is displayed.

4. Open your schema from the **Schemas** tab.
5. Select a category, group, or item type from **Dictionary**, and then click **Copy to Schema** from the menu.

**Result**

The **Copy Type** dialog box is displayed.

6. Toggle the **Create New Schema** switch to select the destination schema. You can either create a schema or select an existing schema.

**Choose from:**

- To copy the artifacts of a schema into a new schema, go to step 5.
  - To copy the artifacts of a schema into an existing schema, go to step 6.
7. Copy the artifacts to a new schema by performing the following steps:
    - a. Toggle the **Create New Schema** switch to the on position to create a schema.
    - b. Enter a name for the new schema.
    - c. Toggle the **Copy Child Types** switch to the on position to copy all the child types of the selected type.
    - d. Toggle the **Copy Referenced Types** switch to the on position if you want to copy the types referenced as the components of the group.



**Note:** The referenced types are always copied to the same path as the source schema.

8. Copy the artifacts to an existing schema by performing the following steps:
  - a. Toggle the **Create New Schema** switch to the off position to select an existing schema.
  - b. Select a schema from the **Schema** drop-down list.
  - c. Toggle the **Retain Same Type Paths** switch to the on position if you want to copy the types to the same path in the destination schema.



**Note:** If you toggle the switch to the off position, then you must provide the type path in the **Type Path** field.

- d. Toggle the **Copy Child Types** switch to the on position to copy all the types and the child types of the selected type.
  - e. Toggle the **Copy Referenced Types** switch to the on position if you want to copy the types referenced as components of the group.
  - f. Toggle the **Overwrite Existing Types** switch to the on position to overwrite the type in the destination schema that already exists.
9. Click **Copy**.
  10. Save the changes.

### Results

You have successfully copied the artifacts of a schema into another schema.

## Schema management

In a schema you can perform certain tasks such as editing and deleting of a schema. You can also split a schema view.

### Editing a schema

At any point in time, you can edit the details of a schema of your project.

#### Before you begin

You must have completed the following tasks:

- Logged into HCL OneTest™ Server.
- Created schemas in HCL® OneTest™ Data. See [Schema fabrication on page 161](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.


#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click the **Schemas** tab to view the list of schemas.
5. Identify the schema that you want to edit.
6. Click the Menu icon  and then click **Edit**.

#### Result

The **Edit Schema** dialog box is displayed.

7. Edit the schema details and click **OK**.

## Results

You have edited the selected schema.

## Deleting a schema

If you do not need a schema anymore, you can remove that specific schema from your project.

### Before you begin

You must have completed the following tasks:

- Logged into HCL OneTest™ Server.
  - Created schemas in HCL® OneTest™ Data. See [Schema fabrication on page 161](#).
1. Log in to HCL OneTest™ Server and open the team space that contains your project.
  2. Open your project.


#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click the **Schemas** tab to view the list of schemas.
5. Identify the schema that you want to delete.
6. Click the Menu icon , and then click **Delete**.
7. Click **Yes** to confirm and save the changes.

## Results

You have deleted the selected schema.

## Splitting a schema view

The schema designer displays a single schema. However, you can split the schema designer to view two schemas simultaneously. This is useful for copying, editing, or viewing different components of very large schemas.

## Schema design overview

When you compose a schema by using HCL® OneTest™ Data you must define the types of the schema. You can also modify the types of the schema when you import any schema from local file system or use the schema generated by using any external resource.

You can find information about the tasks that you must perform to design and create the structural view of the schema in your project.

## Defining types

For optimum test data generation, you must define entire data of schema into types. Types are the components of schemas. After you design schema in the schema designer, you must define entire data as types by using type designer.

## Adding types to a schema

The types in the schema are arranged in a hierarchy. A new schema by default has Root type. You can add subtypes to the Root type to define a schema.

### Before you begin

You have a schema for your project.

### About this task

Types are the set of data objects. A new schema has a single Root type. All types are added as subtypes of the Root type and then as subtypes of other types.

1. Select the type under which you want to add a subtype. For example, select "Root".
2. From the menu of the selected type, click **Add**.

### Result

The subtype is created of the similar type as of the selected type and it appears under the selected type.



**Note:** You cannot add any type to a JSON or an XSD schema.

3. Save the changes.

### Results

You can view the nested structure of types.

### What to do next

You must define the properties of the types of the schema. For information about how to setup type properties, see related links at the end of this page.

## Copying a type

To create more types with similar properties, you can copy an existing type.

### About this task

In addition to using the drag-and-drop method, you can use the menu to copy a type.

1. Select the type that you want to copy and click **Copy** from the menu.
2. Select the target type—the type under which you want to copy the given type.
3. Click **Paste** from the menu of the target type.

## Results

You can view a new type with the properties similar to an existing type.

## What to do next

You can modify the name, description, and the properties of the copied type.

## Deleting a type

When you want to remove any type from the schema, you can delete it.

1. Select the type that you want to delete and click **Delete** from the menu.
2. Click **Yes** to confirm and save the changes.

You can select more than one type simultaneously by using **Multiselect** option.



**Note:** You cannot delete any type of a JSON or an XSD schema.

## Results

The selected type is deleted.

---

Related information

[Objects, types, and classes on page 633](#)

## Setting type properties

After you add types to your schema, you must set type properties.

### Before you begin

You must have a schema with types.

1. Select a type and click the menu of the type.
2. Click **Properties**.

#### Result

The **Properties** dialog box is displayed.

3. Enter the name, class, and description details of the type.
4. Choose the class of the type as **Category**, **Group**, or **Item**.

#### Result

The properties of the selected type is displayed.

5. Define the required properties for the selected type.
6. Click **Advanced properties** to provide additional property details of the type.

You can set multiple properties for an item type.





**Note:** If you want to import any JSON or XSD format schema, you cannot modify the properties of any type of the schema.

7. Save your changes.

### Results

You have set the type properties of schema and saved them successfully.

### What to do next

When you finish setting the properties for all the types of your schema, you must prepare a structural view of the schema by using the drag-and-drop method. See [Designing a structural view of the schema on page 184](#).



### Note:

If you set multiple properties for an item type, then HCL® OneTest™ Data considers the following order of priority while executing the properties:

- Data generation for multiple schemas based on schema integrity
- Pairwise data property
- Property to set Javascript rules
- Setting output rule in **Rule Editor**
- Property to set **Implied Default** value
- Regular expression property
- Property to set weightage values by using a file
- Property to set restrictions

Read the following topics to understand how to set certain properties for any item type:

---

Related information

[Type properties on page 633](#)

## Value assignation for item types

When you want to use specific item type values to generate the test data, then you can assign those values to the item types. To assign the values to any item type, you can either import the item type values or add the values manually.

### Prerequisites

You must have completed the following tasks:

- Created a project in HCL OneTest™ Server.
- Created a schema in HCL® OneTest™ Data.

You can assign values to any item type by using any one of the following methods:

- [Assigning values for item types from a file on page 174](#)
- [Inserting values for item types manually on page 175](#)

## Assigning values for item types from a file

When a file contains values for an item type on your computer, then you can assign those values to the item type to generate the test data.

### Before you begin

You must have created a file with item type values on your computer.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click **Files** and import the file of item type values from your computer into your project.
5. Open your schema from the **Schemas** tab.
6. Select the item type for which you want to assign the values.
7. Select **Restrictions** from the **Item Properties** drop-down list of the **Properties** dialog box.
8. Enable the **Values from file** toggle button.
9. Provide the details of the file that you want to use to get the values of the item type in the **Values file** field.  
You can use any one of the following methods to provide the details of the file:

#### Choose from:

- Upload the file from your computer.
  - Provide the name of the imported file with an extension.
10. Enter the column number of the file where the item type values are specified in the **Values column number** field.
  11. Save the changes.

### Results

You have successfully assigned the values for item types from a file.

### What to do next

You must prepare the structural view of the schema. See [Designing a structural view of the schema on page 184](#).

## Inserting values for item types manually

When you want to assign specific values to an item type that does not exist in any file on your computer, then you can assign those values by inserting them manually.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.




### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

### Result

The **Data Fabrication** page is displayed.

4. Open your schema from the **Schemas** tab.
5. Select the item type for which you want to assign the values.
6. Assign a default value or multiple values.
7. Perform the following steps to assign a default value to the item type:
  - a. Select **Item Properties** from the **Properties** dialog box.
  - b. Click the **Advanced Properties** icon  .
  - c. Enter a default value in the **Implied Default** field.
8. Perform the following steps if you want to assign multiple values to the item type:
  - a. Select **Restrictions** from the **Item Properties** drop-down list of the **Properties** dialog box.
  - b. Click **Include**.
  - c. Enter the value for the selected item type in the first row.
  - d. **Optional:** Click **Description** to provide the details of the values of the item type.
  - e. Click Menu  , and then click the **Value include restrictions** icon  to add another row.
9. Save the changes.

### Results

You have successfully inserted the values for item types manually.

### What to do next

You must prepare the structural view of the schema. See [Designing a structural view of the schema on page 184](#).

## Setting weightage values

You can set weightage values for an item type to determine how often the value of that item type can be displayed in the generated test data.

### Before you begin

- You must have a project and a schema.
- You must have created a weightage file that can be of CSV or Microsoft Excel Open XML format.



**Note:** The weightage file consists of one column for values of item types and another column for weightage values for each item type value. The weightage values must be integers.

### About this task

The property of weightage values is set as a restriction on an item type. The probability to get higher accuracy increases in the results of the weightage values in the following scenarios:

- When HCL® OneTest™ Data processes a large amount of data.
- When there is a large difference between the weightage values.

### Computation of weightage value

The probability of occurrence of any item type value in the generated test data can be computed by using the following formula:

Probability of occurrence of any item type value = Weight of each item type value / Total weight of all the item type values

The following table shows an example when you have a weightage file with five item type values and its corresponding weightage values:

Item type values	Weightage values
Red	50000
Blue	5000
Green	500
Orange	50
Purple	5

The total weightage value of all the item type values =  $50000 + 5000 + 500 + 50 + 5 = 55555$ .

The probability of occurrence of each item type value in the generated test data can be computed as shown in the following table:

Item type values	Probability of occurrence	Computed values
Red	$50000/55555$	0.9
Blue	$5000/55555$	0.09
Green	$500/55555$	0.009

Item type values	Probability of occurrence	Computed values
Orange	50/55555	0.0009
Purple	5/55555	0.00009

From the computed values, `Red` has a higher probability value than other colors.

When you use this weightage file to set the weightage value for any specific item type and generate the test data, then you can find the generated test data with more occurrences of `Red` as compared to the other colors.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

**Result**

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

**Result**

The **Data Fabrication** page is displayed.

4. Click **Files** and import the weightage file from the local file system into your project.
5. Open your schema from the **Schemas** tab.
6. Select the item type on which you want to apply the weightage value.
7. Select **Restrictions** from the **Item Properties** drop-down list of the **Properties** dialog box.
8. Enable the **Values from file** toggle button.
9. Provide the name of the weightage file in the **Values file** field.
10. Enter the column number in **Values column number** and **Weights column number**.
11. Click **Save**.
12. Select the group and generate the test data.

## Results

You can view more occurrences of the values that are set with higher weightage values as a result of the test data generation.

## What to do next

You must prepare the structural view of the schema. See [Designing a structural view of the schema on page 184](#).

---

### Related information

[Value assignation for item types on page 173](#)

## Setting regular expressions

When you want values of an item type to appear in a certain pattern in the generated test data, you must set the **Regular Expression** property for that item type in a defined schema.

## Before you begin

- You must have logged in to HCL OneTest™ Server.
- You must have created a project and a schema.

## About this task

The regular expression is a special text to describe a pattern. For example, you can specify regular expressions for email address, phone number, or website. HCL® OneTest™ Data provides you some sample regular expressions.

To set regular expressions to any item type, you can either insert a value or select from the sample list of regular expressions that best suits your testing requirements. HCL® OneTest™ Data then validates the value you inserted for the **Regular Expression** field and generates the values of the selected item type in a similar pattern.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

### Result

The **Data Fabrication** page is displayed.

4. Open your schema from the **Schemas** tab.
5. Select the item type for which you want to set regular expressions.
6. Click **Regular Expression** from the **Properties** dialog box.

The sample list of regular expressions is displayed.

7. Set the regular expression in the **Regular Expressions** field by using any one of the following methods:
  - Type the regular expression. HCL® OneTest™ Data then validates the regular expression.

For example, `[0-9]{18}`



**Tip:** If you want to reuse the validated regular expression then click **Add to sample list**, to add the expression in the list.

- Select the regular expression from the sample list. The selected regular expression is displayed in the **Regular Expressions** field.
8. Click **Save**.

## Results

You have successfully set the regular expression for the selected item type.

## What to do next

You must prepare the structural view of the schema. See [Designing a structural view of the schema on page 184](#).

---

Related reference

[Regular expressions on page 657](#)

## Integrating sample files into a schema

When you want to generate the test data, you must define a schema. HCL® OneTest™ Data provides sample files with item type values so that you can integrate the sample file into a schema by mapping it to any item type.

### Before you begin

- You must have logged in to HCL OneTest™ Server.
- You must have created a project and a schema.

### About this task

The **Files** tab on the **Data Fabrication** page is one of the design components of a project. From the **Files** tab, you can integrate a file into a schema by using any one of the following options:

- Create a file by retrieving the data.
- Upload a file from the local file system.
- Use any sample file.

Sample files contain the values for a specific item type. The sample files are in CSV and Excel file formats. You can map these sample files to an item type in a schema.



**Note:** You can map only one sample file to one item type in a schema.

You can use the **Search** field to search for any specific sample file from the **Sample Files** list. After you select a sample file, you can use the menu to manage the file for the following actions:

- Edit: You can modify the file name and the description of the selected sample file.
- Delete: You can delete the selected sample file from the list of files you can view in the **File** tab.
- Download: You can download the selected sample file into your local file system.



**Note:** If you want to modify the selected sample file, you must first download the file into your local file system, and then edit the file for changes.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

### Result

The **Data Fabrication** page is displayed.

4. Click **Use sample** from the **Files** tab.

The **Sample Files** dialog box is displayed.

5. Select one or more sample files that you want to import into your project, and then click **Select**.
6. Click the **Files** tab, and then download the selected sample file that you imported into your project.



**Note:** You can open the sample file and check the column number of the file that you want to map with the item type.

7. Open your schema from the **Schemas** tab.
8. Select **Restrictions** from the **Item Properties** drop-down list of the **Properties** dialog box.
9. Enable the **Values from file** toggle button.
10. Provide the file name of the selected sample file with an extension in the **Values file** field.
11. Enter the column number of the sample file where the item type values are specified in the **Values column number** field.
12. Click **Save**.

### Results

- You can view the list of all the sample files you have selected in the **Files** tab.
- You have successfully integrated the sample file into a defined schema.

### What to do next

You must prepare the structural view of the schema. See [Designing a structural view of the schema on page 184](#).

## Adding JavaScript rules

When you have a JavaScript file with defined rules to create values for any item type, then you can add the JavaScript rules to create the values for the mapped item type.

### Before you begin

- You must have created a project and a schema.
- You must have created a JavaScript file with defined rules in your local file system.

### About this task

If you want to add the JavaScript rules to create the values for the mapped item type, then you must use the following command to define the rules in the JavaScript file:

```
function getData()
```

You can additionally set any one of the following properties for the mapped item type by using the JavaScript rules:



- Regular expression: The JavaScript rule uses the regular expression that you enter in the **Properties** dialog box as an input for the regular expression rule.
- Seed value: The JavaScript rule uses the seed value that you enter in the **Generate Data** dialog box as an input for the seed value rule.

To set either regular expression or seed value property, you must define the rules in the JavaScript file by using the following commands:

Property	Command	Parameter description
Regular expressions	function getData( <b>param</b> )	( <b>param</b> ): Represents the regular expression pattern in the <b>Regular Expression</b> field.
Seed value	function getData( <b>param</b> )	( <b>param</b> ): Represents the seed value in the <b>Numeric Seed Value</b> field of the <b>Generate Data</b> dialog box.



**Note:** If you set the values for both regular expression and seed value properties in HCL® OneTest™ Data and define the rule in the JavaScript file by using the command function getData(**param**), then only the value set for regular expression property is accepted while creating the item type values.

After you define rules in the JavaScript files, you can import the JavaScript file from your local file system into your project, and then map the file to any item type in a schema.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.


**Result**

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

**Result**

The **Data Fabrication** page is displayed.

4. Click **Files** and import the JavaScript file from the local file system into your project.
5. Open your schema from the **Schemas** tab.
6. Select the item type for which you want to create the values by using the JavaScript file.
7. Select **Item Properties** from the drop-down list of the **Properties** dialog box.
8. Select **Text** in the **Item Subclass** field.
9. Click the **Advanced Properties** icon .
10. Optional. Provide the pattern in the **Regular Expression** field, if you want to set the regular expression for the mapped item type.
11. Provide the name of the JavaScript file with an extension in the **JavaScript Name** field.



**Important:** You must enter the correct file name of the JavaScript file that you imported into your project. For example, `address.js`.

12. Click **Save**.

## Results

You have successfully added the JavaScript rules for the item type.

## What to do next

You must prepare the structural view of the schema. See [Designing a structural view of the schema on page 184](#).

## The pairwise data generation method

When you test any application, you might want to ensure that the test data that you use covers all possible combinations of item type values. To achieve this goal of maximum coverage of all possible scenarios, you must apply the pairwise data generation method on the required item types of a schema.

For testing any application, you must attain the maximum test coverage. You can achieve the maximum test coverage and discover more defects with less effort and time by applying the pairwise data generation method. The pairwise data generation method in HCL® OneTest™ Data helps to generate all the possible combinations of the item type values for each pair of item types. The use of this method decreases the number of the generated test data but covers all the possible combinations of the item type values.

For example, consider that there are three item types in a schema such as a list box with five values (zero, one, two, three, four), one radio button with two values (selected or unselected), and one check box with two values (selected or cleared). Therefore, the combination becomes  $5 \times 2 \times 2$  as the generated test data, which is equal to 20 values. These 20 values are required to cover all combinations of the item types of the schema without applying the pairwise data generation method. After you apply the pairwise data generation method, you can observe that the generated test data comprises all combinations of item type values for the three pairs of item types among the three item types. This pairing of item types results in 12 combinations of values of the generated test data.

For more information about how to set the pairwise data generation method in HCL® OneTest™ Data, see [Setting the pairwise data property on page 182](#).

## Setting the pairwise data property

When you want to generate an optimum test data that covers all possible combinations of item type values for each pair of item types in a schema, you can set the pairwise data property for those item types.

### Before you begin

You must have completed the following tasks:

- Created a schema in HCL® OneTest™ Data.
- Read the details about how to assign the values to an item type. See [Value assignment for item types on page 173](#).

### About this task

The pairwise data generation method is a combinational method to find all the possible values of the selected pairs of item types. The pairwise data generation method is efficient when you have a large number of item types and the item type values in a schema. When you apply the pairwise data generation method in such cases, the number of generated test data decreases drastically although covering all the combinations. To implement the pairwise data generation method, you must have multiple item types in a group type of a schema. To assign item type values, you must either import the values of item type from a file or insert values for each item type.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.


3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Open your schema from the **Schemas** tab.
5. Select the item type from **Dictionary** for which you want to set the pairwise data property.
6. Select **Restrictions** from the **Item Properties** drop-down list of the **Properties** dialog box.
7. Set values for the selected item type by using any one of the following methods:

#### Choose from:

- Import item type values from the file by using **Values from file**.
  - Include values for the selected item type in the **Properties** dialog box.
8. Save the changes.
  9. Select **Item Properties** from the drop-down list of the **Properties** dialog box.
  10. Click the **Advanced Properties** icon .
  11. Select **Yes** from the **Pairwise Data** drop-down list.



**Note:** The default value of the **Pairwise Data** drop-down list is **No**.

12. Save the changes.



**Important:** You must select multiple item types to apply the pairwise data generation method.



**Note:** When you want to apply the pairwise data generation method for each item type, you must repeat step 4 through step 10.

## Results

You have successfully set the pairwise data property for the selected item type.

## What to do next

You must select a group type to generate the test data of a schema. See [Generating test data on page 214](#).



**Note:** The group type that you select to generate the test data can comprise both the item types set for the pairwise data property and the item types that are not set for this property.



**Important:** When you generate the test data for a group type with a few or all of the item types set for the pairwise data property, then during the test data generation, the value you enter for the **Number of records** field is ignored. Based on the pairwise data property, HCL® OneTest™ Data auto calculates the number of records.

---

### Related information

[Value assignment for item types on page 173](#)

## Designing a structural view of a schema

To define how the item and group types are organized and are related to each other, you must design the structural view of a schema before generating the test data.

### Before you begin

You must have a group type in your schema.

### About this task

When you design the structural view of a massive schema, you can filter the item or group types of your schema by using **Filter**.

1. Double-click any group type from **Dictionary**.

Alternatively, you can click the menu of the selected group, and then select **Structure**.

**Result**

The **Structure** dialog box appears.

2. Drag and drop the types from **Dictionary** to the **Structure** dialog box to map **Item** and **Group** types.



**Note:** You can use the **Category** type to organize the data dictionary entities. You cannot map the **Category** type in the structure view.

3. Save the changes.

**What to do next**

You can set restrictions and apply functions to define the component rules to any item type. You can also generate the test data without applying restrictions and functions. See [Generating test data on page 214](#).

## Setting a range

After you define the structure of a schema, you can set a number range to specify the number of occurrences to generate an item type in the test data.

**Before you begin**

- You must have a group type in your schema.
- You must have a structural view of your schema.

**About this task**

You can set the number range only for item types and group types of your schema.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

**Result**

The **Overview** page is displayed.


3. Click **Author > Data Fabrication**.

**Result**


The **Data Fabrication** page is displayed.

4. Select the generated schema from the **Schemas** tab.

For example, *schema\_<number>*

5. Click the **Menu** icon  of the group type of the schema and select **Structure**.


The structural view of the schema is displayed.

6. Click the **Menu** icon  of all item types in the **Structure** dialog box to set the restriction for several occurrences of the item types in the generated test data.

You can set the following restriction rules:

- **Set Optional**
- **Set Required**
- **Set Range**
- **Set Unlimited**

In the following table, you can find the value of the minimum and maximum occurrences that you can set as restrictions for the item types:

Item properties	Minimum occurrences	Maximum occurrences
Optional	0	1 or more
Required	1	1
Unlimited	1	1 or more
Set Range	Any minimum value   <b>Note:</b> Minimum value should be equal to or less than the maximum value	Any maximum value

7. Save the changes.

You can modify the properties of the selected item type by using **Properties** in the context menu.

## Results

You have set the number range for the selected item types.

## What to do next

You can apply functions to define component rules to any item type. You can also generate test data without applying functions. See [Generating test data on page 214](#).

---

Related information

[Restrictions settings on page 694](#)

## Applying functions

You can apply functions on the item types of a schema to define the component rules.

## Before you begin

You must have a defined schema in a schema designer. See [Schema design overview on page 170](#).

## About this task

You can also apply functions on the types of the imported JSON or XSD schemas.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

### Result

The **Data Fabrication** page is displayed.

4. Click **Schemas**, and then select the schema for which you want to generate the test data.
5. Select the group type, and then open the structural view of the schema.
6. Select the item type on which you want to apply the function, in the **Structure** dialog box.
7. Click the menu on the right side of the **Structure** dialog box.

### Result

A blank field is displayed next to the item type.

8. Click the blank field.


### Result

The **Rule Editor** pane is displayed on the page.

9. Enter the equal sign (=) in the **Rule Editor** pane.
10. Provide the function in the **Rule Editor** pane.

You can enter the function by performing any one of the following methods:

### Choose from:

- Enter the function manually.
- Select from the list of functions by using the following steps:
  - a. Click the **Insert mapping functions** icon .
  - b. Select the function that you want to apply as a component rule, and then click **Insert** from the menu of the selected function.

For example, if you want the generated value of an item type in uppercase, then you must enter or select the `UPPERCASE ( )` function.

```
=UPPERCASE ( RANDDATA ( ) )
```

11. **Optional:** Enter a parameter for the selected function.

You can provide the parameter value either as a constant value or as a reference of the item type.

You must use the following syntax to provide the parameter in any function:

```
<item type name>:Out1
```

where:

- *item type name* - Represents the name of the item type.
- :Out1 - Represents a standard output argument.

For example, consider that you have a group type as `Flowers` with `Colors` and `Pattern` as two item types. If you want to generate the value of `Colors` item type in uppercase, then you can select the item type `Pattern` in the **Structure** dialog box. You must open the **Rule Editor** pane, and then select the `UPPERCASE()` function with reference of `Colors` item type as the parameter.

```
= Uppercase (Colors:Out1)
```

12. Press the **Enter** key.

#### Result

The selected function is displayed in the item type field of the **Structure** dialog box.

13. Save the changes.

## Results

You have successfully applied the function on an item type of the schema.

## What to do next

You can generate the test data for the designed schema. See [Generating test data on page 214](#).

### Related information

[Functions and expressions on page 733](#)

[Designing a structural view of a schema on page 184](#)

[Setting a function to concatenate the values of item types on page 188](#)

## Setting a function to concatenate the values of item types

You can set a function to concatenate the values of multiple item types so that the values are displayed as the value of another item type.

### Before you begin

You must have completed the following tasks:

- Defined a schema. See [Schema design overview on page 170](#).
- Created the structural view of the schema. [Designing a structural view of a schema on page 184](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.



**Result**

The **Overview** page is displayed.

- Click **Author > Data Fabrication**.

**Result**

The **Data Fabrication** page is displayed.

- Click **Schemas**, and then select the schema for which you want to generate the test data.
- Open the structural view of the schema.



**Important:** You must have all the item types in the same group type for the values that you want to concatenate.

**Result**

The **Structure** dialog box is displayed.

- Select the item type for which you want the values of the generated test data in the concatenated format.



**Note:** You must not place the item type, for which you want the values in the concatenated format, in the same group type.

For example, consider that a schema includes a group type, `group_name1` and three item types such as `item type1`, `item type2`, and `item type3`. If you want to concatenate the values of `item type1` and `item type2` as the generated value of `item type3`, then you must add `item type1` and `item type2` item types in the `group_name1` group type. You must ensure that `item type3` does not exist in the `group_name1` group type.

- Click the menu on the right side of the **Structure** dialog box.

**Result**

A blank field is displayed next to all the item types.

- Click the blank field of the item type for which you want to set the function for concatenation.

**Result**

The **Rule Editor** pane is displayed on the page.

- Enter the equal sign (=) in the **Rule Editor** menu.
- Select the `SERIESTOTEXT()` function from the **Insert mapping functions**  icon.



**Note:** You can also provide the following rule manually in the **Rule Editor** pane: `"item type1" + "item type2"`.

- Provide the name of the group type as the parameter for this function.

For example, if you have a `group_name1` group type, then you must provide the following function:

```
SERIESTOTEXT(group_name1:OUT1)
```

- Press Enter.

**Result**

The selected function is displayed in the item type field of the **Structure** dialog box.

13. Save the changes.

## Results

You have successfully set the function to concatenate the values of multiple item types.

## What to do next

You can generate the test data for the designed schema. See [Generating test data on page 214](#).

---

Related information

[Functions and expressions on page 733](#)

## Test data generation from structured schemas

When you want to test your application by using randomly generated test data, then you must generate the test data from the structured schemas.

You can generate the test data for schemas that you import, create, or generate in HCL® OneTest™ Data by using any external resource. You can generate the test data simultaneously for multiple schemas that are related to each other through common item types. The item types of the child schema refer to the item types of the parent schema. After you view the relationship among the selected schemas, you can generate the test data. When you generate the test data by using the related schemas, the generated values of referring item types of the child schema are available in the generated values of the referred item types of the parent schema, based on referential integrity.

You can also establish or delete parent and child relationships among multiple non-JDBC schemas, based on referential integrity.



### Notes:

- When you generate the test data by using multiple schemas, you must select either JDBC-supported or non-JDBC schemas.
- If you generate the test data for JDBC-supported schemas, then all the selected schemas must be generated from the same JDBC connection.

When you generate the test data for JDBC-supported schemas, then you can either download the generated test data or can insert it into the connected database. You can insert the test data into the following databases if there is a successful connection between HCL® OneTest™ Data and the database:

- JDBC-supported databases
- MongoDB

## HCL OneTest Data support to write the generated test data in JDBC-supported databases

When you want the generated test data in any Java Database Connectivity (JDBC) supported database, you can generate the test data by using HCL® OneTest™ Data. Subsequently, HCL OneTest Data inserts the generated test data directly in the configured JDBC-supported database.

To enable HCL® OneTest™ Data to write the generated test data in the database, you must create a sample schema with tables in the installed JDBC-supported database, establish a connection with the installed database, select the generated schema in HCL® OneTest™ Data, and then generate the test data of the selected schema.

HCL® OneTest™ Data supports the following JDBC-supported databases:

- MySQL Server
- Microsoft SQL Server
- IBM DB2

HCL® OneTest™ Data supports the test data generation for the following data types:

- INTEGER
- VARCHAR
- CHAR
- DATETIME
- DATE
- TIME
- BOOLEAN



**Note:** You must add the restriction values for BOOLEAN data type as 0 and 1.

### Prerequisites

The following prerequisites must be met to enable HCL OneTest Data to write the generated test data:

- You must have access to HCL OneTest™ Server.
- You must have installed any JDBC-supported database, such as MySQL, Microsoft SQL Server, and so on.

### Task flow

You must perform the following tasks in sequence to enable HCL OneTest Data to write the generated test data directly in the JDBC-supported database. The table also provides you the links to the information about the tasks.

Task	More Information
Create a sample schema in a database	<a href="#">Creating a sample schema in a database on page 192</a>

Task	More Information
Establish a JDBC connection with HCL OneTest Data	<a href="#">Establishing a JDBC connection with HCL OneTest Data on page 194</a>
Generate a schema in HCL OneTest Data	<a href="#">Generating a schema in HCL OneTest Data on page 195</a>
Insert the generated test data in the database	<a href="#">Inserting the generated test data in a database on page 196</a>

## Creating a sample schema in a database

To generate a schema in HCL® OneTest™ Data, you must create a schema with a table in the database.

### About this task

After you download the connector jar files of the JDBC-supported database, you must copy the jar files from your computer to the pods such as *HIP REST* and *HIP Server*. These pods are responsible for the following actions:

- *HIP Server* is responsible to establish a connection with the JDBC-supported database. This pod also generates a schema by using the tables of the database. The structure of a schema is a flat hierarchy, where all the item types are in a single group called a *Row*. The item types in a schema refer to the columns of the table.
- *HIP REST* is responsible to run the map and generate the data. This pod also writes the generated data to the JDBC-supported database.

Both the pods use the connector jar file to communicate with the JDBC-supported database.

You can download the connector jar files from the following locations:

Connector jar	Location
Mysql Server connector jar	<a href="https://dev.mysql.com/downloads/connector/j/">https://dev.mysql.com/downloads/connector/j/</a>
Microsoft SQL Server connector jar	<a href="https://docs.microsoft.com/en-us/sql/connect/jdbc/download-microsoft-jdbc-driver-for-sql-server?view=sql-server-ver15">https://docs.microsoft.com/en-us/sql/connect/jdbc/download-microsoft-jdbc-driver-for-sql-server?view=sql-server-ver15</a>
IBM DB2	<a href="https://www.ibm.com/support/pages/db2-jdbc-driver-versions-and-downloads">https://www.ibm.com/support/pages/db2-jdbc-driver-versions-and-downloads</a>

1. Download the JDBC-supported database connector jar file to your local file system.



**Note:** You must download the connector jar file that is compatible with the database server.

2. Copy the connector jar file from your local file system to the pods by using the following commands from the command prompt.

For example, run the following commands if you are using the MySQL database:

For *HIP REST*

```
kubectl cp /<source_path>/mysql-connector-java-8.0.19.jar {my-ots}-<hip_rest_podname>:/opt/runtime/
extjar/mysql-connector-java-8.0.19.jar -n test-system
```

For *HIP Server*

```
kubectl cp /<source_path>/mysql-connector-java-8.0.19.jar {my-ots}-<hip_server_podname>:/opt/hcl/hip/
extjar/mysql-connector-java-8.0.19.jar -n test-system
```



**Notes:**

- Depending on the database you use, you must copy different connector jar files to the pods.
- While copying the jar files to the pods, you must ensure that the names of pods are same as those installed on HCL OneTest™ Server.

For example:

- *HIP Rest* is displayed as `{my-ots}-hip-rest-6757c99d9-4qjtl`
- *HIP Server* is displayed as `{my-ots}-hip-server-0`
- When you install or restore HCL® OneTest™ Data, you must copy the connector jars files again.

3. Create a schema with a table in the database by using the following commands from the command prompt.

For example, you can use the following command to create a table in a schema:

```
create schema <schema_name>;
use <schema_name>;
CREATE TABLE IF NOT EXISTS <table_name> (
  Column1 <DataType> Constraint,
  Column2 <DataType> Constraint,
  Column2 <DataType> Constraint,,,,,
);
```



**Remember:** In HCL® OneTest™ Data, you must refer to the schema name and the table name that you created by using this command.

## Results

You have created a sample schema with a table in the database.

## What to do next

After you create a schema in the database, you must establish the connection between HCL® OneTest™ Data and the JDBC-supported database. See [Establishing a JDBC connection with HCL OneTest Data on page 194](#).

You can view the schema name that you created in the database as the value of the **Catalog** or **Schema** property in HCL® OneTest™ Data.

## Establishing a JDBC connection with HCL OneTest Data

You must establish a connection between HCL® OneTest™ Data and the JDBC supported database to generate the schema and to write the generated test data in the database.

### Before you begin

You must have created a sample schema in the database. See [Creating a sample schema in a database on page 192](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click **New** from the **Connections** tab to create a new connection.
5. Select *JDBC* as a connection type and click **Next**.
6. Enter the following properties of the connection in the **Properties** dialog box.

- a. Enter the URL of the database.

You must provide the URL in the following format: `jdbc:<DatabaseType>://<HostName/IPAddress>:<PortNumber>/<CustomSettings>`



**Note:** You must provide the database name in `<CustomSettings>` for establishing connection with IBM DB2 database.

For example,

- MySQL Server - `jdbc:mysql://10.0.2.15:8081`
- Microsoft SQL Server - `jdbc:sqlserver://10.134.59.16:1433`
- IBM DB2 - `jdbc:db2://10.134.59.16:50000/testdb`

- b. Enter the name and password to access the database.

7. Click **Test** to verify whether the connection is established successfully.



**Note:** If the connection fails to establish, it can be because of one of the following reasons:

- Mismatch in the version of the drivers.
- Unreachable database.
- Missing database.
- Unsuccessful copying of the connector jar file.

You can run the following command to check the logs of *HIP Server* to get more information about the reason of failure:

```
kubectl logs {my-ots}-hip-server-0 -n test-system
```

### Results

You have successfully established the connection between HCL® OneTest™ Data and the database.

### What to do next

You must provide a name for the connection, and then generate a schema. See [Generating a schema in HCL OneTest Data on page 195](#).

## Generating a schema in HCL OneTest Data

After you establish the connection between HCL® OneTest™ Data and the JDBC-supported database, you can generate a schema in HCL® OneTest™ Data from the configured database.

1. Click the **Connections** tab and select the connection that you created.
2. Click the context menu and select **New Action**.
3. Select **Target** as the **Action Type** in the **New Action** dialog box and click **Next**.
4. Enter the following properties of the schema that you want to generate, and then click **Next**.

Properties	Actions	Required/Optional
Target	Select the target as <i>Table</i> that you created in the database from the list.	Required
Catalog	Select the schema that you created in the database from the list.	Required
Schema	Select the schema that you created in the database from the list.	Optional

Properties	Actions	Required/Optional
	The name of schema is displayed either in the <b>Catalog</b> or in the <b>Schema</b> field for the selected database.	
Table	Select the table that you created in the database from the list.	Required



**Note:** You can click the **Advanced Properties** icon  to set the additional properties of the schema.

5. Click **Generate**.

The schema is generated in HCL® OneTest™ Data by using the schema you created in the database. HCL® OneTest™ Data generates the schema name as *schema\_<number>*.



**Note:** You must select this schema name as the schema for HCL® OneTest™ Data to generate the test data.

6. Select the schema type as **Row** and click **Next**.

7. Provide the identification details of the new action, and then click **OK**.

The new action is created and is listed under the JDBC connection.

## Results

You have successfully generated the schema in HCL® OneTest™ Data.



**Important:** If you want to generate schemas for other tables in the same database, then instead of creating a new connection, you must click **New Action** on the existing JDBC connection.

## What to do next

You can provide the action name under the **Identification** dialog box, and then generate the test data in the database. See [Inserting the generated test data in the database on page 196](#).

## Inserting the generated test data in a database

After you generate the schema from the configured database in HCL® OneTest™ Data, you can directly insert the generated test data into the database.

## Before you begin



You must have generated the schema by using the database. See [Generating a schema in the database on page 195](#).

1. Select the schema that you generated with the following name from the **Schema** tab:

*schema\_<number>*

2. Click the menu of the group type of the schema and select **Structure**.

You can view the list of all item types associated with the selected group type in the structure view.

3. Click the menu for all the item types to set the property as **Set Required**, and then click **Save**.



**Note:** If you have a numeric item type, you must set **Min size (digits)** and **Max size (digits)** properties.

For example, when you select `Integer` as **Presentation** then you can set the **Min size (digits)** as 1 and the **Max size (digits)** as 9.

4. Select the group type in the **Dictionary** dialog box and click **Generate Data** from the menu.

The **Generate Data** dialog box is displayed.

5. Select the **Connection** option and provide values for the **Number of records** and **Numeric Seed Value**.

The seed value acts as an instance of random data that can produce the same set of data multiple times. By default, the seed value is blank.



**Note:** The following table lists the maximum values for numeric seed value and number of records:

Description	Maximum values
Numeric Seed Value	8 digits (99,999,999)
Number of records	9 digits (999,999,999)

6. Select the connection type from the list, and then click **OK**.

## Results

You have successfully inserted the generated test data from HCL® OneTest™ Data in the JDBC supported database.

## What to do next

You can click the **Jobs** tab to view the status of the job. The tooltip of the job displays the following details:

- URL of the connection.
- Name of the table.
- Schema name from which the test data is generated.

 **Important:**

You cannot download the generated test data from the **Jobs** page because the test data is written to the configured database and is not available in HCL® OneTest™ Data.

You can log in to the configured database to view the generated test data.

## HCL OneTest Data support to generate test data for an SAP IDoc or DXOB file

When you want to generate the test data for an SAP Intermediate Document (IDoc) or Data Transfer Object (DXOB) file, you can generate a schema in HCL® OneTest™ Data by using the structured schema of the SAP IDoc or DXOB file. You can then generate the test data by using the generated schema in HCL® OneTest™ Data.

The SAP applications provide file-based interfaces, such as the SAP IDoc and DXOB interfaces. The SAP IDoc interface supports Application Link Enabling (ALE) and Electronic Data Interchange (EDI) file formats, and the SAP DXOB interface supports the DXOB files.

You must perform the following tasks to generate the test data for any SAP IDoc or DXOB file:

- [Schema generation for an SAP IDoc or DXOB file on page 198](#)
- [Generating test data on page 214](#)

---

Related information

[SAP applications on page 625](#)

## Schema generation for an SAP IDoc or DXOB file

When you want to generate the test data for an SAP IDoc or DXOB file, then you must generate a schema for the selected SAP IDoc or DXOB file.

The SAP IDoc or DXOB file is a standard data structure in SAP applications to transfer data between SAP system applications and external resources. Every SAP IDoc or DXOB file has a unique number.

If you have an SAP IDoc or DXOB file that you want to use for test data generation in your local file system, then you must import the file into your HCL® OneTest™ Data project. After importing the file, you must establish a connection between the SAP IDoc or DXOB interface and HCL® OneTest™ Data. The established connection helps to transfer and interpret the data of the imported SAP IDoc or DXOB file. The interpreted information then helps HCL® OneTest™ Data to generate a schema.

You must perform the following tasks to generate a schema in HCL® OneTest™ Data for the SAP IDoc or DXOB file:

## Importing an SAP IDoc or DXOB file

When you want to establish a connection with an SAP IDoc or DXOB interface, you must import the SAP IDoc or DXOB file into your HCL® OneTest™ Data project from your local file system.

### Before you begin

You must have completed the following tasks:

- Created a project.
- Created the SAP IDoc or DXOB file in your local file system.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click **Files**, and then click **New** to import the SAP IDoc or DXOB file from the local file system into your project.

### Results

You have successfully imported the SAP IDoc or DXOB file from the local file system into your project.

### What to do next

You must establish a connection between the SAP IDoc or DXOB interface and HCL® OneTest™ Data. See

[Establishing an SAP IDoc or DXOB interface connection with HCL OneTest Data on page 199](#).

## Establishing an SAP IDoc or DXOB interface connection with HCL OneTest Data

When you want to generate a schema for an SAP IDoc or DXOB file, you must establish a connection with the SAP IDoc or DXOB interface.

### Before you begin

You must have imported the SAP IDoc or DXOB file into your project.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click **Connections**.

5. Click **New** to create a new connection type.

The **New Connection** dialog box is displayed.

6. Select **SAP IDoc** from the **Type** drop-down list, and then click **Next**.
7. Set the following properties under **Connection Properties**:
  - Enable **Bypass Client Connection**.
  - Select 2 (Unicode) as **Bytes Per Character**.
8. Click **Next**, and then provide a name to the established connection.
9. Click **OK**.

You can see the name of the SAP IDoc connection under the **Connections** tab.

## Results

You have successfully established the connection between HCL® OneTest™ Data and the SAP IDoc or DXOB interface.

## What to do next

You must generate a schema in HCL® OneTest™ Data by using the SAP IDoc or DXOB file. See [Generating schema by using an SAP IDoc or DXOB file. on page 200](#)

## Generating schema by using an SAP IDoc or DXOB file

When you want to generate the test data for an SAP IDoc or DXOB file, then you must generate a schema for the selected SAP IDoc or DXOB file.

## Before you begin

You must have established the connection with the SAP IDoc or DXOB interface.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

### Result

The **Data Fabrication** page is displayed.

4. Click **Connections**.
5. Select the SAP IDoc connection name that you established, and then click **New Action** from the menu.

The **New Action** dialog box is displayed.

6. Select **Target** as the action type, and then click **Next**.
7. Enter the following properties for the new action:

- a. Select one of the following file types from the **Data record type** drop-down list.
    - ALE - 1000 byte segments (ALE/RFC port)
    - EDI - Segments terminated by line feed (file port)
    - DXOB - Data Transfer Object
  - b. Select the imported SAP IDoc or SAP DXOB file name from the **IDoc or DXOB File** drop-down list.
  - c. Select **Native** from the **Character set** drop-down list.
8. Click **Next**.
  9. Click **Generate** to generate a new schema by using the imported SAP IDoc or SAP DXOB file.

A schema is generated as `anonymous_schema_<number>`.

10. Select **Schema type**.
11. Click **Next**.
12. Provide the name to the new action, and then click **OK**.

The new action is created and listed under the SAP IDoc or DXOB connection name that you established.



**Note:** You can create multiple actions for each established connection.

13. Click **Save**.

### Results

You have successfully generated a schema for the SAP IDoc or DXOB file in HCL® OneTest™ Data.



**Note:** You can modify the schema name `anonymous_schema_<number>` from the **Schemas** tab.

### What to do next

You can generate the test data by using the generated schema. See [Generating test data on page 214](#).

## HCL OneTest Data support to generate test data for SAP BAPI

When you want to generate test data for SAP Business Application Programming Interface (BAPI), you can generate a schema in HCL® OneTest™ Data based on structured schema of SAP BAPI function modules. You can then generate the test data by using the generated schema in HCL® OneTest™ Data.

SAP BAPIs are API methods of SAP Business Object Types. HCL® OneTest™ Data uses SAP BAPI to access the application layer of the SAP System.

### Prerequisites

You must have access to HCL OneTest™ Server.

### Task flow

You must perform the following tasks in sequence to generate the test data for SAP BAPI in HCL® OneTest™ Data.

The table also provides you the links to the information about the tasks:

Task	More Information
Install SAP connector files in HCL® OneTest™ Data	<a href="#">Installing the SAP connector files in HCL OneTest Data on page 202</a>
Establish an SAP BAPI connection with HCL® OneTest™ Data	<a href="#">Establishing an SAP BAPI connection with HCL OneTest Data on page 203</a>
Generate a schema in HCL® OneTest™ Data	<a href="#">Generating a schema in HCL OneTest Data on page 204</a>
Generate test data in HCL® OneTest™ Data	<a href="#">Generating test data on page 214</a>

Related information

[SAP applications on page 625](#)

## Installing the SAP connector files in HCL OneTest Data

To generate a schema in HCL® OneTest™ Data, you must install the SAP connector files in HCL® OneTest™ Data.

1. Download the SAP connector files `libsapjco3.so` and `sapjco3.jar` to your local file system.
2. Copy the connector files from your local file system to the *HIP Server* pod by using the following commands from the command prompt:

```
kubectl cp /source_path/libsapjco3.so {my-ots}-hip-server-0:/opt/hcl/hip/libs/libsapjco3
```

```
kubectl cp /source_path/sapjco3.jar {my-ots}-hip-server-0:/opt/hcl/hip/libs/sapjco3.jar
```



**Note:** When you install or restore HCL® OneTest™ Data, you must copy the connector files again.

3. Enter the following command to change the current user as the root user:

```
chown <user>:root libsapjco3.so sapjco3.jar
```

4. Run the following command to modify the access permissions of the connector file:

```
chmod 777 libsapjco3.so sapjco3.jar
```

### Results

You have successfully installed the SAP connector files in HCL® OneTest™ Data.

### What to do next

You must establish the connection between HCL® OneTest™ Data and the SAP BAPI connector. See [Establishing an SAP BAPI connection with HCL OneTest Data on page 203](#).

## Establishing an SAP BAPI connection with HCL OneTest Data

When you want to generate a schema for SAP BAPI, you must establish a connection between HCL® OneTest™ Data and SAP BAPI.

### Before you begin

You must have installed the SAP connector files in HCL® OneTest™ Data. Ensure that you have installed the connector files a few minutes back. See [Installing SAP connector files in HCL OneTest Data on page 202](#)

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click **Connections**.
5. Click **New** to create a new connection type.

The **New Connection** dialog box is displayed.

6. Select **SAP BAPI** from the **Type** drop-down list, and then click **Next**.
7. Set the following properties under **Connection Properties**, and then save the changes:
  - Enter the SAP System host name in the **Host ID** field.
  - Enter **Client Number** and **System ID** of the SAP System.
  - Provide the SAP System credentials in the **User** and **Password** fields for authentication.
8. Go to the **Connection Properties** pane of the **Data Fabrication** page, and then click **Test** to verify whether the connection is established successfully.



**Note:** Your connection might fail to establish because of any one of the following reasons:

- Trouble at the SAP server port.
  - Firewall blockage between HCL OneTest™ Server and the SAP server.
  - Unsuccessful copying of the connector files.
  - Unsuccessful refresh of the *HIP Server* pod.
9. Click **Next**, and then provide a name for the SAP BAPI connection that you established.
  10. Click **OK**.

You can view the name of the SAP BAPI connection under the **Connections** tab.

### Results

You have successfully established the connection between HCL® OneTest™ Data and SAP BAPI.

### What to do next

You must generate a schema in HCL® OneTest™ Data by using the SAP BAPI. See [Generating a schema in HCL OneTest Data on page 204](#).

## Generating a schema in HCL OneTest Data

After you establish the connection between HCL® OneTest™ Data and an SAP BAPI, you can generate a schema in HCL® OneTest™ Data.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click **Connections**.
5. Select the SAP BAPI connection name that you established, and then click **New Action** from the menu.

The **New Action** dialog box is displayed.

6. Select **Target** as the action type, and then click **Next**.
7. Enter the following properties for the new action:
  - a. Click **Fetch** to select the function module category from the **BAPI Function Module category** drop-down list.
  - b. Click **Fetch** to select the function module of the selected function module category from the **BAPI Function Module** drop-down list.
8. Click **Next**.
9. Select **New**, and then click **Generate** to generate a new schema in HCL® OneTest™ Data.

The schema with a name as *schema\_<number>* is generated in HCL® OneTest™ Data.

The **Schema Type** is populated automatically based on the selected BAPI function module.



**Note:** You must select this schema name as the schema for HCL® OneTest™ Data to generate the test data.

10. Click **Next**, and then provide the identification details of the new action.
11. Click **OK**.

The new action is created and is listed under the SAP BAPI connection.

### Results



You have successfully generated a schema for SAP BAPI in HCL® OneTest™ Data.



**Note:** You can modify the schema name `schema_<number>` from the **Schemas** tab.

### What to do next

You can generate the test data by using the generated schema. See [Generating test data on page 214](#).

## HCL OneTest Data support to write the generated test data in MongoDB

When you use MongoDB as a database in your application testing environment and you want the generated test data in MongoDB, then you can generate the test data by using HCL® OneTest™ Data. Subsequently, HCL® OneTest™ Data inserts the generated test data directly in MongoDB.

To enable HCL® OneTest™ Data to write the generated test data in the database, you must either import or create a schema in HCL® OneTest™ Data, and then establish a connection with MongoDB. After you establish the connection, you must associate the selected schema with MongoDB, and then generate the test data by using HCL® OneTest™ Data. Subsequently, HCL® OneTest™ Data writes the generated test data directly in MongoDB.



**Important:** HCL® OneTest™ Data always writes the generated test data in MongoDB in the JSON format.

### Prerequisites

You must have access to HCL OneTest™ Server.

### Task flow

You must perform the following tasks in sequence to enable HCL® OneTest™ Data to write the generated test data directly in MongoDB. The table also provides you the links to the information about the tasks.

Task	More information
Create a schema in HCL® OneTest™ Data	<a href="#">Create a schema in HCL OneTest Data on page 165</a>
Import a JSON or XSD into HCL® OneTest™ Data project	<a href="#">Import a JSON or XSD into HCL OneTest Data project on page 162</a>
Establish a MongoDB connection with HCL® OneTest™ Data	<a href="#">Establishing a MongoDB connection with HCL OneTest Data on page 206</a>
Associate a schema with MongoDB	<a href="#">Associating a schema with MongoDB on page 207</a>
Insert the generated test data in a database	<a href="#">Inserting the generated test data in a database on page 208</a>

## Establishing a MongoDB connection with HCL OneTest Data

You must establish a connection between HCL® OneTest™ Data and MongoDB to associate a schema with the database.

### Before you begin

You must have completed the following tasks:

- Created or imported a schema in the HCL® OneTest™ Data project.
- Installed MongoDB on your computer and the MongoDB instance must be ready to connect. For more information, refer to [Install MongoDB](#).
- Created a database in MongoDB. For more information, refer to [Database Methods](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click **Connections**.
5. Click **New** to create a new connection.

The **New Connection** dialog box is displayed.

6. Select *MongoDB* as a connection type and click **Next**.
7. Enter the following properties of the connection in the **Properties** dialog box:
  - a. Enter the host name of MongoDB in the **Host** field.
  - b. Enter the server port number of the computer where MongoDB is running in the **Port** field.
  - c. Enter the user name and password to access MongoDB.
8. Click **Test** to verify whether the connection is established successfully.



**Note:** If the connection fails to establish, it can be because of any one of the following reasons:

- Unreachable database.
- Missing database.
- Incorrect host name and port number.

9. Click **Fetch** to upload all the databases available in the configured MongoDB in the **Database** drop-down list, and then select the database that you want to use to write the generated test data.
10. Click **Next**, and then provide a name to the established MongoDB connection.
11. Click **OK**.

You can see the name of the MongoDB connection under the **Connections** tab.

**Results**

You have successfully established the connection between HCL® OneTest™ Data and MongoDB.

**What to do next**

You must associate a schema with a database of MongoDB. See [Associating a schema with MongoDB on page 207](#).

**Associating a schema with MongoDB**

After you establish the connection between HCL® OneTest™ Data and MongoDB, you must associate a schema with a database of MongoDB. You must select the schema that you want to use to generate the test data.

**Before you begin**

You must have created a collection in the selected MongoDB database.

For more information, refer to `db.createCollection()` section of [Database Methods](#).

**About this task**

You can associate any schema with the MongoDB database that can generate the test data in the JSON format.

The following schema formats support the output format in JSON:

- Excel
- JDBC
- XSD
- CSV

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

**Result**

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

**Result**

The **Data Fabrication** page is displayed.

4. Click the **Connections** tab and select the MongoDB connection name that you created.
5. Click the menu and select **New Action**.
6. Select **Target** as the **Action Type** in the **New Action** dialog box, and then click **Next**.
7. Click **Fetch** to extract a list of all the collections of the selected database, and then select a collection from the **Collection** drop-down list.



**Note:**



You can retain the default property settings for all other boxes.

8. Click **Next** to provide the schema details.
9. Click the **Schema** drop-down list to select the schema that you want to use to generate the test data.
10. Click the **Schema Type** drop-down list to select the group type for which you want to generate the test data, and then click **Next**.
11. Provide the identification details of the new action, and then click **OK**.

The new action is created and listed under the MongoDB connection.

## Results

You have successfully associated the schema with the database.



**Tip:** If you want to associate any other schema in the same database, then you must click **New Action** on the existing MongoDB connection. You need not create a new connection.

## What to do next

You can generate the test data from the schema that you associated with the database of MongoDB. The generated test data is directly inserted into the database. See [Inserting the generated test data in a database on page 208](#).

## Inserting the generated test data in a database

After you associate a schema with a database of MongoDB, you can generate the test data and insert the generated test data directly in the database by using HCL® OneTest™ Data.

### Before you begin

You must have associated the schema with MongoDB. See [Associating a schema with MongoDB on page 207](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Select the schema that you associated with the database from the **Schemas** tab.
5. Select the group type of the schema for which you want to generate the test data.
6. Click the menu of the selected group type, and then click **Generate Data**.

You can view the list of all item types associated with the selected group type in the structure view.

#### Result

The **Generate Data** dialog box is displayed.

7. Select the **Connection** option.
8. Select the **MongoDB** connection name that you established from the **Connections** drop-down list.



**Note:**

When you select MongoDB to insert the generated test data, you can generate only one record of test data at a time.

9. Provide a value for **Numeric Seed Value**, and then click **OK**.

## Results

You have successfully generated and inserted the test data in MongoDB.

## What to do next

You can click the **Jobs** tab to view the status of the job. When you hover over the status of the job, you can view the following details in the tooltip:

- Connection URL that includes details of host name and port number.
- Name of the collection.
- Database name with which the schema is associated.



**Important:**

You cannot download the generated test data from the **Jobs** page because the test data is written directly in the MongoDB database and the generated test data is not available in HCL® OneTest™ Data.

You must log in to the database to view the generated test data.

## HCL OneTest Data support to generate test data for an Excel file

When you use Microsoft Excel as a database in your application testing environment and you want to use the structure of an Excel file to generate test data, then you can generate the test data for the selected Excel file by using HCL® OneTest™ Data.

When you want to generate the test data for the Excel file, then you must import the file into your HCL® OneTest™ Data project. After importing the file, you must establish a connection between Excel and HCL® OneTest™ Data. The established connection helps to transfer the data and interpret the data of the imported Excel file into your HCL® OneTest™ Data project. The interpreted information then helps HCL® OneTest™ Data to generate a schema. You can then use the generated schema to generate the test data in HCL® OneTest™ Data.

## Prerequisites

The following prerequisites must be met to enable HCL® OneTest™ Data to write the generated test data:

- You must have access to HCL OneTest™ Server.
- You must have an Excel file on your computer.

### Task flow

You must perform the following tasks in sequence to enable HCL® OneTest™ Data to generate the test data for the Excel file. The table also provides you the links to the information about the tasks.

Task	More Information
Import an Excel file.	<a href="#">Importing an Excel file. on page 210</a>
Establish a connection between Excel and HCL® OneTest™ Data.	<a href="#">Establishing a connection between Excel and HCL OneTest Data on page 211.</a>
Generate a schema based on an Excel file.	<a href="#">Generating a schema based on an Excel file on page 211.</a>
Generate test data by using the generated schema.	<a href="#">Generating test data by using the generated schema on page 213.</a>

## Importing an Excel file

When you want to establish a connection between HCL® OneTest™ Data and an Excel file, you must import the Excel file into your HCL® OneTest™ Data project from your computer.

### Before you begin

You must have completed the following tasks:

- Created a project in HCL OneTest™ Server.
  - Opened an Excel file or created a file if it did not exist.
1. Select your project in HCL OneTest™ Server and navigate to the **Data Fabrication** page.
  2. Click **Files**, and then click **New** to import the Excel file from the computer into your project.

### Results

You have successfully imported the Excel file from the computer into your project.

### What to do next

You must establish a connection between Excel and HCL® OneTest™ Data. See [Establishing a connection between HCL OneTest Data and Excel on page 211.](#)

## Establishing a connection between HCL OneTest Data and Excel

When you want to generate a schema based on an Excel file, you must establish a connection between HCL® OneTest™ Data and Excel.

### Before you begin

You must have imported an Excel file from your computer into your project. See [Importing an Excel file on page 210](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click **Connections**.
5. Click **New** to create a new connection.

The **New Connection** dialog box is displayed.

6. Select **Excel** as a connection type, and then click **Next**.
7. Provide a name for this connection.
8. Click **OK**.

You can see the name of the connection under the **Connections** tab.

### Results

You have successfully established the connection between HCL® OneTest™ Data and Excel.

### What to do next

You must generate a schema in HCL® OneTest™ Data based on the Excel file. See [Generating a schema based on Excel file on page 211](#).

## Generating a schema based on an Excel file

After you establish the connection between HCL® OneTest™ Data and Excel, you can generate a schema that is based on an Excel file in HCL® OneTest™ Data.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click the **Connections** tab and select the connection that you created.
5. Click the menu, and then select **New Action**.
6. Select **Target** as the **Action Type** in the **New Action** dialog box, and then click **Next**.
7. Enter the following properties of the schema that you want to generate, and then click **Next**:
  - Select the Excel file that you imported into your project from the **Excel File** drop-down list.
  - Click **Fetch** to upload all the worksheets of the Excel file in the **Worksheet** drop-down list.
  - Select the worksheet from the **Worksheet** drop-down list.



**Note:**

You can retain the default property settings for all other fields.

8. Select **New**, and then click **Generate**.

The schema is generated in HCL® OneTest™ Data. HCL® OneTest™ Data generates the schema name as *schema\_<number>*.



**Note:** You must select this schema name as the schema for HCL® OneTest™ Data to generate the test data.

9. Select the schema type as **Sheet**.
10. Click **Next** to provide the identification details of the new action, and then click **OK**.

The new action is created and listed under the connection that is established between HCL® OneTest™ Data and Excel.

## Results

You have successfully generated the schema in HCL® OneTest™ Data.



**Note:**

You can modify the schema name *schema\_<number>* from the **Schemas** tab.



**Tip:** If you want to generate schemas for other Excel files, then you must click **New Action** on the existing Excel connection. You need not create a new connection.

## What to do next

You can view the generated schema listed under the **Schemas** tab. You can generate the test data for the generated schema in HCL® OneTest™ Data. See [Generating test data by using the generated schema on page 213](#).



## Generating test data by using the generated schema

After you generate the schema in HCL® OneTest™ Data, you can use this schema to generate the test data in HCL® OneTest™ Data.

### Before you begin

You must have generated the schema by using the connection that is established between HCL® OneTest™ Data and Excel. See [Generating a schema based on Excel file on page 211](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Select the schema name *schema\_<number>* from the **Schemas** tab.

#### Result

The schema is displayed.

5. Select the **Sheet** group type, and then select **Structure** from the menu.

You can view a group as **Row** in the structure of the schema. All item types of a schema are listed under the **Row** group type.

6. Click the menu for the **Row** group type, and then set the property as **Set Required**.
7. Click the menu for all the item types that you want to use to generate the test data, and then set the property as **Set Required**.



**Note:** If you have a numeric item type, you must set **Min size (digits)** and **Max size (digits)** properties.

For example, when you select `Integer` as **Presentation**, then you can set the **Min size (digits)** as 1 and the **Max size (digits)** as 9.

8. Click **Save**.
9. Select the **Sheet** group type in the **Dictionary** dialog box, and then click **Generate Data** from the menu.

The **Generate Data** dialog box is displayed.

10. Provide the number of records that you want to generate in **Number of records**.
11. **Optional:** Provide a seed value that acts as a reference number to generate the same set of data in **Numeric Seed Value**.
12. Select the output file format for the generated test data from the list, and then click **OK**.

### Results

You have successfully generated the test data by using the generated schema.

## What to do next

You can click the **Jobs** tab to view the status of the job. You can then download the generated test data of the jobs that are completed successfully.

## Generating test data

When you have a defined and structured schema for the entire data, you can generate test data to perform application testing.

### Before you begin

The following prerequisites must be met to generate the test data:

- You must have created a project.
- The schema must have a group type.
- You must have a structured view of your defined schema.

### About this task

You can generate the sample test data from the group type of schema. When you import the XSD format schema, then HCL® OneTest™ Data generates only XML test data. If you import JSON format as input, the application generates only JSON test data.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click the **Schemas** tab and select the schema for which you want to generate the test data.
5. Select the group type that you want to use to generate the test data from the **Dictionary** dialog box.
6. Click **Generate Data** from the menu.

#### Result

The **Generate Data** dialog box is displayed.

7. Enter the number of records that you want to generate in the **Number of records** field.



**Important:** When you select the pairwise data generation property for any item type, then the number of records you enter is ignored. Based on the pairwise data generation approach, HCL® OneTest™ Data autocalculates the number of records.

8. Enter seed value in the **Numeric Seed Value** field.

The seed value acts as an instance of random data that can produce the same set of data multiple times. By default, the seed value is blank.



**Note:** The following table lists the maximum values for numeric seed value and number of records:

Description	Maximum values
Numeric Seed Value	8 digits (99,999,999)
Number of records	9 digits (999,999,999)

9. Choose the output file format of the test data that you want to generate from the **Output File Format** list.

HCL® OneTest™ Data supports the following formats:

- CSV
- Native
- JSON
- XML
- Excel



**Notes:**

- If you create a schema by using the schema designer, by default, the output file format is CSV.
- If you import a schema into your project or have a schema with nested groups, you can select only the Plain text file as the output file format.
- You can generate line numbers in the output by creating an auto-increment rule by using functions and expressions.

10. Click **OK**.

### Results

The test data is generated successfully.



**Note:** The column headings for the generated test data are available only for the CSV and Excel file output formats.

### What to do next

You can view and download the generated test data on the **Jobs** page. See [Status of generated test data on page 237](#).

## Generating the test data by using multiple schemas

You can generate the test data either by using multiple JDBC-supported or non-JDBC schemas.

### Before you begin

You must have completed the following tasks:

- Logged into HCL OneTest™ Server.
- Installed any JDBC-supported database, such as Oracle, MySQL, and so on, for generating the test data from JDBC-supported schemas.

### About this task

You can generate the test data for multiple schemas simultaneously. To generate the test data you can either use only JDBC-supported schemas or non-JDBC schemas.

To generate the test data by using multiple JDBC-supported schemas, you must create tables in a database that are supported by the JDBC connection. After you establish the JDBC connection between the database and HCL® OneTest™ Data, you must generate multiple schemas. You can then import the selected schemas into the **Schema Canvas** page. After you import the schemas into the **Schema Canvas** page, you can view an existing parent-child relationship. You must select a single schema or multiple schemas for which you want to generate the test data, and then generate the test data.

To generate the test data by using multiple non-JDBC schemas, you must create a schema in HCL® OneTest™ Data. You must then define the properties and structural view of the schemas. When the schemas are created, you can import the selected schemas into the **Schema Canvas** page. After you import the schemas into the **Schema Canvas** page, you can view an existing parent-child relationship or establish a relationship among the schemas. You must select the schemas, and then generate the test data.



#### Notes:

- When you generate the test data by using multiple schemas, you must select either JDBC-supported or non-JDBC schemas.
- When you select a child schema on the **Schema Canvas** page, the parent schema is automatically selected. Similarly, if you select a parent schema, then all the child schemas are selected on the **Schema Canvas** page.

The output formats that are supported in HCL® OneTest™ Data for the generation of test data for multiple schemas by using referential integrity are CSV, Excel, Native, JSON, and XML. You can also insert the generated test data into the database by using the JDBC connection.



**Note:** You can generate the test data among the common output formats that are supported by all selected schemas.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

**Result**

The **Data Fabrication** page is displayed.

4. Choose any one of the following types of schemas that you want to use to generate the test data:

**Choose from:**

- JDBC

You must perform the following tasks to generate the test data in HCL® OneTest™ Data for multiple JDBC-supported schemas:

- a. [Create a sample schema in a database on page 192.](#)
- b. [Establish a JDBC connection with HCL OneTest Data on page 194.](#)
- c. [Generate a schema in HCL OneTest Data on page 195.](#)
- d. [Define properties of schemas on page 219.](#)
- e. [Import schemas into Schema Canvas on page 220.](#)
- f. [Select schemas on page 225.](#)
- g. [Generate test data for selected schemas on page 215.](#)

- Non-JDBC

You must perform the following tasks to generate the test data in HCL® OneTest™ Data for multiple non-JDBC schemas:

- a. [Create schemas on page 165.](#)
- b. [Design schemas on page 170.](#)
- c. [Import schemas into Schema Canvas on page 220.](#)
- d. Establish a parent-child relationship among schemas.
- e. [Select schemas on page 225.](#)
- f. [Generate test data for selected schemas on page 215.](#)

**Results**

You have generated the test data by using multiple schemas simultaneously.

## Selecting multiple schemas

After you define the properties of schemas, you must select schemas and view the relationships among them to generate the test data for multiple schemas simultaneously.

**Before you begin**

You must have completed the following tasks:

1. Created a project.
2. Generated schemas in HCL® OneTest™ Data by using a JDBC connection. See [Generating a schema in HCL OneTest Data on page 195.](#)
3. Defined properties for the generated schemas. See [Defining properties of schemas on page 219.](#)

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

**Result**

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

**Result**

The **Data Fabrication** page is displayed.

4. Click **Schema Integrity** to select schemas.

**Result**

The **Select Schemas** dialog box is displayed.

5. Click the field to view the list of schemas that are generated by using the JDBC connection.
6. Select multiple schemas.



**Note:** You can select a maximum of 10 schemas.

7. Click **Import** to import the selected schemas into the **Schema Integrity** page.

You can find the structural view of all the selected schemas with their relationships with each other.



**Note:** You can use the scroll wheel or pinch-to-zoom option on the **Schema Integrity** page to view the structural view of all schemas and relationships.

8. Verify the relationships among the selected schemas.
9. Select schemas based on the following criteria:
  - If a schema does not have a relationship with any other schema on the **Schema Integrity** page, then you must select that schema manually. Otherwise, click **Select All** to select all schemas to generate the test data.
  - If a schema has a relationship with other schemas and is selected, then all the related schemas get selected automatically.



**Notes:**

- You cannot select schemas that belong to different JDBC connections.
- You can use **Modify canvas** to add or remove the selected schemas.
- If any of the selected schema is deleted from the workspace, then the view of the **Schema Integrity** page changes as the deleted schema is removed from the **Schema Integrity** page.

## Results

You have successfully selected multiple schemas for the generation of test data.

## What to do next

You can generate the test data for the selected schemas. See [Generating test data for selected schemas on page 228](#).

## Defining properties of schemas

After you generate the schema from the configured database in HCL® OneTest™ Data, you must define the required properties for all schemas.

### Before you begin

You must have completed the following tasks:

1. Created a project.
2. Generated schemas in HCL® OneTest™ Data by using a JDBC connection. See [Generating a schema in HCL OneTest Data on page 195](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.


3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Select the generated schema from the **Schemas** tab.

For example, *schema\_<number>*

5. Click the **Menu** icon  of the group type of the schema and select **Structure**.

You can view the list of all item types associated with the selected group type in the structural view of the schema.

6. Click the **Menu** icon  of all item types in the **Structure** dialog box to set the following properties:

- a. Click **Set Required** to get at least one occurrence of the item type in the generated test data, and then click **Save**.

- b. Click **Properties** for the selected item types and perform the following steps for **Item Properties**:
  - i. Verify the values of the **Item Subclass** and **Presentation** fields.

For example, if the item type is of a *number* data type, then the value for **Item Subclass** is *Number* and **Presentation** is *Integer*.

- ii. Set the **Min size** and **Max size** for the item types of the schema.

If the value of **Subclass** is `Number` and **Presentation** is `Integer` for an item type, then set the value of **Min size** and **Max size** as the number of digits that can be generated in the test data.



**Note:** The value for **Min size (digits)** should be less than or equal to **Max size (digits)**.

For example, if the item type is of a `number` data type, then you can set the **Min size (digits)** as 1 and the **Max size (digits)** as 9.



**Notes:**

- The values of both the **Min size (digits)** and **Max size (digits)** properties for item types that act as a primary and foreign keys must be same.
- You must set the value of **Min size** and **Max size** for an `Integer` item type such that the number of digits of the generated test data is between the range of 1 and 9 and the generated test data is between -2147483648 to 2147483647.

7. Save the changes.

You must repeat steps 2 through step 5 to set the properties for all schemas.

## Results

You have defined the properties for all schemas.

## What to do next

You can click **Schema Canvas** to select the defined and structured schemas that you want to use to generate the test data. See [Selecting multiple schemas on page 217](#).

---

Related information

[Item properties on page 643](#)

[Setting a range on page 185](#)

## Importing schemas into the Schema Canvas page

When you want to generate the test data or the Data Definition Language (DDL) statements by using single or multiple schemas, you must import schemas into the **Schema Canvas** page. You can import multiple schemas simultaneously.

### Before you begin

You must have fabricated schemas in HCL® OneTest™ Data.

### About this task



To import the schemas, you must select the type of schemas that you want to import. You can import either the JDBC-supported schemas or non-JDBC schemas into the **Schema Canvas** page. You can view the list of schemas under the **Schemas** tab of the **Canvas Settings** dialog box. Under the **Groups** tab, you can view the list of all the group types of the schemas. You can also enter the text in the **Search** field to find the schemas or the group types that match with the characters of the search text.

The schemas or the group types that you select to import are listed under the **Selected Schemas** panel.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

**Result**

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

**Result**

The **Data Fabrication** page is displayed.

4. Click **Schema Canvas** to select the schemas.

**Result**

The **Canvas Settings** dialog box is displayed.

5. Select any one of the following options in the **Canvas Settings** dialog box to display the list of schemas:
  - **JDBC**: When you select **JDBC**, you can view only the list of JDBC-supported schemas. To import the JDBC schemas, go to Step 4.
  - **Non-JDBC**: When you select **Non-JDBC**, you can view only the list of non-JDBC schemas. To import the non-JDBC schemas, go to Step 5.
6. For **JDBC** schemas: Select a schema by using any one of the following methods:

**Choose from:**

- Use the **Database** tab:
  - a. Click the **Database** tab.
  - b. Select a schema.


The group types within the selected schemas are selected.



**Notes:**

- You can also search for the schema by entering the name of the schema in the **Search** field.
  - Each JDBC schema includes only one group type.
  - The group types are selected automatically with the selection of schemas.
- Use the **Table** tab:
    - a. Click the **Table** tab.
    - b. Select the group type.

The group types along with their schemas are selected.

 **Note:** You can find only one group type in JDBC schemas.

### Result

The selected schemas are displayed in the **Selected Schemas** panel.

 **Notes:**


- If you select a child schema, then the parent schema is selected automatically.
- If you select a child group, then the parent group of parent schema is selected automatically.
- If you clear the selection of any group type of schema, then you can view the updated list of the selected schemas in the **Selected Schemas** panel.

7. For **Non-JDBC** schemas: Select a schema by using any one of the following methods:


#### Choose from:

- Use the **Schema** tab:
  - a. Click the **Schema** tab.
  - b. Select a schema.

The group types within the selected schema are displayed.

 **Note:** You can also search for the schema by entering the name of the schema in the **Search** field.

- c. Select the group types of the selected schema that you want to import.

 **Note:** You can select multiple group types from a single schema.

- Use the **Group** tab:
  - a. Click the **Group** tab.
  - b. Select the group types that you want to import.

The group types along with their schemas are selected.

### Result

The selected schemas are displayed in the **Selected Schemas** panel.

 **Notes:**



- If you select a child schema, then the parent schema is selected automatically.
- If you select a child group, then the parent group of the parent schema is selected automatically.
- If you clear the selection of any group type of schema, then you can view the updated list of the selected schemas in the **Selected Schemas** panel.

#### 8. Click **Set Canvas**.

##### **Result**

The selected schemas are imported into the **Schema Canvas** page.



**Note:** If you select multiple group types from the same schema and import that schema into the **Schema Canvas** page, then the structure of each group is represented separately on the **Schema Canvas** page.

##### **Results**

You have imported the non-JDBC schemas into the **Schema Canvas** page.

##### **What to do next**

You must select schemas from the **Schema Canvas** page before generating the DDL statements or the test data. See [Selecting schemas on page 225](#).

You can also establish a new parent-child relationship among non-JDBC schemas. See [Establishing a parent-child relationship among schemas](#).

After you import schemas into the **Schema Canvas** page, if you want to modify or remove the selection of schemas, then you must click the **Modify canvas** button on the **Schema Canvas** page. See [Modifying the selection of imported schemas on page 227](#).

## Modifying a relationship among schemas

When you want to alter the existing relationship between schemas, then you can modify the established relationship on the **Schema Canvas** page.

##### **Before you begin**

You must have completed the following tasks:

1. Created a project.
2. Fabricated the schemas in HCL® OneTest™ Data. See [Schema creation on page 165](#).
3. Defined properties for the fabricated schemas. See [Defining properties of schemas on page 219](#).
4. Imported schemas from the **Schema Canvas** page. See [Importing schemas into the Schema Canvas page on page 220](#).
5. Established relationships among non-JDBC schemas. See [Establishing a parent-child relationship among schemas](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

**Result**

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

**Result**

The **Data Fabrication** page is displayed.

4. Click **Schema Canvas**.

**Result**

The **Schema Canvas** page is displayed.

5. Identify an item type of a relationship that you want to modify.
6. Hover over the relationship line that you selected.
7. Click the end of the relationship line that points to the item type you want to modify.
8. Drag-and-drop the line to the item type with which you want to establish a new relationship.



**Notes:**

- You can select an item type either from the same schema or a different schema to establish a new relationship.
- You can modify only the child schemas in a relationship.

**Results**

The relationship is modified among the item types of non-JDBC schemas.

**What to do next**

You can establish another relationship among the item types of other schemas. See [Establishing a parent-child relationship among schemas](#).

You can also select the schemas on the **Schema Canvas** page to generate the test data. See [Selecting schemas on page 225](#).

## Deleting a relationship among schemas

When any established relationship among the item types of non-JDBC schemas is no longer required, you can delete that relationship on the **Schema Canvas** page.

**Before you begin**

You must have completed the following tasks:

1. Created a project.
2. Fabricated the schemas in HCL® OneTest™ Data. See [Schema creation on page 165](#).
3. Defined properties for the fabricated schemas. See [Defining properties of schemas on page 219](#).

4. Imported schemas from the **Schema Canvas** page. See [Importing schemas into the Schema Canvas page on page 220](#).
5. Established relationships among non-JDBC schemas. See [Establishing a parent-child relationship among schemas](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

**Result**

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

**Result**

The **Data Fabrication** page is displayed.


4. Click **Schema Canvas**.

**Result**


The **Schema Canvas** page is displayed.

5. Identify a relationship among schemas that you want to delete.
6. Hover over the relationship line of the selected relationship.

**Result**

You can view a red cross mark  on the line that is drawn to show the relationship among the schemas.

7. Click the red cross mark  to delete the relationship.

Alternatively, click the arrow  of the drawn line, and then drag-and-drop the line anywhere on the **Schema Canvas** page.

**Result**

The **Detach relationship** dialog box is displayed for your confirmation before the deletion of the relationship.

8. Click **Detach**.

**Result**

The line that is drawn among the schemas is erased.

**Results**

The relationship is deleted among the item types of non-JDBC schemas.

**What to do next**

You can establish another relationship among the item types of other schemas. See [Establishing a parent-child relationship among schemas](#).

You can also select the schemas on the **Schema Canvas** page to generate the test data. See [Selecting schemas on page 225](#).

## Selecting schemas

After you import schemas into **Schema Canvas**, you must select schemas to view the relationships among them, generate the Data Definition Language (DDL) statements, or generate the test data for multiple schemas simultaneously.

## Before you begin

You must have completed the following tasks:

1. Created a project.
2. Fabricated the JDBC-supported or non-JDBC schemas in HCL® OneTest™ Data:
  - For JDBC-supported schemas, see [Generating a schema in HCL OneTest Data on page 195](#).
  - For non-JDBC schemas in HCL® OneTest™ Data, see [Schema creation on page 165](#).
3. Defined properties for the generated schemas. See [Defining properties of schemas on page 219](#).
4. Imported schemas into the **Schema Canvas** page. See [Importing schemas into the Schema Canvas page on page 220](#).

## About this task

You can view all the imported schemas on the **Schema Canvas** page. When you select a schema on the **Schema Canvas** page, the other schemas are also selected if they are related to each other based on referential integrity. Otherwise, you can select **Select All** to select all the schemas that are imported on the **Schema Canvas**.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

### Result

The **Data Fabrication** page is displayed.

4. Click **Schema Canvas**.

### Result

The **Schema Canvas** page is displayed.

You can view the structural view of all the imported schemas with their relationships with each other.



### Notes:

- You can use the scroll wheel or pinch-to-zoom option on the **Schema Canvas** page to view the structural view of all schemas and relationships.
  - When you rename or delete any schema or item types that are imported to the **Schema Canvas** page, the relationships are altered and this alteration results in the failure of a generation of test data.
5. Verify the relationships among the selected schemas.
  6. Select schemas based on the following criteria:

- If a schema does not have a relationship with any other schema on the **Schema Canvas** page, then you must select that schema manually. Otherwise, select **Select All** to select all schemas to generate the test data.
- If a schema has a relationship with other schemas and if it is selected, then all the related schemas are selected automatically.



#### Notes:

- You cannot select schemas that belong to different JDBC connections.
- You can use **Modify canvas** to add or remove the selected schemas.
- You must not select schemas with nested group types for the generation of DDL statements.
- If any of the selected schemas is deleted from the workspace, then the view of the **Schema Canvas** page changes because the deleted schema is removed from the **Schema Canvas** page.

### Results

You have selected multiple schemas.

### What to do next

You can perform any one of the following tasks:

- Generate DDL statements for the selected schemas. See [Generating DDL statements on page 236](#).
- Generate the test data. See [Generating test data for selected schemas on page 228](#).

## Modifying the selection of imported schemas

After you import schemas into the **Schema Canvas** page, if you want to modify or remove the selected schemas, then you can modify the **Schema Canvas** page.

### Before you begin

You must have completed the following tasks:

- Created a project.
- Fabricated the schemas in HCL® OneTest™ Data. See [Schema fabrication on page 161](#).
- Defined the properties of schemas. See [Schema design overview on page 170](#).
- Imported the schemas. See [Importing schemas into the Schema Canvas page on page 220](#).

### About this task

After you import the JDBC-supported or non-JDBC schemas from the **Canvas Settings** dialog box into the **Schema Canvas** page, if you want to modify the selection of group types of a schema or number of schemas, then you must use the **Modify canvas** button. You can use the **Canvas Settings** dialog box to reselect or clear the selection of schemas and the group types.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

**Result**

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

**Result**

The **Data Fabrication** page is displayed.

4. Click **Schema Canvas** to select the schemas.

**Result**

The **Schema Canvas** page is displayed.

5. Click the **Modify canvas** button.

**Result**

The **Canvas Settings** dialog box is displayed.

6. Modify the selection of schemas or group types.
7. Click **Set Canvas**.

**Result**

The modified **Schema Canvas** page is displayed.

## Results

You have modified the **Schema Canvas** page.

## What to do next

You must select schemas from the **Schema Canvas** before generating the DDL statements or the test data. See [Selecting schemas on page 225](#).

You can also establish a new parent-child relationship among non-JDBC schemas. See [Establishing a parent-child relationship among schemas](#).

## Generating test data for selected schemas

After you select multiple schemas on the **Schema Canvas** page, you can generate the test data of all the selected schemas simultaneously.

## Before you begin

You must have completed the following tasks:

1. Created a project.
2. Fabricated the schemas in HCL® OneTest™ Data. See [Schema creation on page 165](#).
3. Defined properties for the fabricated schemas. See [Defining properties of schemas on page 219](#).
4. Imported schemas from the **Schema Canvas** page. [Importing schemas into the Schema Canvas page on page 220](#).



5. Established relationships among non-JDBC schemas. Establishing a parent-child relationship among schemas.
6. Selected the schemas. See [Selecting schemas on page 225](#).

### About this task

You can generate the test data for multiple schemas simultaneously. To generate the test data, you can either use only JDBC-supported schemas or non-JDBC schemas.

When you select multiple schemas from the **Schema Canvas** page, and one or more schemas among the selected schemas are enabled with the pairwise property, then the following number of records are generated for those schemas:

- If the selected schema is enabled with the pairwise property and is not referenced to any other schema, then the number of records generated are auto calculated based on the algorithm of the pairwise data generation.
- If the selected schema is enabled with the pairwise property and is referenced to at least one parent schema, then the number of records generated are among the least of the following values:
  - The number of records of each referenced parent schema.
  - The auto calculated number of records of the schema with the pairwise property as enabled.
- If the selected schema is enabled with the pairwise property and is referenced by a child schema, which is not a pairwise enabled schema, then the number of records generated are among the least of the following values:
  - The number of records of each parent schema.
  - The value of the number of records that you enter.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click **Generate data** to generate the test data for the selected schemas on the **Schema Canvas** page.

#### Result

The **Generate data** dialog box is displayed.

5. Select the **Connections** option for the output location to generate the test data for the JDBC-supported schemas.



**Note:** When you want to generate the test data for non-JDBC schemas, the options to select the output location are not displayed because the **File** option is considered as the default option.

6. Enter the number of records that you want to generate in the **Number of records** field.
7. Enter a seed value in the **Numeric Seed Value** field.

The seed value acts as an instance of random data that can generate the same set of data multiple times. The default seed value is blank.



**Note:** The following table lists the maximum values for numeric seed value and number of records:

Description	Maximum values
Numeric seed value	8 digits (99,999,999)
Number of records	9 digits (999,999,999)

8. Choose the output file format of the test data that you want to generate from the **Output File Format** list.
9. Click **Analyze** to validate the selected schemas.

If the selected schemas are validated successfully, then you can view the order of schemas execution for generating the test data. All the parent schemas are executed first followed by the child schemas.



**Note:** If the validation fails, it might be because of one of the following reasons:

- The parent schemas of all the selected schemas are missing.
- The connection and actions of selected schemas do not exist in the project.
- Any one of the following properties of all item types of the selected schemas are not set:
  - **Min size (digits)** and **Max size (digits)** properties for each integer item type.
  - **Set Required** property.

You must resolve the cause of failure for the successful validation of the schemas.

10. Click **OK**.

## Results

You have generated the test data by using multiple schemas simultaneously.

HCL® OneTest™ Data first generates the test data of the parent schema followed by the child schemas.

## What to do next

You can view and download the generated test data of all the selected jobs on the **Jobs** page. See [Status of generated test data on page 237](#).

---

#### Related information

[Inserting the generated test data in a database on page 196](#)

[Item properties on page 643](#)

[Setting a range on page 185](#)

## Generating DDL statements for schemas

When you want to know the Structured Query Language (SQL) statements to create a table from non-JDBC schemas that you created in HCL® OneTest™ Data, you can generate the Data Definition Language (DDL) statements for those selected schemas.

### Before you begin

You must have completed the following tasks:

- Logged into HCL OneTest™ Server.
- Created a project.
- Created schemas in HCL® OneTest™ Data. See [Schema fabrication on page 161](#).

### About this task

You can generate the SQL file with DDL statements to non-JDBC schemas only. You can also generate the DDL statements for multiple schemas simultaneously. To generate the DDL statements for any schema, you must create a schema in HCL® OneTest™ Data, define the properties of the schema, and then import all the non-JDBC schemas for which you want to generate the DDL statements on the **Schema Canvas** page. After you import the schemas into the **Schema Canvas** page, you must select the schemas, and then generate the DDL statements.

You can find the generated SQL file with DDL statements for the selected schemas on the **Jobs** page.



#### Notes:

- You cannot generate the DDL statements for schemas with nested groups.
- The DDL statements for all the selected schemas that are available on the **Schema Canvas** page are generated in a single SQL file.

1. [Create schemas on page 165](#).
2. [Define properties of schemas on page 219](#).
3. [Import non-JDBC-supported schemas on page 220](#).
4. [Select schemas on page 225](#).
5. [Generate DDL statements on page 236](#).

### Results

You have generated the DDL statements for the selected non-JDBC schemas.

### What to do next

You can download the generated DDL statements for each schema from the **Jobs** page. See [Status of generated test data on page 237](#).

## Importing schemas into the Schema Canvas page

When you want to generate the test data or the Data Definition Language (DDL) statements by using single or multiple schemas, you must import schemas into the **Schema Canvas** page. You can import multiple schemas simultaneously.

### Before you begin

You must have fabricated schemas in HCL® OneTest™ Data.

### About this task

To import the schemas, you must select the type of schemas that you want to import. You can import either the JDBC-supported schemas or non-JDBC schemas into the **Schema Canvas** page. You can view the list of schemas under the **Schemas** tab of the **Canvas Settings** dialog box. Under the **Groups** tab, you can view the list of all the group types of the schemas. You can also enter the text in the **Search** field to find the schemas or the group types that match with the characters of the search text.

The schemas or the group types that you select to import are listed under the **Selected Schemas** panel.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click **Schema Canvas** to select the schemas.

#### Result

The **Canvas Settings** dialog box is displayed.

5. Select any one of the following options in the **Canvas Settings** dialog box to display the list of schemas:
  - **JDBC**: When you select **JDBC**, you can view only the list of JDBC-supported schemas. To import the JDBC schemas, go to Step 4.
  - **Non-JDBC**: When you select **Non-JDBC**, you can view only the list of non-JDBC schemas. To import the non-JDBC schemas, go to Step 5.
6. For **JDBC** schemas: Select a schema by using any one of the following methods:

#### Choose from:

- Use the **Database** tab:
  - a. Click the **Database** tab.
  - b. Select a schema.

The group types within the selected schemas are selected.



**Notes:**

- You can also search for the schema by entering the name of the schema in the **Search** field.
- Each JDBC schema includes only one group type.
- The group types are selected automatically with the selection of schemas.

- Use the **Table** tab:
  - a. Click the **Table** tab.
  - b. Select the group type.

The group types along with their schemas are selected.



**Note:** You can find only one group type in JDBC schemas.

**Result**

The selected schemas are displayed in the **Selected Schemas** panel.



**Notes:**

- If you select a child schema, then the parent schema is selected automatically.
- If you select a child group, then the parent group of parent schema is selected automatically.
- If you clear the selection of any group type of schema, then you can view the updated list of the selected schemas in the **Selected Schemas** panel.

7. For **Non-JDBC** schemas: Select a schema by using any one of the following methods:

**Choose from:**

- Use the **Schema** tab:
  - a. Click the **Schema** tab.
  - b. Select a schema.

The group types within the selected schema are displayed.



**Note:** You can also search for the schema by entering the name of the schema in the **Search** field.

- c. Select the group types of the selected schema that you want to import.



**Note:** You can select multiple group types from a single schema.

- Use the **Group** tab:
  - a. Click the **Group** tab.
  - b. Select the group types that you want to import.

The group types along with their schemas are selected.

### Result

The selected schemas are displayed in the **Selected Schemas** panel.



#### Notes:

- If you select a child schema, then the parent schema is selected automatically.
- If you select a child group, then the parent group of the parent schema is selected automatically.
- If you clear the selection of any group type of schema, then you can view the updated list of the selected schemas in the **Selected Schemas** panel.

8. Click **Set Canvas**.

### Result

The selected schemas are imported into the **Schema Canvas** page.



**Note:** If you select multiple group types from the same schema and import that schema into the **Schema Canvas** page, then the structure of each group is represented separately on the **Schema Canvas** page.

### Results

You have imported the non-JDBC schemas into the **Schema Canvas** page.

### What to do next

You must select schemas from the **Schema Canvas** page before generating the DDL statements or the test data. See [Selecting schemas on page 225](#).

You can also establish a new parent-child relationship among non-JDBC schemas. See [Establishing a parent-child relationship among schemas](#).

After you import schemas into the **Schema Canvas** page, if you want to modify or remove the selection of schemas, then you must click the **Modify canvas** button on the **Schema Canvas** page. See [Modifying the selection of imported schemas on page 227](#).

## Selecting schemas

After you import schemas into **Schema Canvas**, you must select schemas to view the relationships among them, generate the Data Definition Language (DDL) statements, or generate the test data for multiple schemas simultaneously.

### Before you begin

You must have completed the following tasks:

1. Created a project.
2. Fabricated the JDBC-supported or non-JDBC schemas in HCL® OneTest™ Data:
  - For JDBC-supported schemas, see [Generating a schema in HCL OneTest Data on page 195](#).
  - For non-JDBC schemas in HCL® OneTest™ Data, see [Schema creation on page 165](#).
3. Defined properties for the generated schemas. See [Defining properties of schemas on page 219](#).
4. Imported schemas into the **Schema Canvas** page. See [Importing schemas into the Schema Canvas page on page 220](#).

### About this task

You can view all the imported schemas on the **Schema Canvas** page. When you select a schema on the **Schema Canvas** page, the other schemas are also selected if they are related to each other based on referential integrity. Otherwise, you can select **Select All** to select all the schemas that are imported on the **Schema Canvas**.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click **Schema Canvas**.

#### Result

The **Schema Canvas** page is displayed.

You can view the structural view of all the imported schemas with their relationships with each other.



**Notes:**



- You can use the scroll wheel or pinch-to-zoom option on the **Schema Canvas** page to view the structural view of all schemas and relationships.
- When you rename or delete any schema or item types that are imported to the **Schema Canvas** page, the relationships are altered and this alteration results in the failure of a generation of test data.

5. Verify the relationships among the selected schemas.

6. Select schemas based on the following criteria:

- If a schema does not have a relationship with any other schema on the **Schema Canvas** page, then you must select that schema manually. Otherwise, select **Select All** to select all schemas to generate the test data.
- If a schema has a relationship with other schemas and if it is selected, then all the related schemas are selected automatically.



**Notes:**

- You cannot select schemas that belong to different JDBC connections.
- You can use **Modify canvases** to add or remove the selected schemas.
- You must not select schemas with nested group types for the generation of DDL statements.
- If any of the selected schemas is deleted from the workspace, then the view of the **Schema Canvas** page changes because the deleted schema is removed from the **Schema Canvas** page.

## Results

You have selected multiple schemas.

## What to do next

You can perform any one of the following tasks:

- Generate DDL statements for the selected schemas. See [Generating DDL statements on page 236](#).
- Generate the test data. See [Generating test data for selected schemas on page 228](#).

## Generating DDL statements

After you select non-JDBC schemas on the **Schema Canvas** page, you can generate the Data Definition Language (DDL) statements for all the selected schemas.

### Before you begin

You must have selected non-JDBC schemas on the **Schema Canvas** page.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

### Result

The **Overview** page is displayed.



3. Click **Author > Data Fabrication**.

**Result**

The **Data Fabrication** page is displayed.

4. Click **Schema Canvas**.

**Result**

The **Schema Canvas** page is displayed.

5. Verify the selected schemas.

6. Click **Generate DDL**.

**Result**

A notification is displayed that the DDL statements for the selected schemas are successfully generated.

## Results

You have generated the DDL statements of the selected schemas.

## What to do next

You can view and download the generated DDL statements of the selected schemas on the **Jobs** page. See [Status of generated test data on page 237](#).

## Status of generated test data

You can view the status of all the test data generated jobs and Data Definition Language (DDL) statements generated jobs performed in your project, on the **Jobs** page.

You can view the status of test data generated jobs in the **Status** column on the **Jobs** page. When the test data generation of any schema initiates, the status of that job is shown as *Started*. As the test data generation progresses, the status of that job changes to *In Progress*. After the test data generation completes successfully, the status changes to *Complete*. If the generation of test data fails, you can view the status of that particular job as *Failed*.

---

### Related information

[Usage of test data generation job on page 237](#)

## Usage of test data generation job

The generation of test data or Data Definition Language (DDL) statements for single or multiple schemas in HCL® OneTest™ Data is considered a job. You can view the list of all jobs on the **Jobs** page.

When the job to generate the test data or DDL statements is complete, you can download the following generated jobs files from the **Jobs** page:

- Generated test data file
- Structured Query Language (SQL) file

You can use the job details to troubleshoot the failed jobs and to regenerate the successful jobs.

When you generate the DDL statements, each job includes one SQL file and each SQL file includes the DDL statements generated for all the selected schemas.

You can access the API history for each generation job of the test data or DDL statements. If you want to reuse any schema to generate the test data, then you can regenerate the test data.



**Note:** The **Regenerate Job** option is disabled for the **Schema canvas jobs** and **DDL job**.

You can find different methods of using the generated test data jobs.

## Downloading jobs

When you want to test any application by using the generated test data or to reuse the data of any specific schema definition for some other project, you can download the test data from the job into your local file system. You can also download the Structured Query Language (SQL) file from the job that includes the generated Data Definition Language (DDL) statements of multiple non-JDBC schemas.

### Before you begin

You must have completed the following tasks:

- Created a project.
- Generated the test data. See [Generating test data for selected schemas on page 228](#).
- Generated the DDL statements. See [Generating DDL statements on page 236](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

#### Result

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

#### Result

The **Data Fabrication** page is displayed.

4. Click **Jobs**.

#### Result

The **Jobs** page is displayed. You can view the list of all the jobs that include the generated test data or DDL statements.



#### Notes:

- If you want to view only the jobs related to referential integrity, then select **Schema canvas jobs** from the menu.
- If you want to view only the jobs related to DDL statements, then select **DDL job** from the menu.

5. Select a job for which you want to download the generated test data.

**Notes:**

- You can select multiple jobs at a time to download the test data.
- You can download the test data only for the jobs that are completed successfully.
- The download option is disabled for the jobs that are set to insert the generated test data into the database.
- The download option is disabled for multiple DDL jobs.

6. Click the **Download job(s)** button or  icon.

**Results**

You have downloaded any one of the following files:

- The generated test data in the compressed format in your local file system. When you download multiple jobs, then the compressed file of the jobs is organized by time.



**Note:** You can also download the generated test data file from the following location of the HCL® OneTest™ Data pod:

```
/opt/hcl/hip-rest/  
output/<accountId>/<userId>/<projectId>/<schemaId>/<genMapPath>
```

- An SQL file that includes the generated DDL statements of all the selected schemas.



**Note:** You can also download the DDL statements from the following location of the HCL® OneTest™ Data pod:

```
/opt/data/hipfiles/<accountId>/<projectId>/yyyy:MM:dd-HH:mm:ss.sql
```

## Regenerating test data

When you want to generate another set of test data by using an existing schema, then instead of defining a schema again, you can regenerate test data from the **Jobs** page.

**Before you begin**

- You must have a project.
- You must have generated test data.

**About this task**

On the **Jobs** page, you can view the list of all test data generated jobs. If you want to use the existing data model or schema to generate different values of test data, you must regenerate with an empty seed value. However, if the seed value is not empty, then the same test data is generated as it was generated earlier for that schema.



**Note:** You cannot regenerate the test data for the jobs that are generated for multiple schemas by using referential integrity and Data Definition Language (DDL) statements.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

**Result**

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

**Result**

The **Data Fabrication** page is displayed.


4. Click **Jobs**.

**Result**

You can view the list of data generated jobs.

5. Select a job for which you want to regenerate the test data.

You can select multiple jobs at a time to regenerate the test data.

6. Click **Regenerate data** from the menu or click the **Regenerate data** icon .

## Results

The test data for the required schema is regenerated and is listed on the **Jobs** page.

## What to do next

You can download the regenerated test data. See, [Downloading test data on page 238](#).

## Configuring job settings

HCL® OneTest™ Data periodically deletes all the generated test data and the jobs of test data generation with API history to clear the memory. However, if you want to change the default number of days to retain the generated test data jobs in memory, you must configure the job settings.

### Before you begin

- You must have cluster-admin permissions.
- You must have the IP address of the computer where HCL OneTest™ Server is installed.

### About this task

As a default configuration, HCL® OneTest™ Data retains the jobs with their generated test data and API history for five days after the jobs are completed. All the jobs that are created before five days are deleted. If you want to retain the jobs with their generated data for more number of days in HCL® OneTest™ Data, then you must modify the value of the **data\_rotation\_time** property to the number of days you want to retain the jobs in the `configmap` file.



**Note:** If you modify the value of the **data\_rotation\_time** property to zero, then you can disable the deletion of generated test data jobs and API history.

1. Log in to the SSH console of HCL OneTest Server.
2. Run the following command to edit the configuration file:

```
kubectl edit configmap -n test-system {my-ots}-data-config -o yaml
```

3. Search for the **data\_rotation\_time** property in the configuration file, and then set the value to the number of days you want the jobs to retain in the server.



**Note:** The default value of **data\_rotation\_time** is 5.

For example, if you want to retain the jobs for 10 days, set the value of **data\_rotation\_time** property as *10*.

4. Save your changes, and then exit from the configuration file.
5. Run the following command to delete and restart the `<onetest data>` pod:

```
kubectl delete pod {my-ots}-data-app-0
```

## Results

You have successfully configured the setting to retain the jobs with generated test data and API history in HCL® OneTest™ Data.

## Jobs management

When you run a job to generate the test data, you can view the job details on the **Jobs** page. The **Jobs** page helps you to view and to manage the list of all the jobs that you execute.

You can perform a quick search for the test data generated jobs based on the name of the schema or the date range by selecting the *From* and *To* dates. If the *To* field is blank, you can view the results till the most recent date. After you perform the search, if you want to continue to search for any schema or specific date, you must clear the **Search** field by using the **Clear Filter** button.

You can delete and cancel the generated test data jobs.

## Deleting a test data generated job

When any of the generated test data is of no further usage, you can delete that specific test data from your project.

### Before you begin

- You must have a project.
- You must have generated test data.

### About this task

On the **Jobs** page, you can view the list of all the projects for which you have generated the test data. You can remove the reference of any specific test data.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

**Result**

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

**Result**

The **Data Fabrication** page is displayed.


4. Click **Jobs**.

**Result**

You can view the list of data generated jobs.

5. Select a job for which you want to delete the test data.

You can select multiple jobs at a time to delete.

6. Click **Delete Job** from the menu or click the **Delete** icon .

## Results

The selected test data is deleted from your project.

## Canceling a test data generation job

After initiating a test data generation job, if you want to modify the schema or terminate the job before the job completes, then you can cancel the running test data generation job.

### Before you begin

You must have a test data generation job in the `In Progress` status.

### About this task

You can cancel only one test data generation job at a time. You can view the details of API endpoints of the canceled job on the **API History** page. Later, if you want to reuse the same schema to generate the test data, you must select **Regenerate Job** to regenerate the test data. However, you cannot download the test data of the canceled job.



**Note:** You can hover the cursor over the job to view the details of the member who canceled the job along with the timestamp.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.

**Result**

The **Overview** page is displayed.

3. Click **Author > Data Fabrication**.

**Result**

The **Data Fabrication** page is displayed.

4. Go to the **Jobs** page and select the test data generation job that you want to cancel.
5. Click **Cancel Job** from the menu of the selected job.

A notification is displayed that the selected job is canceled.

### **Results**

You successfully canceled the running job. The canceled job appears on the **Jobs** page with the status as `Canceled`.

### **What to do next**

You can modify the schema by using the schema designer and generate the test data.

# Chapter 6. Test Execution Specialist Guide

This guide describes tasks that you must complete before you can configure and run tests in HCL OneTest™ Server. You can find information about configuring runs for the different test types that are supported. You can also find information about other tasks that you can perform on the Resource Monitoring Service, Virtualization, and Integrations with third-party applications. This guide is intended for testers or test execution specialists.

## System modeling

In this topic, you can find information about using the system model feature. You can also find the tasks that you can perform to create a system model. You must configure a repository to save the system model that you created for the application under test.

### Introduction to system model

A system model is a logical presentation of the components contained in a system under test and the relationships between components. You can create a system model when you want to visualize the different components and their relationships as a diagrammatic representation.

You can create only one system model within a team space on HCL OneTest™ Server. A team space repository is required to contain the system model that you create. Before you can create a system model, you must add a team space repository. The system model that is created or edited is stored in the team space repository. All projects within a team space share the same system model. All members of a team space can create, edit, and view the system model.

After you log in to HCL OneTest™ Server, you can go to the **System Model** page to create a system model for your team space. Alternatively, you can go to any of the projects in your team space and create a system model from the **System Model** page within a project. The system model is accessible from all the projects of your team space repository.

### Components in a system model

Components are the basic building blocks which are used to represent a piece of software in an application.

You can model a system by using components. You can create components that represent the type of asset or resource used by the system or application under test and can be of the following types:

Component type	Description
Database	The component type represents the database resource. You can select this component type when you want to represent a database asset or resource in the system model.
Service	The component type represents the virtual service resource. You can select this component type in the system model to represent a virtual service in the team space repository or associate a virtual service resource that is in the project repository.



Component type	Description
UI	The component type represents the User Interface (UI) resource. You can select this component type when you want to represent the User Interface resource in the system model.











You can create multiple components to depict the different resources in the system or application under test. You can also associate virtual service resources with the components. The components in the system model can then be associated with the virtual services stored in the repositories that are configured for each of the projects.

You can create the following types of relationships between the components:

- Components can be related as *child* components to *parent* components.
- Components can be related as *dependent* components. The dependency is displayed by a line-arrow with the arrow-head pointing to the dependent component. For example, if *Component A* depends on *Component B*, then the relationship is displayed with a line-arrow from *Component A* to *Component B* and the arrow points to *Component B*. If *Component B* is also dependent on *Component A*, then the relationship is displayed with a line-arrow from *Component B* to *Component A* and the arrow points to *Component A*.








## Using the system model UI


The following table lists the icons used in a system model and the icons that are displayed on the **System Model** page:

Icon	Description
	Represents a system model.
	Represents Suites and Tests in a repository.
	Represents a database resource. The component name is prefixed with this icon when you select the component type as database.
	Represents a service resource. The component name is prefixed with this icon when you select the component type as service.
	Represents a User Interface (UI) resource. The component name is prefixed with this icon when you select the component type as User Interface.
	Used to add components, child components, and dependent components.
	Used to search for components in the system model.
	Represents the virtual services.
	Used to publish the system model to the team space repository.
	Represents the dependency between the components.

Icon	Description
—	Represents the inherited dependencies.

You can perform the following operations when you view the model displayed on the **System Model** page:

Operation	Action
Pan the view of the model	<ol style="list-style-type: none"> <li>1. Click the <b>Pan</b> icon .</li> <li>2. Click on the area of the model that you want to pan and move the area keeping the mouse pressed.</li> </ol> <p> <b>Note:</b> You must click the <b>Pan</b> icon to exit from the pan mode.</p>
Zoom-in the view of the model	<ol style="list-style-type: none"> <li>1. Click the <b>Zoom-in</b> icon .</li> </ol> <p>The view is enlarged or zoomed-in.</p> <p> <b>Note:</b> You can repeatedly click the <b>Zoom-in</b> icon to further enlarge the view.</p>
Zoom-out the view of the model	<ol style="list-style-type: none"> <li>1. Click the <b>Zoom-out</b> icon .</li> </ol> <p>The view is diminished or zoomed-out.</p> <p> <b>Note:</b> You can repeatedly click the <b>Zoom-out</b> icon to further diminish the view.</p>
Reset the view	<ol style="list-style-type: none"> <li>1. Click the <b>Menu</b> icon  on the <b>System Model</b> page.</li> <li>2. Select <b>Reset Zoom</b>.</li> </ol> <p>The view is reset to the default view without any zoom applied. If the view was zoomed-in or zoomed-out, the view is reset to 100% of the size.</p>

Operation	Action
Position the model view	<ol style="list-style-type: none"> <li>1. Click the <b>Menu</b> icon  on the <b>System Model</b> page.</li> <li>2. Select from the following options: <ul style="list-style-type: none"> <li>◦ <b>Center</b>: Select this option to align the view of the model to the center of the page.</li> <li>◦ <b>Fit to View</b>: Select this option to fit the view of the model in the window on the <b>System Model</b> page.</li> </ul> </li> </ol>

### Tasks in system modeling

When you want to model the system under test for the first time after you install the server software and you are a licensed user in the team space, you must first add the team space repository. See [Adding a repository to a team space on page 248](#).

After a repository is configured as the team space repository, you can work with the system model. See [Tasks for working with a system model on page 247](#).

When you want to change, update or delete the repository, you can work with the team space repository. See [Tasks for working with the team space repository on page 248](#).

### Task flows for working with a system model

You can find information about the tasks that you can perform when you want to create a system model, create components, associate the components, create linkages among the components, and then publish the system model to the team space repository. Also, you can find information about the tasks that you must perform when you want to modify an existing system model.

### Tasks for working with a system model

You can find the tasks when you want to create a system model for the application under test in your team space.

Tasks	More information
Add a repository to the team space.	<a href="#">Adding a repository to a team space on page 248</a>
Create a system model.	<a href="#">Creating a system model on page 251</a>
Create components in a system model.	<a href="#">Creating components on page 252</a>
Create child components to existing components in a system model.	<a href="#">Creating child components on page 253</a>

Tasks	More information
Define the relationship between the components in a system model.	<a href="#">Creating linkages between components on page 255</a>
Delete the relationship between the components in a system model.	<a href="#">Deleting linkages between components on page 257</a>
Modify components in a system model.	<a href="#">Modifying components on page 259</a>
Delete components in a system model.	<a href="#">Deleting components on page 260</a>
Associate resources with components in a system model.	<a href="#">Associating resources with components on page 262</a>
View resources associated with components in a system model.	<a href="#">Viewing resources associated with components on page 263</a>
Remove resources that are associated with components in a system model.	<a href="#">Removing resources associated with components on page 267</a>
Publish changes made to the system model.	<a href="#">Publishing changes to the system model on page 261</a>
Publish changes of associating resources in a project with components in the system model.	<a href="#">Publishing changes of associating resources with components on page 268</a>
View system model.	<a href="#">Viewing the system model on page 269</a>
Delete system model.	<a href="#">Deleting a system model on page 270</a>

### Tasks for working with the team space repository

You can find the tasks that you can perform when you want to work with the team space repository.

Tasks	More information
Add a repository to the team space.	<a href="#">Adding a repository to a team space on page 248</a>
Delete a team space repository.	<a href="#">Deleting a repository that is added to a team space on page 270</a>

### Adding a repository to a team space

You must add a repository to the team space when you want to create, modify, or store a system model.

#### Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Team Space Owner* or an *Architect* in the team space. See [Managing members and their roles in a team space on page 576](#).
- Been provided valid credentials to access the Git repository that you want to add.

### About this task

As a *Team Space Owner* or an *Architect*, you can add a repository to the team space. You can add a repository to the team space in any of the following methods:

- Add an existing repository that can contain test assets, which are created in the desktop products.
- Add a bare repository.



**Attention:** The repository that you want to add must not contain any projects or `.project` files under the root directory.

When you add a repository, the Git repository is cloned to the team space. While adding a repository, you must provide the necessary authentication credentials that are set for the Git repository. For example, if the authentication type is SSH, then you must provide the Git URL, a deploy key, and a passphrase.



**Important:** You can add only one repository to the team space. After you add a repository to the team space, the option to add another repository is not available. You must delete an existing repository before you can add a different repository.

1. Log in to HCL OneTest™ Server.
2. Open the team space to which you want a team space repository.

The **Active projects** page is displayed.

3. Click the **Settings** icon  from the left navigation pane, and then select **Manage team space**.

The **Details** page is displayed.

4. Click **Repository** on the left navigation pane.

The **Repository** page is displayed.

5. Click **Add repository**.

The **Add repository** dialog is displayed.

6. Enter the URL of the Git repository, which you want to add to the team space, in the **Git Repository** field.

The required fields are displayed based on the type of Git URL that you entered.

7. Enter the required credentials based on any of the following authentication methods configured in the repository.

To gain access to the repository, you must use any one of the authentication methods:

Authentication method	Credentials required
SSH	<ul style="list-style-type: none"> <li>◦ Deploy key</li> <li>◦ Passphrase</li> </ul>
HTTPS	<ul style="list-style-type: none"> <li>◦ User name</li> <li>◦ Password</li> </ul>
HTTP	<ul style="list-style-type: none"> <li>◦ User name</li> <li>◦ Password</li> </ul>



**Notes:**

- You must have defined the authentication type and set the authentication credentials in the Git repository.
- If you use SSH to connect to your remote repository and HCL OneTest™ Server displays an `Auth Fail` exception while using the deliver changes option, you can resolve this exception error by regenerating your SSH keys by using the `-m PEM` option.

8. Click **Add**.

The Git repository is added to the team space on HCL OneTest™ Server.



**Note:** Depending on the size of the repository you are cloning, it can take a few to several minutes to clone the repository.

### What to do next

You can perform the following actions on the repository that you added:

- Update the authentication credentials if they are changed in the Git repository configuration. See [Updating the authentication credentials of the repository on page 594](#).
- Refresh a repository to fetch and synchronize changes from the remote repository.
- Configure a webhook to notify the server if there is a push event in the remote repository. See [Creating webhooks on page 598](#).
- Add a system model to the repository. See [Creating a system model on page 251](#).
- Delete a repository that is added to the team space, if it is no longer required.

---

#### Related information

[Managing repositories on page 592](#)

## Creating a system model

Before you can create components in a system model, you must create an empty system model and publish the system model to the team space repository.

### Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as an *Architect* in the team space, a *Project Owner* or a *Tester* in the project. See [Managing members and their roles in a team space on page 576](#) or [Managing access to server projects on page 588](#).
- Added a repository to the team space. See [Adding a repository to a team space on page 248](#).

1. Log in to HCL OneTest™ Server.
2. Open the team space to which a team space repository is added.

The **Active projects** page is displayed.

3. Select an option to create a system model based on your role in the team space or in a project in the team space:
  - If you are a *Team space owner* or an *Architect*, go to [Step 4 on page 251](#).
  - If you are not a *Team space owner* or an *Architect* but a *Project Owner* or *Tester*, go to [Step 5 on page 251](#).
4. Perform the following steps, if you are a *Team space owner* or an *Architect*:

- a. Click **System Model** on the left navigation pane.

The **System Model** page is displayed.

- b. Click the **Add system model** icon .

An empty system model is created.

- c. Go to [Step 6 on page 252](#).

5. Perform the following steps, if you are not a *Team space owner* or an *Architect* but a *Project owner*:

- a. Open your project that is displayed under **My projects** on the **Active projects** page.

The **Overview** page is displayed.

- b. Click **System Model** on the left navigation pane.

The **System Model** page is displayed.

- c. Click the **Add system model** icon .

An empty system model is created.

d. Go to [Step 6 on page 252](#).

6. Click the **See unpublished commits** icon , and then select the **Publish changes** option.



**Note:** If you do not want to commit the changes that you made to the system model, you can select the **Discard changes** option.

7. Enter a commit message in the **Description of change** field, and then click **Publish**.

The empty system model is committed and published to the team space repository. When the system model is published to the team space repository, other members of the team space can work with the same system model from any of their projects in the team space.

## Results

You have created an empty system model and published it to the team space repository.

## What to do next

You can perform any of the following tasks:

- Create components in the system model. See [Creating components on page 252](#).
- Create child components for the components that you added to the system model. See [Creating child components on page 253](#).

## Creating components

The first step in building a system model is to create components. The components serve as the building blocks for the system model.

### Before you begin

You must have completed the following tasks:

- Created and published the system model. See [Creating a system model on page 251](#).

1. Log in to HCL OneTest™ Server.
2. Open the team space to which a team space repository is added.


The **Active projects** page is displayed.

3. Click **System Model** on the left navigation pane.

The **System Model** page is displayed.

4. Perform the following steps to add a component to the system model:






a. Click the **Add component** icon  in the **Components** row on the right pane.

The **Add a component** dialog is displayed.

b. Enter a name for the component in the **Name** field.

c. Select the type of component that you want to add from the following options displayed in the **Type** field.

-  **Service**
-  **Database**
-  **UI**

d. Click **Add**.

The component is displayed as a circular block on the system model pane with the icon and name of the component below the block. The added component is also displayed under the **Components** list in the right pane.

You create components of other types or the same type in the system model following the preceding steps.

## Results

You have created components in the system model.

## What to do next

You can perform any of the following tasks:

- Publish the changes that you made to the system model. See [Publishing changes to the system model on page 261](#).
- Create child components for the components that you added to the system model. See [Creating child components on page 253](#).
- Create linkages between components in the system model. See [Creating linkages between components on page 255](#).

## Creating child components

You can create components as children of existing components in the system model.

### Before you begin

You must have completed the following tasks:

- Created and published the system model. See [Creating a system model on page 251](#).
- Created components in the system model. See [Creating components on page 252](#).

### About this task

You can create multiple components in your system model and multiple child components under any component in the system model. After you create a child for a component, the child is represented as a circular block within the circular component block in the system model.

1. Log in to HCL OneTest™ Server.
2. Open the team space to which a team space repository is added.


The **Active projects** page is displayed.

3. Click **System Model** on the left navigation pane.

The **System Model** page is displayed with the existing components.




4. Create child components in any of the following methods:
  - To create child components from the left pane on the **System Model** page, go to [Step 5 on page 254](#).
  - To create child components from the right pane on the **System Model** page, go to [Step 6 on page 254](#).
5. Perform the following steps to create child components from the left pane:
  - a. Click the component under which you want to create child components in the left pane.

The details of the component are displayed in the right pane.

- b. Click the **Add child** icon  in the **Children** row on the right pane.

The **Add a child to...** dialog is displayed.

- c. Enter a name for the component in the **Name** field.
- d. Select the type of component that you want to add from the following options displayed in the **Type** field.


-  **Service**
-  **Database**
-  **UI**




- e. Click **Add**.

The child component is displayed as a circular block within the circular block of the parent component on the system model page. The added child component is also displayed under the **Children** list under the component in the right pane.

You create child components of other types or the same type for the same or other components in the system model following the preceding steps.

6. Perform the following steps to create child components from the right pane of the stem model page:

- a. Click the component name listed under the **Components** in the right pane.
- b. Click the **Add child** icon  in the **Children** row on the right pane.  
The **Add a child to...** dialog is displayed.
- c. Enter a name for the component in the **Name** field.
- d. Select the type of component that you want to add from the following options displayed in the **Type** field.

-  **Service**
-  **Database**
-  **UI**

- e. Click **Add**.

The child component is displayed as a circular block within the circular block of the parent component on the system model pane. The added child component is also displayed under the **Children** list under the component in the right pane.

You create child components of other types or the same type for the same or other components in the system model following the preceding steps.

## Results

A component is created and it is associated as a child with the selected component and the name of the child is displayed under the **Children** section in the right pane.

## What to do next

You can perform any of the following tasks:

- Publish the changes that you made to the system model. See [Publishing changes to the system model on page 261](#).
- Create linkages between components in the system model. See [Creating linkages between components on page 255](#).

## Creating linkages between components

You can create linkages between components that display the relationship between them. The linkages can be between components or between child components of the same or different parent components.

### Before you begin

You must have completed the following tasks:

- Created and published the system model. See [Creating a system model on page 251](#).
- Created components in the system model. See [Creating components on page 252](#).
- Created child components. See [Creating child components on page 253](#).

**About this task**

You can create multiple components in your system model and multiple child components under any component in the system model. You can specify the dependency among the components.

1. Log in to HCL OneTest™ Server.
2. Open the team space to which a team space repository is added.

The **Active projects** page is displayed.


3. Click **System Model** on the left navigation pane.


The **System Model** page is displayed.

4. Perform the following steps to create a relationship between the components in the system model:
  - a. Click the component that you want.

The details of the component are displayed in the right pane.

- b. Select from the following types of relationships that you want to create:

Option	Action
Depends on	Perform the following steps, if the component you selected is dependent on another component: <ol style="list-style-type: none"> <li>i. Click the <b>Add dependency</b> icon  in the row of the <b>Depends on</b> in the right pane.</li> <li>The <b>Add dependency</b> dialog is displayed.</li> <li>ii. Select the component from the list of components displayed under <b>Depends on</b>.</li> <li>iii. Click <b>Add</b>.</li> </ol> The components linked are listed under <b>Depends on</b> in the right pane. The linkage between the first to the second component you select is shown with an arrow that starts from the first and ends in the second component.
Is a dependency of	Perform the following steps, if the component you selected has a dependency of another component:

Option	Action
	<p>i. Click the <b>Add dependent</b> icon  in the row of the <b>Is a dependency of</b> in the right pane.</p> <p>The <b>Add dependent</b> dialog is displayed.</p> <p>ii. Select the component from the list of components displayed under <b>Is a dependency of</b>.</p> <p>iii. Click <b>Add</b>.</p> <p>The components linked are listed under <b>Is a dependency of</b> in the right pane. The linkage between the first to the second component you select is shown with an arrow that starts from the second and ends in the first component.</p>

### Results

You have created linkages between components in the system model.

### What to do next

You can publish the changes that you made to the system model. See [Publishing changes to the system model on page 261](#).

## Deleting linkages between components

When you do not want to retain the linkages between components that you created in the system model, you can delete the linkages.

### Before you begin

You must have completed the following tasks:

- Created and published the system model. See [Creating a system model on page 251](#).
- Created components in the system model. See [Creating components on page 252](#).
- Created child components. See [Creating child components on page 253](#).
- Created linkages between components in the system model. See [Creating linkages between components on page 255](#).

1. Log in to HCL OneTest™ Server.
2. Open the team space to which a team space repository is added.

The **Active projects** page is displayed.

3. Click **System Model** on the left navigation pane.



The **System Model** page is displayed.

4. Perform the following steps to delete a relationship between the components in the system model:

- a. Click the component that you want.

The details of the component are displayed in the right pane.

- b. Select from the following types of relationships that you want to delete:

When...	Action
<p>The component depends on another component.</p>	<p>Perform the following steps, if the component you selected is dependent on another component:</p> <ol style="list-style-type: none"> <li>i. Select the component from the list of components displayed under the <b>Depends on</b> section in the right pane.</li> <li>ii. Click the <b>Delete selected</b> icon  inline with <b>Depends on</b> section.</li> </ol> <p>The <b>Delete relationship</b> dialog is displayed with the details of the component and the dependent component.</p> <ol style="list-style-type: none"> <li>iii. Click <b>Delete</b>.</li> </ol> <p>The component linkage is removed from the system model pane.</p>
<p>The component has a dependency by another component.</p>	<p>Perform the following steps, if the component you selected has a dependency of another component:</p> <ol style="list-style-type: none"> <li>i. Select the component from the list of components displayed under the <b>Is a dependency of</b> section in the right pane.</li> <li>ii. Click the <b>Delete selected</b> icon  inline with <b>Is a dependency of</b> section.</li> </ol> <p>The <b>Delete relationship</b> dialog is displayed with the details of the component and the component dependency.</p> <ol style="list-style-type: none"> <li>iii. Click <b>Delete</b>.</li> </ol> <p>The component linkage is removed from the system model pane.</p>

### Results

You have created linkages between components in the system model.

### What to do next

You can publish the changes that you made to the system model. See [Publishing changes to the system model on page 261](#).

## Modifying components

At any point in time, after you create a component, you can modify and update the details of the components such as the component type, and the relationships with other components.

### Before you begin

You must have completed the following tasks:

- Created and published the system model. See [Creating a system model on page 251](#).
- Created components in the system model. See [Creating components on page 252](#).
- Created child components. See [Creating child components on page 253](#).

1. Log in to HCL OneTest™ Server.
2. Open the team space to which a team space repository is added.

The **Active projects** page is displayed.



3. Click **System Model** on the left navigation pane.





The **System Model** page is displayed.

4. Click the component that you want to edit.

The component details are displayed in the right pane.

5. Perform the actions for the task to modify components as listed in the following table:

Task	Action
Change the component type.	<p>Perform the following steps:</p> <ol style="list-style-type: none"> <li>a. Identify the component that you want to modify from the components listed under <b>Components</b> in the right pane.</li> <li>b. Click the component checkbox.</li> </ol> <p>The <b>Edit selected</b> icon  is enabled.</p> <ol style="list-style-type: none"> <li>c. Click the <b>Edit</b> icon .</li> </ol> <p>The <b>Edit component</b> dialog box is displayed.</p> <ol style="list-style-type: none"> <li>d. Select the component type.</li> <li>e. Click <b>Modify</b></li> </ol> <p>The type of the component is modified.</p>
Change the name of a component.	Perform the following steps:

Task	Action
	<p>a. Identify the component that you want to modify from the components listed under <b>Components</b> in the right pane.</p> <p>b. Click the component checkbox.</p> <p>The <b>Edit selected</b> icon  is enabled.</p> <p>c. Click the <b>Edit</b> icon .</p> <p>The <b>Edit component</b> dialog box is displayed.</p> <p>d. Edit the component name in the <b>Name</b> field.</p> <p>e. Click <b>Modify</b>.</p> <p>The name of the component is modified.</p>
Change the parent of a component.	<p>Perform the following steps:</p> <p>a. Identify the component that you want to modify from the components listed under <b>Components</b> in the right pane.</p> <p>b. Click the component checkbox.</p> <p>The <b>Change parent</b> icon  is enabled.</p> <p>c. Click the <b>Change parent</b> icon .</p> <p>The <b>Change parent</b> dialog box is displayed.</p> <p>d. Select another parent for the selected components that are listed under the <b>Parent</b> list.</p> <p>e. Click <b>Change</b>.</p> <p>The parent of the component is changed.</p>

### Results

You have edited components and updated the details successfully.

### What to do next

You can publish the changes that you made to the system model. See [Publishing changes to the system model on page 261](#).

## Deleting components

When you do not require a component, you can delete the component.

### Before you begin



You must have completed the following tasks:

- Created and published the system model. See [Creating a system model on page 251](#).
- Created components in the system model. See [Creating components on page 252](#).

1. Log in to HCL OneTest™ Server.
2. Open the team space to which a team space repository is added.

The **Active projects** page is displayed.

3. Click **System Model** on the left navigation pane.

The **System Model** page is displayed.

4. Click the component that you want to delete.

The component details are displayed in the right pane.

5. Click the **Delete component** icon  inline with the component name.

The **Delete component** dialog is displayed.

6. Click **Delete**.

## Results

You have deleted a component from the system model.

## What to do next

You can perform any of the following tasks:

- Publish the changes that you made to the system model. See [Publishing changes to the system model on page 261](#).
- Create components in the system model. See [Creating components on page 252](#).

## Publishing changes to the system model

When you create, delete, or modify components in a system model, the changes are saved locally and are not visible to the other project members or team space members. You must publish the system model changes to the team space repository to make your changes visible for all users of the team space.

### Before you begin

You must have completed the following tasks:


- Created and published the system model. See [Creating a system model on page 251](#).
- Performed actions to add or modify the system model that you want to publish.

1. Log in to HCL OneTest™ Server.
2. Open the team space to which a team space repository is added.

The **Active projects** page is displayed.

3. Click **System Model** on the left navigation pane.

The **System Model** page is displayed.

4. Perform any of the following actions:
  - Create a system model.
  - Delete a system model.
  - Modify a system model.
5. Click the **See unpublished commits** icon , and then select the **Publish changes** option.



**Note:** If you do not want to commit the changes that you made to the system model, you can select the **Discard changes** option.

6. Enter a commit message in the **Description of change** field, and then click **Publish**.

## Results

You have published the changes that you made in the system model to the team space repository. Other members of the team space can now view and use the system model.

## Associating resources with components

After creating components in the system model, you can navigate to a project in the team space to associate the Suites and Tests or virtual services resources in the project repositories with the components in the system model.

### Before you begin

- You must have created components.
- You must be a member of the project that contains the Suites and Tests or virtual service resources.


1. Click **Projects**.

The **Projects** page is displayed.

2. Click the project that contains the Suites and Tests or virtual service resources that you want to associate with components.
3. Click **System Model**.
4. Click the component that you want to associate with the Suites and Tests or virtual services resources.

The component details are displayed in the right pane.

5. Click the **Associate** tab.

The associated resources are displayed. To associate more resources, click the **Add association** icon  next to the resource type.



**Note:** Click the **Add** button to associate resources with the component for the first time.

To associate a resource with a component for the first time, you must perform the following steps:

- a. Click **Add**.

A drop-down list is displayed.

- b. Select one of the following resource types:

- **Suites and Tests:** Click this option if you want to associate your component with the different types of test suites.
- **Virtual Services:** Click this option if you want to associate your component with the virtual service resources.

The **Associate Assets** page is displayed.




**Notes:**

- You can view all the available resources of Suites and Tests or virtual services by selecting the **Suites and Tests** and **Virtual Resources** check boxes.
- You can use the **Search** field to search for virtual service resources by entering the name of the Suites and tests or virtual service resources.

- c. Select one or more resources that you want to associate, and then click **Add**.

The added resources are displayed in the **Associate** and **Overview** tabs of the component.

6. **Optional:** Associate more resources by clicking the **Add association** icon  next to the resource type.

## Results

You have associated components with the Suites and Tests or virtual service resources.

## Viewing resources associated with components

After you associate Suites and Tests or virtual services resources with components in the system model, you can view the associated resources from the **System Model** page. You can opt to start an instance of the associated resource or stop a running instance of the associated resource.

### Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See [Managing access to server projects on page 588](#).
- Created and published the system model. See [Creating a system model on page 251](#).
- Created components in the system model. See [Creating components on page 252](#).
- Associated resources in a project with components in the system model. See [Associating resources with components on page 262](#).

1. Log in to HCL OneTest™ Server.
2. Open the team space to which a team space repository is added.

The **Active projects** page is displayed.

3. Click **Active projects > My projects > project\_name** to open the project that contains the test assets.

The **Overview** page of the project is displayed.

4. Click **System Model** on the left navigation pane.

The **System Model** page is displayed.




5. Click the component that you associated with the resources in the project repository.





The resources associated with the component are displayed under the resource type in the **Associate** tab.

6. Click the **Overview** tab, if it is not already displayed.






The resources that are associated with the component are displayed under the **Suites and Tests** or **Virtual Services** sections.





7. View and perform the actions for the resource type that you want from the following options:
  - For **Suites and Tests** resources, go to [8 on page 264](#).
  - For **Virtual Services** resources, go to [9 on page 265](#).
8. Identify the test resource that are listed under the **Suites and Tests** section, and perform the action that you want described in the following table:



Task	Description	Action
<b>Collapse</b> icon 	You can collapse the resource sections by clicking the <b>Collapse</b> icon  .   <b>Note:</b> When the section is collapsed, you can view the number that is displayed of the re-	Click the icon to toggle between the collapsed and expanded display of the <b>Suites and Tests</b> section.

Task	Description	Action
	 sources that are associated.	
<b>Type</b>	You can view the type of the Suite or test.	Hover over the Suites and Tests icon under the <b>Type</b> column to view the type of Suite or test.
<b>Name</b>	You can view the name of the associated Suite or test and the path in the repository.	There is no action to perform.
<b>Status</b>	You can view the details of the test runs.	Click each of the squares to view the details of the specific test run. For example, if the status of a test run is represented as shown in the image  , then it indicates that the Suite or test has run only two times. After you click any colored square, the <b>Results</b> page is displayed.
<b>Action</b>	You can start the execution of the Suite or test from the <b>Execution</b> page when you click the <b>Show in execution page</b> icon  .	Click the <b>Show in execution page</b> icon  to view the Suite or test on the <b>Execution</b> page.  The Suite or test is displayed on the <b>Execution</b> page.

9. Identify the virtual service resource that are listed under the **Virtual Services** section, and perform the action that you want described in the following table:

Field	Description	Action
<b>Collapse</b> icon 	You can collapse the resource sections by clicking the <b>Collapse</b> icon  .	Click the icon to toggle between the collapsed and expanded display of the <b>Virtual Services</b> section.
	 <b>Note:</b> When the section is collapsed, you can view the number that is displayed of the resources that are associated.	
<b>Show all</b> icon 	You can opt to view the virtual services that are associated with the component either on the <b>Resources</b> or <b>Instances</b> page.	Perform the following steps: a. Click the <b>Show all</b> icon  .
		b. Select any of the following actions:

Field	Description	Action
	<p>You can start an instance of the virtual service from the <b>Resources</b> page while you can stop a running instance from the <b>Instances</b> page.</p>	<ul style="list-style-type: none"> <li>▪ To start an instance of the virtual service:               <ol style="list-style-type: none"> <li>i. Select <b>Show all in resources page</b> to go to the <b>Resources</b> page.</li> <li>ii. Click the <b>Execute</b> icon  in the row of the virtual service.</li> <li>iii. Configure the settings for the run of the virtual service, if you want to change any of the settings, else click <b>Execute</b>.</li> </ol> <p>The virtual service starts to run.</p> </li> <li>▪ To stop a running instance of the virtual service:               <ol style="list-style-type: none"> <li>i. Select <b>Show all in instances page</b> to go to the <b>Instances</b> page.</li> <li>ii. Identify the instance that you want to stop and click the <b>Stop</b> icon  in the <b>Actions</b> column of the selected virtual service instance.</li> <li>iii. Click <b>Ok</b> in the <b>Stop virtual service instance</b> dialog that is displayed.</li> </ol> <p>The running virtual service instance is stopped.</p> </li> </ul>
<b>Name</b>	<p>You can view the name of the associated virtual service and the path in the repository.</p>	<p>There is no action to perform.</p>
<b>Instances</b>	<p>You can view the state of the running instance of the virtual service resource. The state is displayed as <code>Running</code> only if the instance is running.</p>	<p>There is no action to perform.</p>
<b>Action</b>	<p>You can go to the <b>Resources</b> page to view the virtual service resource when you click the <b>Show resource</b> icon .</p>	<p>Click the <b>Show resource</b> icon  to view the virtual service on the <b>Resources</b> page.</p> <p>The virtual service resource is displayed on the <b>Resources</b> page.</p>

Field	Description	Action
	You can start an instance of the virtual service from the <b>Resources</b> page.	
<b>Expand icon</b> 	<p>You can expand the virtual service card only if the virtual service is a running instance.</p> <p>In the expanded card, you can view the details such as the Environment, the total number of requests received by the virtual service, and the relative time from when the virtual service was started.</p> <p>You can stop a running instance of the virtual service from the <b>Instances</b> page.</p>	<p>Perform the following steps:</p> <p>a. Expand the virtual service card for a running instance.</p> <p>You can view the details of the instance.</p> <p>b. Click the <b>Show instance</b> icon  to go to the <b>Instances</b> page to view the running instance of the virtual service.</p> <p>The running instance of the virtual service is displayed on the <b>Instances</b> page.</p>

### Results

You have viewed the resources that are associated with a component from the **System Model** page.

## Removing resources associated with components

When you do not want to retain the resource that you associated with a component, you can delete the resource in the system model.

### Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See [Managing access to server projects on page 588](#).
- Created and published the system model. See [Creating a system model on page 251](#).
- Created components in the system model. See [Creating components on page 252](#).
- Associated resources in a project with components in the system model. See [Associating resources with components on page 262](#).

1. Log in to HCL OneTest™ Server.
2. Open the team space to which a team space repository is added.

The **Active projects** page is displayed.

3. Click **Active projects > My projects > *project\_name*** to open the project that contains the test assets.

The **Overview** page of the project is displayed.

4. Click **System Model** on the left navigation pane.

The **System Model** page is displayed.

5. Click the component that you associated with the resources in the project repository.

The resources associated with the component are displayed under the resource type in the **Associate** tab.

6. Select the resource that you want to remove from the component by clicking the checkbox of the resource.

You can select any or all resources for removal.

7. Click the **Delete selected** icon  inline with the resource type.

The **Disassociate...** dialog is displayed for the type of resource that you selected. For example, the **Disassociate Suites and Tests** dialog is displayed when you selected the test assets.

8. Click **Delete**.

For the changes to be reflected in the system model for all members of the project, you must publish the changes to the project repository.

9. Open the **Changes** page by clicking **Author > Changes**.

10. Verify the changes listed and then click **Publish changes**.

11. Enter a commit message in the **Description of change** field, and then click **Publish**.

## Results

You have removed resources that were associated with components in the system model.

## Publishing changes of associating resources with components

After you associate resources in a project with a component in the system model, you must publish the changes made to the branch in the project repository, if you want the associations to be visible to the other members in the project.

### Before you begin

You must have completed the following tasks:

- Created and published the system model. See [Creating a system model on page 251](#).
- Created components in the system model. See [Creating components on page 252](#).
- Associated resources in a project with components in the system model. See [Associating resources with components on page 262](#).

1. Log in to HCL OneTest™ Server.
2. Open the team space to which a team space repository is added.



The **Active projects** page is displayed.

3. Click **Active projects > My projects > project\_name** to open the project that contains the test assets.

The **Overview** page of the project is displayed.

4. Open the **Changes** page by clicking **Author > Changes**.

The **Changes** page is displayed. You can find the following information about the changes that you made to the system model when you associated resources with components:

- The resources that you associated with components are listed under the **Asset name** column.
  - The type of change is listed under the **Changes** column.
  - The name of the member who made the change is displayed under the **Last changed by** column.
  - The relative time when the change was made is displayed under the **Last updated** column.
  - The number of changes that the *Edit* branch is ahead of from the project repository branch is indicated by the **commits ahead** option. You can view the details by clicking the **commits ahead** option.
  - The number of changes that the *Edit* branch is behind from the project repository branch is indicated by the **commits behind** option. The *Edit* branch can be behind the project repository branch if another member has published or committed other changes in the system model to the same project repository branch. You can view the details by clicking the **commits behind** option.
5. Click the **Publish changes** option.



**Note:** If you do not want to commit the changes you made to the system model, you can select the **Discard changes** option.

6. Enter a commit message in the **Description of change** field, and then click **Publish**.

## Results

You have published the changes that you made in the system model to the repository in the project. Other members of the team space can now view and use the system model.

## Viewing the system model

You can view an existing system model or you can create a system model if a system model does not exist on the **System Model** page.

### Before you begin

You must be a licensed user.

1. Log in to HCL OneTest™ Server.
2. Click **System Model**.

The **System Model** page is displayed.

3. Perform any of the following actions:

- Add a team space repository, if no repository is configured as the team space repository.
- Create a system model, if no model is created or exists.
- View the system model that exists.
- Modify the existing system model.

### Results

You have viewed an existing system model from the **System Model** page.

## Deleting a system model



When you want to remove an existing system model to create a new system model, you can delete the existing system model.

### Before you begin

You must have ensured that a system model exists in the **Initial Team Space**.

1. Log in to HCL OneTest™ Server.
2. Click **System Model**.

The **System Model** page is displayed.

3. Click the **Open action menu** icon  in the right-pane.
4. Click the **Delete** icon .

The **Delete System Model** dialog is displayed.

5. Click **Delete**.

### Results

You have deleted the system model from the team space repository.

### What to do next

You must publish the changes to the team space repository. See [Publishing changes to the system model on page 261](#).

## Deleting a repository that is added to a team space

You can delete a repository that you added to the team space when you do not want to use a team space repository.


### Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Team Space Owner* or an *Architect* in the team space. See [Managing members and their roles in a team space on page 576](#).
- Added a repository to the team space. See [Adding a repository to a team space on page 248](#).

1. Log in to HCL OneTest™ Server.
2. Open the team space to which a team space repository is added.


The **Active projects** page is displayed.

3. Click the **Settings** icon  from the left navigation pane, and then select **Manage team space**.

The **Details** page is displayed.

4. Click **Repository** on the left navigation pane.

The **Repository** page is displayed.


5. Click the **Open action menu** icon .
6. Click **Delete** in the action menu list.

The **Delete repository** dialog is displayed.

7. Select the **I understand...** checkbox, and then click **Delete**.

## Results

You have deleted the team space repository.

 **Restriction:** When you delete the team space repository, the system model if created is also deleted, and you or other members of the team space cannot add a system model.

## Prerequisites to running tests

Before you can configure and run a test in a project on HCL OneTest™ Server, you must read the information about the different tests. You might want to add a remote static agent or a remote Docker host to the project as an alternate location to run the tests.

You can find the following information about the prerequisite tasks:

- [Test run considerations for AFT Suites on page 272](#)
- [Test run considerations for API Suites on page 273](#)
- [Test run considerations for schedules on page 276](#)
- [Considerations for using Jaeger traces in reports on page 277](#)
- [Test run considerations for JMeter tests on page 278](#)
- [Test run considerations for JUnit tests on page 279](#)
- [Test run considerations for Postman tests on page 280](#)
- [Management of agents on page 282](#)
  - [Test run considerations for running tests on remote agents on page 282](#)
- [Management of Docker hosts on page 288](#)
  - [Test run considerations for running tests on remote Docker hosts on page 289](#)

Related information

Tests configurations and test runs

## Test run considerations for AFT Suites

Before you configure an AFT Suite run, you must read the considerations you must take into account.

When you want to run an AFT Suite from a project on HCL OneTest™ Server, you can check if the AFT Suite meets any of the following conditions:

If...	Then...
The AFT Suite uses the settings configured in an AFT XML file for a remote agent	<p>You must ensure that you have configured the location element in the AFT XML to point to a remote agent.</p> <p>For more information about setting the location element, refer to <a href="#">Using an XML file to run multiple Web UI tests and compound tests simultaneously</a>.</p>
You want to run an AFT Suite on a remote agent	<p>You must run the test on the remote agent before you commit the test asset to the remote repository.</p> <p>When you run the test on the remote agent, the following events occur:</p> <ul style="list-style-type: none"> <li>• The agent is added as the location in the AFT XML file, on which the test is to be run.</li> <li>• The agent is displayed as the agent on which the test is to be run, under the <b>Host</b> column within the <b>Location</b> tab in the <b>Execute test asset</b> dialog box.</li> </ul>
You commit the test asset to the remote repository without running the test on the remote agent	The agent is not displayed under the <b>Host</b> column within the <b>Location</b> tab in the <b>Execute test asset</b> dialog box.

Before you configure the run for an AFT Suite, you must complete the following tasks, if they are applicable:

- Add the remote agents, on which the test is to be run, to the project on HCL OneTest™ Server. See [Adding an agent to a project](#).
- After you add the agent, you can select the agent configured in the AFT XML file as the location for the test run or select the remote agents that you add to the project, when you configure a run for an AFT Suite.

The remote agents are shown as the agents available for selection under the **Override** column within the **Location** tab in the **Execute test asset** dialog box.

You can then select the remote agent as the location to run the AFT Suite when you are configuring the test run. Alternatively, you can enter the argument `-swaplocation <configured_agent_location>:<overriding_agent_location>` in the **Program Arguments** field in the **Advanced Setting** dialog box when you are configuring the test run.

For running an AFT Suite, see [Configuring an AFT Suite run on page 306](#).

---

Related information

Tests configurations and test runs

## Test run considerations for API Suites

Before you configure an API Suite run, you must read the considerations you must take into account.

You can find the following information about API Suites:

- [Important information about API Suites on page 273](#)
- [Important information about report configuration in API Suites on page 275](#)

### Important information about API Suites

When you want to run an API Suite from a project on HCL OneTest™ Server and the test suite in HCL OneTest™ API meets any of the following conditions:

- The test suite refers to local stubs that use a transport other than HTTP or MQ.
- The test suite refers to local stubs that use the HTTP transport.
- The transport used in the tests in the test suite is configured with the host name set to *localhost* for the HTTP/TCP proxy.
- The test suite has tests that use a transport and the transport requires third-party application `JAR` files for a successful run.

You must then refer to the following table for the next steps:

If...	Then...
The tests in an API Suite refer to local stubs that use a transport other than HTTP or MQ.	You must perform any of the following actions before you commit the test asset to the remote repository:

If...	Then...
	<ul style="list-style-type: none"> <li>• Remove the reference to the local stub from the test suite.</li> <li>• Publish the stubs to HCL® Quality Server.</li> </ul> <p>To publish and edit stubs, see <a href="#">Publishing stubs</a>.</p>
<p>The tests in an API Suite refer to local stubs that use the HTTP transport.</p>	<p>You must perform the following actions before you commit the test asset to the remote repository:</p> <ol style="list-style-type: none"> <li>1. Create a test suite that contains the stub and tests that run on the stub.</li> <li>2. Configure the physical HTTP transport with the following parameters in the <b>Client</b> tab in the <b>Physical View</b>: <ul style="list-style-type: none"> <li>◦ Set the <code>Host name</code> or <code>IP address</code> of the <code>HTTP Proxy server</code> in the <b>Proxy Host</b> field.</li> <li>◦ Set <code>3128</code> as the proxy server port in the <b>Proxy Port</b> field.</li> </ul> </li> <li>3. Set the reference to the stub for the tests in the test suite scenario from the <b>Scenario Editor</b> dialog box from the <b>Test Factory</b> view by selecting the <i>Live system</i> option in the <b>Satisfied by</b> column to run the stubs on the HTTP proxy registered with HCL OneTest™ Server.</li> <li>4. Save the project.</li> </ol>
<p>The transport used in the tests in an API Suite is configured with the host name set to <i>localhost</i> for the HTTP/TCP proxy.</p>	<p>You must replace the host name with <i>fully-qualified-domain-name</i> or <i>IPAddress</i> of the proxy host in the tests before you commit the test asset to the remote repository.</p>
<p>You plan to run the API Suite on the computer where HCL OneTest™ Server is installed and the API Suite contains tests that use a transport from any of the following third-party applications:</p> <ul style="list-style-type: none"> <li>• Camel</li> <li>• CentraSite</li> <li>• CICS</li> <li>• Coherence</li> <li>• Database</li> <li>• IMS</li> </ul>	<p>You must complete the following tasks:</p> <ol style="list-style-type: none"> <li>1. Identify the location of the third-party application <code>JAR</code> files. You can use Library Manager to know the location where the third-party application <code>JAR</code> files are saved.</li> <li>2. Copy the third-party application <code>JAR</code> files to the computer where HCL OneTest™ Server is installed.</li> </ol>

If...	Then...
<ul style="list-style-type: none"> <li>• Integra</li> <li>• JMS</li> <li>• SAP RFC</li> <li>• Software AG Universal Messaging</li> <li>• TIBCO EMS</li> <li>• TIBCO Rendezvous</li> <li>• TIBCO SmartSockets</li> <li>• WebSphere Application Server Service Integration Bus (SiBus)</li> <li>• WebLogic</li> <li>• Software AG webMethods</li> <li>• WebSphere MQ</li> </ul>	<p>See <a href="#">Copying third-party application Jars to Kubernetes on page 119</a>.</p>
<p>You plan to run the API Suite on a remote Docker host and the API Suite contains tests that use a transport from any of the following third-party applications:</p> <ul style="list-style-type: none"> <li>• Camel</li> <li>• CentraSite</li> <li>• CICS</li> <li>• Coherence</li> <li>• Database</li> <li>• IMS</li> <li>• Integra</li> <li>• JMS</li> <li>• SAP RFC</li> <li>• Software AG Universal Messaging</li> <li>• TIBCO EMS</li> <li>• TIBCO Rendezvous</li> <li>• TIBCO SmartSockets</li> <li>• WebSphere Application Server Service Integration Bus (SiBus)</li> <li>• WebLogic</li> <li>• Software AG webMethods</li> <li>• WebSphere MQ</li> </ul>	<p>You must complete the following tasks:</p> <ol style="list-style-type: none"> <li>1. Identify the location of the third-party application JAR files. You can use Library Manager to know the location where the third-party application JAR files are saved.</li> <li>2. Copy the third-party application JAR files to the computer where the remote Docker host is installed.</li> </ol> <p>See <a href="#">Copying third-party application Jars to a remote Docker host on page 294</a>.</p>

### Important information about report configuration in API Suites

You must configure test suites in HCL OneTest™ API to use a results database so that the details of the test results can be captured and displayed in the HCL OneTest™ Server API results reports. See [Configuring the project results database](#).

If your server projects use either a Microsoft SQL Server, MySQL, or both as a project results database, you must copy the Microsoft SQL Server and MySQL database JAR files to the JDBC folder in the path `/data/JDBC` on HCL OneTest™ Server. See [Copying third-party application Jars to Kubernetes on page 119](#).

The database JAR files that you need are based on the version of Microsoft SQL Server or MSQL that you use with HCL OneTest™ API. See [Adding Microsoft SQL Server and MySQL drivers](#).

To run an API Suite, see [Configuring an API Suite run on page 314](#).

---

Related information

[Scenario reference setting](#)

Tests configurations and test runs

## Test run considerations for schedules

Before you configure a Rate Schedule or VU Schedule run, you must read the considerations that you must take into account.

### Important information about JMeter tests in schedules

When you want to run JMeter tests as part of the VU Schedule or Rate Schedule on HCL OneTest™ Server, you must have completed the following tasks:

- Installed the HCL OneTest™ Performance Agent on a computer.
- Installed the JMeter application on the computer where you have installed the HCL OneTest™ Performance Agent.
- Set the environment variable **JMETER\_HOME** that points to the JMeter installation directory.

### Important information about Resource Monitoring sources in schedules

When you want to run schedules on HCL OneTest™ Server, you must have completed the following tasks in HCL OneTest™ Performance:

- Created a performance schedule of **Rate** or **VU** type.
- Added Resource Monitoring sources, or the labels of Resource Monitoring sources to a schedule. Refer to [Adding resource monitoring sources to a performance schedule by using labels](#).

Before you configure a run of a Schedule, you must complete the following tasks in HCL OneTest™ Server:

- Add Resource Monitoring sources to your project in HCL OneTest™ Server. See [Monitoring host resources on page 450](#).
- Enable HCL OneTest™ Server as Resource Monitoring service only if you want to replace Resource Monitoring labels set in HCL OneTest™ Performance with the labels that you created in HCL OneTest™ Server. See [Controlling resource monitoring sources in a schedule on page 474](#).




- Perform the following steps when you want to override the Resource Monitoring labels in the test asset with the Resource Monitoring labels that you create in HCL OneTest™ Server:

1. Click the **Resource Monitoring** tab.

**Note:**

The **Resource Monitoring** tab displays only if you enabled the **Resource Monitoring from Service** option in HCL OneTest™ Performance when you created the test asset.

2. Click , and press the `Ctrl + Space bar` keys, or enter the initial letter of a label to select a label in the list.

**Note:**

You can select or add labels only if you added the labels in the **Resource Monitoring Sources** page in HCL OneTest™ Server.

For running a Rate Schedule or VU Schedule, see [Configuring a run of a Rate Schedule or VU Schedule on page 368](#).

---

Related information

Tests configurations and test runs

## Considerations for using Jaeger traces in reports

Before you can use Jaeger to report the test results in the supported tests, you must read the considerations that you must take into account.

When you want to enable Suites, Tests, or Schedules that are run from HCL OneTest™ Server to use Jaeger traces for reporting the test results, you must have completed the following tasks:

- Installed Jaeger at the time of installation of the server software. See [Prerequisites for installing the server software on Red Hat OpenShift on page 37](#) or [Prerequisites for installing the server software on Ubuntu on page 58](#).
- Verified that Jaeger is enabled and working.

After you install and verify that Jaeger is enabled, and when you want Jaeger to report the test results for the tests that you configure for a run, you must perform the following steps:

1. Click **Advanced**.
2. Enter `-history jaeger` in the **Program Arguments** field.

---

Related information

Tests configurations and test runs

## Test run considerations for JMeter tests

Before you can run JMeter tests on HCL OneTest™ Server, you must read the considerations that you must take into account and complete the tasks indicated.

When you want to run JMeter tests that you create in JMeter, you must have completed the following tasks:

- You must ensure that the JMeter extension on HCL OneTest™ Server is enabled at the time of installation of the server software or is running before you add the repository that contains JMeter tests to the server project. See [Prerequisites for installing the server software on Red Hat OpenShift on page 37](#) or [Prerequisites for installing the server software on Ubuntu on page 58](#).
- You must have committed the following JMeter assets or resources as a JMeter project to the remote repository:



**Note:** You must have created a project directory and added the associated tests or assets in sub directories, that might be required for the JMeter tests to run successfully.

- The `jmeterRoot.jprj` project file in the project directory.
- The JMeter tests in a sub directory with a name you provide within the project directory.
- Optionally, the Java Key Store that is required at test run time in the `certs` sub directory.
- Optionally, the JMeter properties files that are required at test run time in the `properties` sub directory.
- Optionally, the library jar files that are required at test run time in the `lib` sub directory.
- Optionally, the test dependencies that are required at test run time in the `deps` sub directory.

For example, the project directory can be *myproject* that contains the `jmeterRoot.jprj` project file. In the project directory, you can create the following sub directories:

- The `mytests` sub directory that contains the JMeter tests.
- Optionally, the `certs` sub directory that contains the Java Key Store.

For example, the path to the key store can be specified in the `system.properties` file as follows:

```
javax.net.ssl.trustStore=$(deploy.home)/certs/simpleStore.jks
```

Where *<simpleStore.jks>* is the Java Key Store and the *<deploy.home>* is the keyword to access the key store at the test run time.

- Optionally, the `properties` sub directory that contains properties files such as `system.properties`, `jmeter.properties`, `reportGenerator.properties`, or `user.properties`.

- Optionally, the `lib` sub directory that contains the **lib** and **lib/ext** content to be used at test run time. For example, the **amq client** and the **plugins** required to parse the **jmx** that contains **amqp** calls.
- Optionally, the `deps` sub directory that contains the test dependencies.

The resulting project directory can be as follows:

```
myproject (that contains the jmeterRoot.jprj)
myproject/mytests
myproject/certs
myproject/properties
myproject/lib
myproject/deps
```

- You must ensure that you use a relative path when JMeter tests refer to other test assets. For example, if the JMeter test that is in the `myproject/mytests` folder refers to a `mycsv.csv` file that is in the `myproject/deps` folder, then the `mycsv.csv` is referred as follows: `../deps/mycsv.csv` in the JMeter test.
- You can create the JMeter project file named as `meterRoot.jprj`.

The project file is a java properties file. The JMeter project file is required so that HCL OneTest™ Server can identify the asset as a JMeter project. You can specify additional information in the project file. You can specify the names of the sub directories created and through keywords you can specify the path in the properties files whether they are relative to the root directory or to the tests sub directory.

For example, if you have created the following sub directories in the project directory called *myproject* that contains the `meterRoot.jprj` project file:

- `myproject/mytests`
- `myproject/properties`

If the JMeter tests are contained in the following path: `myproject/mytests/testplan/test1.jmx`, and some dependencies that are also required to be deployed at test run time exists in the following path: `myproject/mytests/testplan/localdir`. You must specify these file paths in the project file as follows:

```
testDir.localtest.sub.dir=localdir
aSecondRootDir.sub.dir=dir2
athirdRootDir.sub.dir=dir3
```

---

#### Related reference

[Troubleshooting issues on page 987](#)

#### Related information

[Configuring a JMeter test run on page 355](#)

## Test run considerations for JUnit tests

Before you can run JUnit tests on HCL OneTest™ Server, you must read the considerations that you must take into account and complete the tasks indicated.

When you want to run JUnit tests, you must have completed the following tasks:

- You must ensure that the JUnit extension on HCL OneTest™ Server is enabled at the time of installation of the server software or is running before you add the repository that contains the JUnit tests to the server project. See [Prerequisites for installing the server software on Red Hat OpenShift on page 37](#) or [Prerequisites for installing the server software on Ubuntu on page 58](#).
- You must have created the JUnit tests that use JDK 8, as part of a Maven V3.6.3 project. The Maven project must contain the Maven Surefire plugin, which is required to run the JUnit tests contained in the Maven project. The Maven project as a `pom.xml` must be committed to the remote repository.



**Note:** If you create the JUnit tests on other versions of JDK, you must copy the JDK version to the `userlibs` pod under the `data/junit-ext/jdks` directory. See [Copying third-party application Jars to Kubernetes on page 119](#).

---

#### Related reference

[Troubleshooting issues on page 987](#)

#### Related information

[Configuring a JUnit test run on page 359](#)

## Test run considerations for Postman tests

Before you can run Postman tests on HCL OneTest™ Server, you must read the considerations that you must take into account and complete the tasks indicated.

When you want to run Postman tests that you create in Postman, you must have completed the following tasks:

- You must ensure that the Postman extension on HCL OneTest™ Server is enabled at the time of installation of the server software or is running before you add the repository that contains Postman collections to the server project. See [Prerequisites for installing the server software on Red Hat OpenShift on page 37](#) or [Prerequisites for installing the server software on Ubuntu on page 58](#).



**Restriction:** If you add the repository with Postman collections to a server project before you enable the Postman microservice, the Postman collections might not be displayed in the **Execution** page for you.

- You must have exported the following resources from Postman and saved the resources on your computer:
  - Collections and their associated variables and environments from Postman in which all collections are exported as a single file.



**Note:** You can export as many collections as you want.

For more information, refer to [Exporting a collection from Postman](#) and [Exporting data dumps](#).

- Environments configured in Postman, if you are exporting only the environments as separate JSON files.

For more information, refer to [Exporting environments from Postman](#).

- You must have committed the following Postman assets or resources to the remote repository:
  - Collections and their associated variables and environments that you exported. You can export all resources that are in your workspace in Postman. You can also export the collections along with the variables as a single file and the environments separately as another file.

For example, you can save the exported file as `MyCollectionsVarEnv.json` that contains the collections, variables, and environments or you can export only the collections and variables as `MyCollectionsVar.json`.

- Environments that you exported, if you are exporting only the environments as separate JSON files. For example, you can save the exported file as `MyEnv.json`.
- An empty text file with the `.postman` extension that is contained in the same directory as the other Postman resources, so that the contents in the directory are attributed as Postman resources. For example, you can create `MyProject.postman`.



**Note:** The `.postman` file enables HCL OneTest™ Server to identify the test resources as Postman resources.

For example, if you have saved the Postman resources that you downloaded from Postman as follows:

- `MyCollectionsVarEnv.json` or `MyCollectionsVar.json` that contains the Postman collections.
- `MyEnv.json` that contains the environments associated with the Postman collection.

The project directory can be as follows:

```
/Project
/Project/MyProject.postman
/Project/MyCollectionsVarEnv.json
```

Alternatively, the directory can be as follows:

```
/Project
/Project/MyProject.postman
/Project/MyCollectionsVar.json
/Project/MyEnv.json
```

---

#### Related reference

[Troubleshooting issues on page 987](#)

#### Related information

[Configuring a Postman test run on page 364](#)

## Management of agents

You can find information about managing the agents that you must install on remote computers and configure the agents to register with HCL OneTest™ Server. After the agents are registered, you can add them to your projects and then select them as locations to run tests.

When you plan to run any of the tests on a remote agent from a project on HCL OneTest™ Server, you must have completed the following tasks:

- Installed the agent of the same version as the version of HCL OneTest™ Server.
- Configured the agent with your offline user token so that the agent can connect and also register with HCL OneTest™ Server.



**Note:** When you configure the agent to connect to the server by using the offline user token that you generated on HCL OneTest™ Server, you become the owner of the agent.

- Verified that you have write permissions to the `majordomo.config` file so that certain transactions can be written by HCL OneTest™ Server to the `majordomo.config` file.



**Note:** You can verify if you have the *Read* and *Write* permissions by checking the properties of the `majordomo.config` file.

- Viewed the agents that are registered with HCL OneTest™ Server. See [Viewing agents that are registered with HCL OneTest Server on page 283](#).
- Added the remote agents that are registered with HCL OneTest™ Server to your project on HCL OneTest™ Server. See [Adding an agent to a project](#).



**Note:** After you add the remote agents to your project, HCL OneTest™ Server determines the capabilities that are provided by the remote agent and displays the details and capabilities of the agent on the **Infrastructure** page. You can also define the capabilities that you want to add, and then view, edit, or delete the capabilities that you defined for the agents in your project. See [Working with agent capabilities on page 286](#).

---

### Related information

[Agent installation](#)

## Test run considerations for running tests on remote agents

Before you configure a test to run on a remote agent from HCL OneTest™ Server, you must read the considerations that you must take into account.

## Supported tests

You can run the following tests that are supported to be run on a remote agent:

- AFT Suite
- Compound Tests that contains performance tests
- Compound Tests that contains Web UI tests
- Compound Tests that contains traditional HTML tests
- Rate Schedule
- VU Schedule

## Prerequisite tasks

Before you can run any of the supported tests on a remote agent from a project on HCL OneTest™ Server, you must have completed the following tasks:

- Installed the agent on the remote computer. See [Management of agents on page 282](#).
- Configured the agent with your offline user token so that the agent can connect to HCL OneTest™ Server.
- Added the remote agents to your project on HCL OneTest™ Server. See [Adding an agent to a project](#).



**Note:** If you want to run an AFT Suite (that contains Web UI tests) on an agent, you must start the agent as a non-admin user so that certain browsers that are configured start correctly during the test run time.

Before you add the tests to the remote repository, you must have completed the following tasks:

- Associated the agents with the performance schedule in HCL OneTest™ Performance.
- Specified the host name of the remote workstation in the AFT XML file.

You can continue to configure a run for any of the following supported test types:

- [Configuring an AFT Suite run on page 306](#)
- [Configuring a run of a Compound Test that contains performance tests on page 347](#)
- [Configuring a run of a Compound Test that contains Web UI tests on page 330](#)
- [Configuring a run of a Compound Test that contains HTML tests on page 322](#)
- [Configuring a run of a Rate Schedule or VU Schedule on page 368](#)

## Viewing agents that are registered with HCL OneTest™ Server

After you install the remote agents and configure them with your offline user token, the agents register with HCL OneTest™ Server and you can view the registered agents on the **Agents and Intercepts** page.

### Before you begin

You must have completed the following tasks:

- Installed the agent of the same version as the version of HCL OneTest™ Server.
- Configured the agent with your offline user token so that the agent can register with HCL OneTest™ Server.



**Note:** When you configure the agent to connect to the server, you must specify the offline user token that you generated on HCL OneTest™ Server. This token defines your ownership over the agent.

1. Log in to HCL OneTest™ Server.
2. Click **Infrastructure > Agents and Intercepts** in the navigation pane.

The **Agents and Intercepts** page is displayed.

You can view the agents, intercepts or Dockers that are registered with HCL OneTest™ Server.

3. You can view the agents that you own in any of the following ways:
  - Search for the agent by entering the name of the agent in the **Search** field.



**Note:** You can enter either the full name or any text that is in the name. The search is enabled for case sensitive text that you can enter.

- Sort the **Type** column to sort the items and display the agents in the rows at the top of the table.
- Sort the **Agents** column and identify the agents by their names or by their owner.

You can view the following details about the agents that are registered with HCL OneTest™ Server:

- The projects to which the agent is added, are displayed in the **Projects** column.
- The status of the agent is displayed in the **Status** column.



**Note:** You can add agents that are in the `Ready` state to your project, and then use them as locations to run tests.

## Results

You have viewed the registered agents from the **Agents and Intercepts** page.

## What to do next

You can add the agents that you own to your project. See [Adding an agent to a project](#).

## Adding an agent to a project

If you have performance tests that distribute workload across different workstations or Accelerated Functional Testing (AFT) Suites that need to run on different platforms and browsers, you can add HCL OneTest™ Performance Agents to your projects to run tests on remote workstations.

## Before you begin

You must have completed the following tasks:



- Read about the considerations that you must take into account before you configure a test run to run on a remote agent. See [Test run considerations for running tests on remote agents on page 282](#).
- Been assigned a *Member* or *Project Creator* role in a team space.
- Been assigned the *Owner* or *Tester* role in the project to which you want to add the agent.
- Owned at least one agent that is not added to the project.

### About this task

Only you can add or remove this agent from the project. However, any member of your project with the permission to run the tests can initiate the run of the performance schedule and AFT Suites that are associated with the agent.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project, and then click **Manage > Infrastructure**.

The **Infrastructure** page is displayed.

3. Click **Add > Add agent**.

The agents that you configured with your user token are displayed.

4. Select the agents that you want to use and click **Add**.

The agents that you added are displayed.

5. Perform the following steps to view details and capabilities of the agents:
  - a. Expand the agent to view the details panel.

You can find the following details of the agent displayed:

Parameter	Description
Version	The version of the installed agent.
Address	The IP address of the computer on which the agent is installed.
License type	The license type that is enabled for you to use the agent.
Load generation service	The status of the agent whether it can be used for load generation service.
Last contact	The time of the last contact with the agent by HCL OneTest™ Server.

- b. Expand the capabilities panel to view the system and application capabilities.

You can find the following capabilities of the agent displayed:

Capabilities	Description
System capabilities	Architecture The architecture of the CPU processor of the computer on which the agent is installed.

Capabilities		Description
	MAC Address	The MAC address of the computer on which the agent is installed.
	Operating system	The operating system of the computer on which the agent is installed.
	RAM	The memory that is allotted to RAM in the computer on which the agent is installed.
Application capabilities	Browser	The browsers along with their version that are installed on the computer on which the agent is installed.
User Defined capabilities	User Defined	The capabilities that you add for an agent. You can also tag an agent so that you can identify the agent to select as a location.

c. Check for the status of the agent from the **Status** column.



**Note:** You can add agents that are in any state to the project.

## Results

You have added remote agents to your project and viewed the agent capabilities displayed for each agent that you own.

## What to do next

You can perform any of the following tasks:

- Run the tests on the agent that is added to the project from the **Execution** page. See [Test run configurations on page 305](#).
- Define the capabilities that you want to add, and then view, edit, or delete the capabilities that you defined for the agents in your project. See [Working with agent capabilities on page 286](#).

## Working with agent capabilities

When you installed static agents V10.1.2 or later, and added them to your project, you can view capabilities of the agents from the **Infrastructure** page. When you installed static agents V10.2.0 or later, you can add, view, edit, or delete properties and capabilities of agents from the **User Defined** panel of that agent.

## Before you begin

You must have completed the following tasks:

- Been assigned a *Member* or *Project Creator* role in a team space.
- Been assigned the *Owner* or *Tester* role in the project to which you want to add the agent.
- Owned at least one agent that is not added to the project.

- Installed the agent V10.2.0. See [Test run considerations for running tests on remote agents on page 282](#).
- Added agents to your project. See [Adding an agent to a project](#).

1. Log in to HCL OneTest™ Server and open the team space.
2. Open your project, and then click **Manage > Infrastructure**.

The agents that you added to your project are displayed on the **Infrastructure** page.




3. Expand the agent.




The **Details** and **Capabilities** tabs are displayed.

4. Click **Capabilities** to view the capabilities of the agent.
5. Click **User Defined**.

The **User Defined Capabilities** dialog box is displayed.

6. Complete the steps for the task that you want to perform as listed in the following table:

Task	Action
Adding capabilities	<p>Perform the following actions:</p> <ol style="list-style-type: none"> <li>a. Click <b>Add new</b>.</li> <li>b. Enter the name of the capability in the <b>Capability Name</b> field.</li> <li>c. Perform any of the following actions in the <b>User Defined Category</b> field: <ul style="list-style-type: none"> <li>▪ Select an existing category from the list.</li> <li>▪ Enter a name of the category for the capability, if the category does not exist.</li> </ul> <p> <b>Note:</b> You can also click <b>Add new</b> to add a category.</p> <p>The category for the capability is either selected or created and the fields for the property name and value are displayed.</p> <p> <b>Note:</b> The property for a capability is a <i>name-value</i> pair.</p> </li> <li>d. Enter a name for the property in the <b>Property Name</b> field and its corresponding value in the <b>Property Value</b> field for the capability. <p> <b>Note:</b> You can add multiple properties for the same capability under the same category or under different categories by clicking <b>Add new property</b>, and then by entering the name and value for the property.</p> </li> <li>e. Click <b>Save</b>.</li> </ol> <p>The capabilities that you added are displayed.</p>

Task	Action
Viewing capabilities that you added	The <b>User Defined Capabilities</b> panel displays the capabilities that you added to the agent.
Editing capabilities that you added	<p>Perform the following actions:</p> <ol style="list-style-type: none"> <li>Click the <b>Edit</b> icon  in the row of the capability that you want to edit.</li> </ol> <p>The user-defined capability is enabled for editing. You can edit the name of the capability, or the property name or property value.</p> <ol style="list-style-type: none"> <li>Edit any of the items that you want for the user-defined capability.</li> <li>Click <b>Update</b>.</li> </ol> <p>The updated capability of the agent is displayed.</p>
Deleting capabilities that you added	<ol style="list-style-type: none"> <li>Click the <b>Delete</b> icon  in the row of the capability that you want to delete.</li> <li>Click <b>Delete</b>.</li> </ol> <p>The capability that you selected is deleted along with all the properties that you added for the capability.</p>
Deleting a property defined for a capability	<ol style="list-style-type: none"> <li>Click the <b>Edit</b> icon  in the row of the capability that you want to edit and remove a defined property.</li> </ol> <p>The user-defined capability is enabled for editing.</p> <ol style="list-style-type: none"> <li>Select and delete the contents in the <b>Property Name</b> field and its corresponding <b>Property Value</b> field.</li> <li>Click <b>Update</b>.</li> </ol> <p>The property is removed for the capability.</p>

### Results

You have added, viewed, edited, or deleted a property or the user-defined capability of an agent.

### What to do next

You can run the tests on the agent from the **Execution** page. See [Test run configurations on page 305](#).

## Management of Docker hosts

You can find information about the tasks that you can perform on remote Docker hosts. After you set up a remote Docker host, you can register the Docker host with HCL OneTest™ Server. You must add the registered remote Docker host to your project before you can run tests on the remote Docker host location.

You can find the following information about managing Dockers:

- [Test run considerations for running tests on remote Docker hosts on page 289](#)
- [Setting up a remote Docker host computer on page 291](#)
- [Setting up a secure remote Docker host computer on page 292](#)
- [Copying third-party application Jars to a remote Docker host on page 294](#)
- [Registering a remote Docker host on page 296](#)
- [Viewing remote Docker hosts that are registered with HCL OneTest Server on page 298](#)
- [Adding a remote Docker host to the project for running tests on page 299](#)
- [Editing configurations of a remote Docker host on page 300](#)
- [Removing a remote Docker host from a project on page 302](#)
- [Unregistering a remote Docker host from HCL OneTest Server on page 304](#)

## Test run considerations for running tests on remote Docker hosts

Before you can configure a test to run on a remote Docker host, you must read the considerations that you must take into account.

You can run all the tests that are supported on HCL OneTest™ Server on remote Docker hosts.



**Important:** You can run virtual services only in the **Default Cluster** location of HCL OneTest™ Server. You cannot run virtual services on a remote Docker host.

When you plan to run any of the tests on a remote Docker host from a project on HCL OneTest™ Server, you must complete the following tasks:

	Tasks	More information...
1.1	Set up non-secure remote Docker hosts.	<a href="#">Setting up a remote Docker host computer on page 291</a>
1.2	Set up secure remote Docker hosts.	<a href="#">Setting up a secure remote Docker host computer on page 292</a>
2	Register the remote Docker host on HCL OneTest™ Server.	<a href="#">Registering a remote Docker host on page 296</a>
3	View the registered Docker hosts.	<a href="#">Viewing remote Docker hosts that are registered with HCL OneTest Server on page 298</a>
4	Add a remote Docker host to the HCL OneTest™ Server project.	<a href="#">Adding a remote Docker host to the project for running tests on page 299</a>

	Tasks	More information...
5	<p>Configure a run for your test that you want to run on the remote Docker host by performing the following actions:</p> <ol style="list-style-type: none"> <li>1. Go to <a href="#">Test run configurations on page 305</a>.</li> <li>2. Check for the prerequisite tasks that you must perform before you configure a test run for the test.</li> <li>3. Select the task for the test that you want to configure for a run, and then follow the steps to run the test.</li> </ol>	<ol style="list-style-type: none"> <li>1. <a href="#">Test run configurations on page 305</a></li> <li>2. <a href="#">Prerequisites to running tests on page 271</a></li> <li>3. <a href="#">Test run configurations on page 305</a></li> </ol>

If you plan to run API Suites that use a transport and the transport requires third-party application `Jar` files for a successful run on a remote Docker host, then you must perform the following task:

- You must copy the third-party application `Jar` files to the third-party application folder on the computer where the remote Docker host is installed. See [Copying third-party application Jars to a remote Docker host on page 294](#).

After you set up and register the remote Docker host with HCL OneTest™ Server, you can add the remote Docker hosts to your project on HCL OneTest™ Server. See [Adding a remote Docker host to the project for running tests on page 299](#).

You can find information about the tasks that you can perform when you use remote Docker hosts that are set up and registered with HCL OneTest™ Server.

- When you configure a test run and in the step in which you select the remote Docker, if you encounter any of the following conditions because these were changed on the remote Docker host:
  - The host name or port of the remote Docker is not displayed correctly.
  - You want to change the mode of authentication with the remote Docker host.

You must edit the configuration settings for the remote Docker host. See [Editing configurations of a remote Docker host on page 300](#).

- When you want to delete a remote Docker host from your HCL OneTest™ Server project, see [Removing a remote Docker host from a project on page 302](#).
- When you want to remove or unregister a remote Docker host from HCL OneTest™ Server, see [Unregistering a remote Docker host from HCL OneTest Server on page 304](#).

Related information

Tests configurations and test runs

[Management of Docker hosts on page 288](#)

## Setting up a remote Docker host computer

You must set up a computer to host the remote Docker host. You can set up the remote host with a non-secure mode of connection with HCL OneTest™ Server.

### Before you begin

You must have identified a remote computer or VM that has a minimum of 16 GB RAM. You must have installed Ubuntu V18.04 or later, on the remote computer.

You must have installed Docker Enterprise Edition for a non-secure mode of connection with HCL OneTest™ Server on the remote Docker host computer. Refer to [Get Docker Engine](#).

You must have the entitlement key that is required for you to connect to the Harbor repository to be able to pull in the software image to the remote Docker host.

### About this task

You must set up a remote computer or a Virtual Machine (VM) on which you want to set up the remote Docker host. You must be a system administrator or get the remote Docker host set up by a system administrator. You must ensure that the system administrator completes the following actions:

- Enables SSH.
  - Creates user credentials for you with root permissions.
  - Provides the name of the Docker container.
  - Provides the IP address of the remote Docker host computer.
1. As a system administrator of the remote system, specify a port to access the Docker daemon in the `docker.service` file located in the directory `/lib/systemd/system`. You must add the configured port at the line in the file:

```
ExecStart=/usr/bin/dockerd -H fd:// -H tcp://0.0.0.0:PORT
```

For example, if the port configured is 4342, then the line must read as:

```
ExecStart=/usr/bin/dockerd -H fd:// -H tcp://0.0.0.0:4342
```

2. Restart the Docker service by using the following commands:

```
$ sudo systemctl daemon-reload
```

```
$ sudo systemctl restart docker
```

3. Perform the following steps as a user with root permissions or ask the system administrator to complete the following steps:

a. Open a terminal and connect to the remote Docker host by using an SSH tool by running the following command:

```
ssh root@<IP_address_of_remote_host>
```

b. Log in to HCL Software's Harbor container registry by running the following command:

```
sudo docker login hclcr.io -u <username> -p <password>
```



**Note:** You must provide the following username and password in the command to log in to the registry:

- Username as email address registered with HCL
- Password as Harbor CLI secret

c. Run the following commands to pull the product images from the registry on to the remote Docker host computer:



**Note:** You must pull in the images with the version which is the same as the version of HCL OneTest™ Server that you use to run the tests.

- `sudo docker pull cp.hclcr.io/ot/hcl-onetest/studio<image_identifier>`
- `sudo docker pull hclcr.io/ot/hcl-onetest/virtualization<image_identifier>`

## Results

You have set up the remote Docker host computer that can be accessed in a non-secure mode.

## What to do next

You can set up a secure remote Docker host computer optionally, that enables a secure connection with HCL OneTest™ Server. See [Setting up a secure remote Docker host computer on page 292](#).

You can register the remote Docker host on HCL OneTest™ Server. See [Registering a remote Docker host on page 296](#).

---

## Setting up a secure remote Docker host computer

You must set up a computer to host the remote Docker host. You can set up the remote host with a secure mode of connection with HCL OneTest™ Server by using certificates issued by the remote host computer that authenticate the connection.

### Before you begin



You must have identified a remote computer or VM that has a minimum of 16 GB RAM. You must have installed Ubuntu V18.04 or later, on the remote computer.

You must have installed Docker Community Edition V18.09 or later. For instructions, refer to [Get Docker Engine](#).



**Note:** For the remote Docker host to be reached in a safe manner through the network, you can enable *TLS* by specifying the **tlsverify** flag and by pointing the **tlscacert** flag of the remote Docker host to a trusted CA certificate. Refer to [Protect the Docker daemon socket](#).

You must have the entitlement key that is required for you to connect to the Harbor repository to be able to pull in the software image to the remote Docker host.

### About this task

You must set up a remote computer or a Virtual Machine (VM) on which you want to set up the remote Docker host. You must be a system administrator or get the remote Docker host set up by a system administrator. You must ensure that the system administrator completes the following actions:

- Enables SSH.
- Creates user credentials for you with root permissions.
- Provides the name of the Docker container.
- Provides the IP address of the remote Docker host computer.

Perform the following steps as a user with root permissions or ask the system administrator to complete the following steps:

- a. Open a terminal and connect to the remote Docker host by using an SSH tool by running the following command:

```
ssh root@<IP_address_of_remote_host>
```

- b. Log in to HCL Software's Harbor container registry by running the following command:

```
sudo docker login hclcr.io -u <username> -p <password>
```



**Note:** You must provide the following username and password in the command to log in to the registry:

- Username as email address registered with HCL
  - Password as Harbor CLI secret
- c. Run the following commands to pull the product images from the registry on to the remote Docker host computer:



**Note:** You must pull in the images with the version which is the same as the version of HCL OneTest™ Server that you use to run the tests.

- `sudo docker pull cp.hclcr.io/ot/hcl-onetest/studio<image_identifier>`
- `sudo docker pull hclcr.io/ot/hcl-onetest/virtualization<image_identifier>`

## Results

You have set up the remote Docker host computer that can be accessed in a secure mode.

## What to do next

You can optionally, set up a non-secure remote Docker host. See [Setting up a remote Docker host computer on page 291](#).

You can register the remote Docker host on HCL OneTest™ Server. See [Registering a remote Docker host on page 296](#).

---

## Copying third-party application Jars to a remote Docker host

You can run API Suites in a project on HCL OneTest™ Server on a remote Docker host. If the API Suites use a transport and the transport requires third-party `Jar` files for a successful run, you must ensure that the third-party application `Jar` files are available at the test run time. To achieve this, you must copy the third-party application `Jar` files to the computer where you have set up the remote Docker host.

### Before you begin

If you want to copy the third-party application `Jar` files to the to the computer where HCL OneTest™ Server is installed on Kubernetes, see [Copying third-party application Jars to Kubernetes on page 119](#).

You must have server administrator privileges.

You must have completed the following tasks:

- Set up and configured the remote Docker host. See [Management of Docker hosts on page 288](#).
- Identified the third-party application `Jar` files that are required and copied the `Jar` files. See [Test run considerations for API Suites on page 273](#).
- Copied the `Jar` files of the third-party application jars to the directory or folder on the remote Docker host from where you can run the docker commands.

### About this task

You can copy the required third-party application `Jar` files to the folder that is specific for the application under the `/myFiles/Userlibs/<application_name>` folder. You need not extract the files.

You can perform this task any time after you have configured the remote Docker host and you plan to run an API Suite on the remote Docker host. The API Suite uses transports and the transports require third-party application `JAR` files to be available at the test run time.

You can get help on the docker commands by running the command: `$ docker help` from the docker command line.

For more information about the docker commands, refer to the [Docker command line documentation](#).

1. Use the following table to find the name of the folder that corresponds to the specific third-party application for the transport used in the API Suite.



**Note:** You must provide the name of the folder listed for the third-party application as the `<application_name>` in the docker command.

**Table 4. Name of the folder for the application**

Application	Name of the folder to use
Camel	Camel
CentraSite	CentraSite
CICS	CICS
Coherence	Coherence
Database	JDBC
IMS	IMS
Integra	Integra
JMS	JMS
SAP RFC	SAP
Software AG Universal Messaging	SoftwareAGUM
TIBCO EMS	TIBCO
TIBCO Rendezvous	
TIBCO SmartSockets	
WebSphere Application Server Service Integration Bus (SiBus)	WAS
WebLogic	WebLogicJMX
Software AG webMethods	webMethods
WebSphere MQ	WMQ

2. Run the following docker command to create or update the volume that contains the application `Jar` files:

```
docker run --rm -v /<anyFolder>/UserLibs/<application_name>:/ulsrc -v userlibs/<application_name>:/uldest
alpine:latest cp -r /ulsrc/. /uldest/UserLibs/<application_name>
```



**Attention:** You must run the docker command when no other tests are running in the Docker host to prevent concurrent access problems.

## Results

You have successfully copied the third-party application `Jar` files to the folder in the remote Docker host.

## What to do next

- You can register the remote Docker host, if you have not already done so. See [Registering a remote Docker host on page 296](#).
- You can configure the API Suite run. See [Configuring an API Suite run on page 314](#).

## Registering a remote Docker host

You must register the remote Docker host with HCL OneTest™ Server before you add the remote Docker host to your project in a team space, which contains the test assets that you want to run on the remote Docker host.

### Before you begin

Depending on the mode of connection with HCL OneTest™ Server, you must have set up the remote Docker host computer either for a secured mode or for an unsecured mode of connection. See [Setting up a remote Docker host computer on page 291](#) or [Setting up a secure remote Docker host computer on page 292](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Click **Infrastructure > Agents and Intercepts** in the navigation pane.

The **Agents and Intercepts** page is displayed.

You can view the agents, intercepts or Dockers that are registered with HCL OneTest™ Server.

3. Click **New docker host**.

The **New docker host** dialog box is displayed.

4. Enter the *host name* and the *port* of the remote Docker host in the format `<host name or IP_address>:<port>` in the **Remote Docker Host** field.

If you do not provide the port that you configured for the remote Docker host, then the default port that is used is as follows:

- Port 2375 for the unsecured mode
  - Port 2376 for the secured mode
5. Choose from any of the following authentication modes to establish the connection between HCL OneTest™ Server and the remote Docker host computer:

- For the secured mode, which is the default mode and the checkbox is displayed as selected.

Perform the following steps:

- Select **Secure mode**, if it is not selected.
- Click the action labels to browse and select the `.pem` files that correspond to `ca_certificate.pem`, `client_certificate.pem`, and `client_key.pem`.



**Note:** You must have generated the `.pem` files on the remote Docker host computer and copied the files to your local drive.

- For the unsecured mode, the **Secure mode** checkbox must not be selected.

Perform the following step:

- Clear the **Secure mode** checkbox, if selected.

- Click **Test connection** to test if a connection is established between HCL OneTest™ Server and the remote host computer.

Any of the following events occur when **Test connection** is clicked:

- On a successful connection, a message is displayed.
- On a failure to connect to the remote host computer, an error message is displayed. You must resolve the error and reattempt to establish a successful connection.



**Important:** The remote Docker host computer must be connected successfully before you register the remote Docker host with HCL OneTest™ Server.

- Click **Register**.

The remote Docker host that you registered is displayed.



**Important:** The remote Docker host must be registered successfully with HCL OneTest™ Server in the team space before you add the remote Docker host to your project in that team space.

- Click **Close** to exit.

## Results

You have registered the remote Docker host in your team space with HCL OneTest™ Server.

## What to do next

You can view the Docker hosts in your team space that are registered with HCL OneTest™ Server. See [Viewing remote Docker hosts that are registered with HCL OneTest Server on page 298](#).

You must add the registered remote Docker host to your project in your team space before you use the remote Docker host as a location to run tests in your project. See [Adding a remote Docker host to the project for running tests on page 299](#).

---

## Viewing remote Docker hosts that are registered with HCL OneTest™ Server

You must register the remote Docker host in your team space with HCL OneTest™ Server before you add the remote Docker host to your project in that team space, which contains the test assets that you want to run on the remote Docker host.

### Before you begin

Depending on the mode of connection with HCL OneTest™ Server, you must have set up the remote Docker host computer either for a secured mode or for an unsecured mode of connection. See [Setting up a remote Docker host computer on page 291](#) or [Setting up a secure remote Docker host computer on page 292](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Click **Infrastructure > Agents and Intercepts** in the navigation pane.

The **Agents and Intercepts** page is displayed.

You can view the agents, intercepts or Dockers that are registered with HCL OneTest™ Server in your team space.

3. You can view the Dockers that you own in any of the following ways:
  - Search for the Docker by entering the name of the Docker host in the **Search** field.



**Note:** You can enter either the full name or any text that is in the name. The search is enabled for case sensitive text that you can enter.

- Sort the **Type** column to sort the items and then identify the Docker by the name displayed.
- Sort the **Agents** column to sort the items and identify the Docker by the owner.

You can view the following details about the Dockers that are registered with HCL OneTest™ Server in the team space:

- The projects in the team space to which the Docker host is added, are displayed in the **Projects** column.
- The status of the Docker host is displayed in the **Status** column.



**Note:** You can add Docker hosts that are in the `Ready` state to your project, and then use them as locations to run tests.

### Results

You have viewed the remote Docker hosts that are registered with HCL OneTest™ Server in your team space.

### What to do next

You must add the registered remote Docker host to your project in your team space before you use the remote Docker host as a location to run tests in your project. See [Adding a remote Docker host to the project for running tests on page 299](#).

## Adding a remote Docker host to the project for running tests

You can choose to run tests on Docker hosts that you have set up on remote host computers. You must register the Docker host with HCL OneTest™ Server and then add them to your project before you run tests on the remote Docker hosts.

### Before you begin

You must have completed the following tasks:

- Read about the considerations you must take into account before you configure a test run to run on a remote Docker host. See [Test run considerations for running tests on remote Docker hosts on page 289](#).
- Set up the remote Docker host system. See [Setting up a remote Docker host computer on page 291](#) or [Setting up a secure remote Docker host computer on page 292](#).
- Registered the remote Docker host with HCL OneTest™ Server. See [Registering a remote Docker host on page 296](#).

1. Log in to HCL OneTest™ Server.
2. Open your project by clicking **Projects > My Projects > *project\_name***.  
The **Overview** page is displayed.
3. Click **Manage > Infrastructure**.

The **Infrastructure** page is displayed.

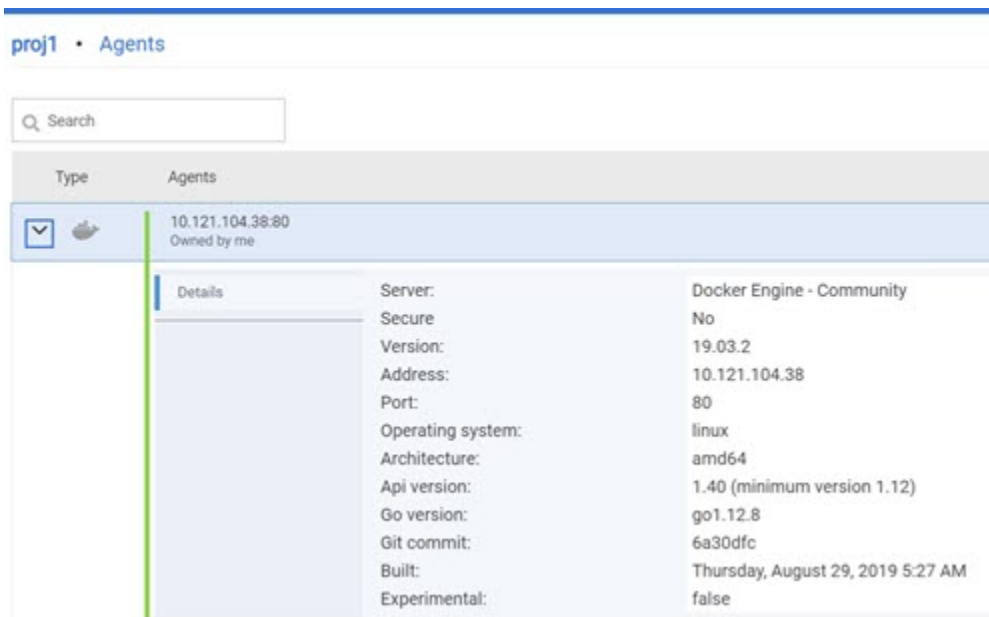
4. Click **Add > Add docker**.
5. Select the Docker host that you want to add to the project from the list of Docker hosts, and then click **Add**.



**Note:** You can add the Docker hosts that you have registered. You can add any number of Docker hosts to your project.

The Docker hosts that you added to the project are displayed.

You can view the details of the Docker host by clicking the **Expand** icon .



The screenshot shows the 'Agents' section of the HCL OneTest Server interface. At the top, there is a search bar and a breadcrumb 'proj1 • Agents'. Below this is a table with columns 'Type' and 'Agents'. One agent is listed with the address '10.121.104.38:80' and 'Owned by me'. A 'Details' tab is selected for this agent, showing the following information:

Server:	Docker Engine - Community
Secure	No
Version:	19.03.2
Address:	10.121.104.38
Port:	80
Operating system:	linux
Architecture:	amd64
Api version:	1.40 (minimum version 1.12)
Go version:	go1.12.8
Git commit:	6a30dfc
Built:	Thursday, August 29, 2019 5:27 AM
Experimental:	false

### What to do next

You can select any of the remote Docker hosts as an alternate location to run the test asset in your project while configuring a test run from the **Execution** page.

#### Related information

Tests configurations and test runs

## Editing configurations of a remote Docker host

You can edit the host name or the port of the registered Docker host computer instead of registering it as a new remote host if any of the parameters are changed on the remote host computer. You can also change the mode of authentication of a registered remote Docker host.

### Before you begin

You must have completed the following tasks:

- Set up the remote Docker host computer. See [Setting up a remote Docker host computer on page 291](#) or [Setting up a secure remote Docker host computer on page 292](#).
- Registered the remote Docker host with HCL OneTest™ Server in your team space. See [Registering a remote Docker host on page 296](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Click **Infrastructure > Agents and Intercepts** in the navigation pane.



The **Agents and Intercepts** page is displayed.

You can view the agents, intercepts or Dockers that are registered with HCL OneTest™ Server in your team space.

3. You can view the Dockers that you own in any of the following ways:

- Search for the Docker by entering the name of the Docker host in the **Search** field.




**Note:** You can enter either the full name or any text that is in the name. The search is enabled for case sensitive text that you can enter.


- Sort the **Type** column to sort the items and then identify the Docker by the name displayed.
- Sort the **Agents** column to sort the items and identify the Docker by the owner.

You can view the following details about the Dockers that are registered with HCL OneTest™ Server in your team space:

- The projects in the team space to which the Docker host is added, are displayed in the **Projects** column.
- The status of the Docker host is displayed in the **Status** column.

4. Identify the Docker host that you want to modify the configuration, and then click the **Edit** icon .




**Note:** The  icon is displayed in the **Actions** column in the row of the Docker only if you own the remote Docker host.

The **Update docker host** dialog box is displayed.

5. Change any of the configurations.

To change...	Do this...
Change the host name or the port of the remote Docker host.	Enter the changed value of the host name or the port in the <b>Remote hostname</b> field.
Change the mode of authentication from a non-secure mode to a secure mode.	<ol style="list-style-type: none"> <li>Select the <b>Secure mode</b> option.</li> <li>Click the action labels to browse and select the <code>.pem</code> files that correspond to <code>ca_certificate.pem</code>, <code>client_certificate.pem</code>, and <code>client_key.pem</code>.</li> </ol>

To change...	Do this...
	 <b>Note:</b> You must have generated the <code>.pem</code> files on the remote host computer and copied the files to your local drive.
Change the mode of authentication from a secure mode to a non-secure mode.	Clear the <b>Secure mode</b> option that is selected.

6. Click **Test connection** to test whether a connection is established between HCL OneTest™ Server and the remote host computer:
  - On a successful connection, a message is displayed.
  - On a failure to connect to the remote host computer, an error message is displayed. You must resolve the error and reattempt to establish a successful connection.
7. Click **Update**.

The remote Docker host with the updated details is displayed.

## Results

You have updated the registered remote Docker host with the changed parameters.

## What to do next

You can now add the updated Docker host to your project. See [Adding a remote Docker host to the project for running tests on page 299](#).

---

### Related information

[Setting up a remote Docker host computer on page 291](#)

[Registering a remote Docker host on page 296](#)

[Removing a remote Docker host from a project on page 302](#)

[Unregistering a remote Docker host from HCL OneTest Server on page 304](#)

## Removing a remote Docker host from a project

You can delete a remote Docker host that you added to your project in HCL OneTest™ Server when you no longer need it to run tests.

### Before you begin

You must have completed the following tasks:

1. Set up the remote Docker host computer. See [Setting up a remote Docker host computer on page 291](#) or [Setting up a secure remote Docker host computer on page 292](#).
2. Registered the remote Docker host with HCL OneTest™ Server in your team space. See [Registering a remote Docker host on page 296](#).
3. Added the registered Docker host to a project in your team space. See [Adding a remote Docker host to the project for running tests on page 299](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project in your team space from the **My Projects** page.

The **Overview** page is displayed.

3. Click **Infrastructure > Agents and Intercepts** in the navigation pane.

The **Agents and Intercepts** page is displayed.

You can view the agents, intercepts or Dockers that are added to your project.

4. You can view the Dockers that you own in any of the following ways:
  - Search for the Docker by entering the name of the Docker host in the **Search** field.




**Note:** You can enter either the full name or any text that is in the name. The search is enabled for case sensitive text that you can enter.


- Sort the **Type** column to sort the items and then identify the Docker by the name displayed.
- Sort the **Agents** column to sort the items and identify the Docker by the owner.

You can view the following details about the Dockers that are registered with HCL OneTest™ Server:

- The projects to which the Docker host is added, are displayed in the **Projects** column.
- The status of the Docker host is displayed in the **Status** column.

5. Identify the Docker host that you want to remove from the project, and then click the **Remove docker host** icon .



**Note:** The  icon is displayed in the **Actions** column in the row of the Docker only if you own the remote Docker host.

A message is displayed that the Docker host is removed from the project successfully.

## Results

You have removed the remote Docker host from your project. The remote Docker host is no longer available as a location while configuring a test run in your project.

## What to do next

You might have to add a remote Docker host to your project to use the remote Docker host as a location to run tests.

---

#### Related information

[Unregistering a remote Docker host from HCL OneTest Server on page 304](#)

[Editing configurations of a remote Docker host on page 300](#)

## Unregistering a remote Docker host from HCL OneTest™ Server

You can unregister a remote Docker host that is registered with HCL OneTest™ Server in a team space when you no longer require it.

### Before you begin

You must have completed the following tasks:

1. Set up the remote Docker host computer. See [Setting up a remote Docker host computer on page 291](#) or [Setting up a secure remote Docker host computer on page 292](#).
2. Registered the remote Docker host with HCL OneTest™ Server in a team space. See [Registering a remote Docker host on page 296](#).



1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Click **Infrastructure > Agents and Intercepts** in the navigation pane.


The **Agents and Intercepts** page is displayed.

You can view the agents, intercepts or Dockers that are registered with HCL OneTest™ Server in your team space.

3. Identify the Docker host that you want to unregister.



**Note:** The  and  icons are displayed in the **Actions** column in the row of the Docker only if you own the remote Docker host.

4. Unregister the Docker host by completing the following steps:
  - a. Click the **Unregister docker host** icon .
  - b. Click **Unregister** in the **Unregister docker host** dialog box.

A message is displayed that the Docker host is unregistered successfully.

### Results

You have unregistered the remote Docker host in a team space on HCL OneTest™ Server. You cannot run tests on this remote Docker host if it is added to your project. You might have to register it again with HCL OneTest™ Server, if you want to use the remote Docker host in your projects.

---

#### Related information

[Editing configurations of a remote Docker host on page 300](#)

[Adding a remote Docker host to the project for running tests on page 299](#)

[Removing a remote Docker host from a project on page 302](#)

## Test run configurations

You can configure and run tests in HCL OneTest™ Server after you add the test resources to your project in your team space.

Before you configure a test run, you must have completed the following tasks:

- Created a project on HCL OneTest™ Server or you must have been granted access to a project with the *Tester* role assigned. See [Test assets and a server project on page 580](#).
- Created tests in the desktop clients and committed the test assets and test resources to a remote repository. You must have added the remote repository to the project.
- Read the considerations you must take into account for certain test types. See [Prerequisites to running tests on page 271](#).

You must be a project *Owner* or a member with the *Tester* role assigned to configure and run a test.

You can find information about how to configure and run the following types of tests:

### Configuring a test for a quick run


You can configure any type of test to be run on HCL OneTest™ Server when you want to quickly ensure that the test runs correctly. You might not want to set the different options for the test nor want to schedule the run.

#### Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in [Prerequisites to running tests on page 271](#) if they apply to the test you want to configure for a run.
  - Ensured that you are assigned a role as a *Team Space Owner* in the team space. See [Managing members and their roles in a team space on page 576](#).
  - Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See [Managing access to server projects on page 588](#).
1. Log in to HCL OneTest™ Server and open the team space that contains your project.
  2. Open the project that contains the test assets or resources that you added from the Git repository, and then click **Execution**.
  3. Select the branch of the repository that contains the test assets or resources that you want to run.

All test assets in the selected branch are displayed on the **Execution** page.

4. Identify and select the test asset or resource that you want to run from the test assets listed.
5. Click the **Execute** icon  in the row of the identified test asset.

The **Execute test asset** dialog box is displayed.

6. Select the version of the test resources that you want to run, if you want to run a different version other than the latest version.
7. Select **Now** to initiate the test run immediately after you click **Execute**.
8. Enter a label for the test run that helps you to identify the test on the **Results** page.

After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page.

9. Click **Execute**.


The test run is initiated.

## Results

You have started a test run.

## What to do next

You can choose to perform any of the following tasks:

- View all the states of the test asset by clicking the **Show in the Progress page** icon  for the test asset for which you started or scheduled the run. See [Viewing the state of test assets on page 378](#).
- View the progress of the test from the **Progress** page. See [Viewing the progress of running test assets on page 381](#).
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See [Test results on page 435](#).

---

### Related information

[Resetting the configuration settings for a test run on page 385](#)

[Viewing the progress of running test assets on page 381](#)

### Monitoring a test run

[Stopping test runs on page 386](#)

[Canceling scheduled test runs on page 389](#)

## Configuring an AFT Suite run

After you added the test resources that you created in the desktop client to the project, you can configure an AFT Suite to be run on HCL OneTest™ Server.

## Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in [Prerequisites to running tests on page 271](#) if they apply to the test you want to configure for a run.
- Read [Test run considerations for AFT Suites on page 272](#) if you want to configure a run for an AFT Suite that has the agent location configured in the `AFT XML` file.
- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See [Managing members and their roles in a team space on page 576](#).
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See [Managing access to server projects on page 588](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the test assets or resources that you added from the Git repository, and then click **Execution**.
3. Select the branch of the repository that contains the test assets or resources that you want to run.



All test assets in the selected branch are displayed on the **Execution** page.

4. Identify the test asset or resource that you want to run by performing any of the following steps:
  - a. Identify the test asset or resource by scrolling through the list.



**Note:** You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents the test asset or resource
	AFT Suite
	API Suite
	Compound Test
	JMeter Test
	JUnit Test
	Postman resources
	Rate Schedule

	Icon	Represents the test asset or resource
		VU Schedule


- b. Search for the test asset or resource by entering any text contained in the test asset or resource name in the **Search** text box.
- c. Create a filter query by using the **New filter** option by performing the following steps:
  - i. Click **New filter**.
  - ii. Select an operator, and add a rule, or a group of rules.
  - iii. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.


You can select a parameter from the following list:

- Type
  - Test Asset Name
  - Test Asset Path
  - Last Result
  - Next Run
  - Components
- iv. Apply the filter query to filter the assets based on the query.


The test assets that match the filter criteria are displayed.

- v. Save the filter query by performing the following steps, if you want to reuse the filter query later:
  1. Click **Save**.
  2. Enter a name for the filter query.
  3. Click **Save**.
- d. Retrieve a saved filter, if you have saved filter queries earlier by performing the following steps:

 **Note:** To open the filter query, you must have created and saved a filter query.

- i. Click the **Open filters** icon .
- ii. Select the saved filter in the **Filters** dialog box.
- iii. Click **Apply** to apply the filter.

The test assets that match the filter criteria are displayed.

5. Click the **Execute** icon  in the row of the identified test asset.

The **Execute test asset** dialog box is displayed.

6. Select the version of the test resources that you want to run by completing any of the following actions:





**Note:** The test resources in the version can contain the test assets, datasets, AFT XML files, API environment tags, and other resources specific to projects created from any of the desktop clients.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

Use the following details about the version of the test resources that are displayed to identify the version that you want:

- Commit message.
- Tags labeled by the user for the version committed.
- The user who committed the version to the repository.
- Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch in the repository. If you do not select any version, then the latest version is selected for the test run.



**Notes:**

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✕ icon, then the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you had selected for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✕ icon.

7. Select the time for scheduling the test run from the following options:

- Select **Now** to initiate the test run immediately after you click **Execute**.



**Important:** Click **Execute** only after you have configured the other settings in this dialog box.

- Select **Later** and configure the date and time for scheduling a test to run at the scheduled date and time.

The default time for scheduling a run is **Now**.



**Notes:**

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

8. Enter a label for the test run that helps you to identify the test on the **Results** page.

After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

The default value for the **Label** field is null or an empty field.



**Important:** The configuration that you set for the test run in the **Execute test asset** dialog box is preserved when you run the same test again. Those changes are not visible when another user logs in to HCL OneTest™ Server. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

If you are running an AFT Suite and the following conditions are true, then you must perform the next step:

- You do not want to run the test on the agent configured in the AFT XML file.
- You have not selected the agent from the **Override** column in the **Location** tab.

9. Click **Advanced** to make the following advanced configurations:

a. Add the following setting in the **Program Arguments** field:

```
-swaplocation <configured_agent_location>:<overriding_agent_location>
```

For example, if the configured agent location is *91.2.352.24* and the remote agent where you want to run the test is *100.35.117.164*, then the entry in the **Program Arguments** field is as follows:

```
-swaplocation 91.2.352.24:100.35.117.164
```

If the test that you want to configure supports Jaeger tracing, then do the required tasks in the following scenarios:

If...	Then...
You want to obtain the test results as a Jaeger trace.	Enter <code>-history jaeger</code> in the <b>Program Arguments</b> field.

If...	Then...
Jaeger is already set as a program argument and you want to remove the Jaeger trace for your test.	Delete the <code>-history jaeger</code> entry in the <b>Program Arguments</b> field.
You want the report as a test log and as a Jaeger trace.	Enter <code>-history jaeger,testlog</code> in the <b>Program Arguments</b> field.



**Note:** The default report format is the *test log* format for the test reports.



**Note:** You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Follow the instructions if you are running a test asset that contains datasets:

- a. Click the **DATA SOURCES** tab, if it is not already open.
- b. Consider the following information about datasets before you select a dataset:

The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.



**Remember:** You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

- c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.



**Note:** If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.



**Note:** If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.





**Important:** If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if the test requires a variable that must be passed to the test at the test run time.

a. Click the **VARIABLES** tab, if it is not already open.

b. Choose one of the following methods to add the variables:

- To add new variables manually, click the **Add Variable** icon , enter the name, and value of the variable.
- To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon  and select the **Upload from local system** or **Browse from server** to select the variable file.



**Note:** You must have created a file with the variables before you can select the file.

The default value for the variables is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions if you are running a test that has static agents configured:

- a. Click the **LOCATION** tab, if it is not already open.

The static agents that are configured in the test asset are listed under the **Host** column. The information about the availability of the agent is displayed.



**Note:** You must have added agents to your project from the **Infrastructure** page for the agents to be displayed under the **Override** column.

- b. Select the agent where you want to run the test asset.

You can select the same agent that is configured in the test asset. Alternatively, you can override the agent with any other agent added to the project by selecting it from the list in the **Override** column.

The default value for the agents is null or an empty field if no agents were configured in the test asset. If the test asset contains agents that are configured, then the default agent is the first item to be displayed on the list of agents listed in the increasing alphabetical order.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

13. Follow the instructions if you want to change the location for running the test:

- a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.



**Important:** You must have added remote Docker hosts that are registered with HCL OneTest™ Server to your project from the **Infrastructure** page. The remote Docker hosts are then displayed under the **Override** column.



**Notes:**

- If remote Docker hosts are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of HCL OneTest™ Server.
- If remote Docker hosts are added to your project, the added Docker hosts are displayed along with their availability status and ownership information.

- b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote Docker hosts are available in your project.
- Select the remote Docker host from the list when a remote Docker host is available in your project.
- Select **No override options**, if you selected any remote Docker host and want to revert to the **Default Cluster** to run the test asset.

If you want to run the test immediately or at the scheduled time, click **Execute**.

14. Click **Execute**.


The test run is initiated.

## Results

You have configured and either started or scheduled a test run of an AFT Suite.

## What to do next

You can choose to perform any of the following tasks after you have initiated or scheduled a run:

- Stop the test run at any point after the test run is initiated, from the **Execution** page. See [Stopping test runs on page 386](#).
- Cancel a scheduled test run, from the **Execution** page. See [Canceling scheduled test runs on page 389](#).
- View all the states of the test asset by clicking the **Show in the Progress page** icon  for the test asset for which you started or scheduled the run. See [Viewing the state of test assets on page 378](#).
- View the progress of the test from the **Progress** page. See [Viewing the progress of running test assets on page 381](#).
- Monitor the test from the **Progress** page. See [Monitoring a test run](#).
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See [Test results on page 435](#).

---

### Related information

[Resetting the configuration settings for a test run on page 385](#)

### Tests configurations and test runs

[Test run configurations on page 305](#)

## Configuring an API Suite run

After you added the test resources that you created in the desktop client to the project, you can configure an API Suite to be run on HCL OneTest™ Server.

### Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in [Prerequisites to running tests on page 271](#) if they apply to the test you want to configure for a run.
- Completed the following tasks if you are running API Suites that use a transport and the transport requires third-party application `jar` files for a successful run:

- Read [Test run considerations for API Suites on page 273](#).
  - Copied the third-party application `JAR` files to the third-party application folder on the computer where HCL OneTest™ Server is installed, if you are running the API Suite in Kubernetes. See [Copying third-party application Jars to Kubernetes on page 119](#).
  - Copied the third-party application `JAR` files to the third-party application folder on the computer where the remote Docker host is installed, if you are running the API Suite on a remote Docker host. See [Copying third-party application Jars to a remote Docker host on page 294](#).
- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See [Managing members and their roles in a team space on page 576](#).
  - Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See [Managing access to server projects on page 588](#).
1. Log in to HCL OneTest™ Server and open the team space that contains your project.
  2. Open the project that contains the test assets or resources that you added from the Git repository, and then click **Execution**.
  3. Select the branch of the repository that contains the test assets or resources that you want to run.



All test assets in the selected branch are displayed on the **Execution** page.

4. Identify the test asset or resource that you want to run by performing any of the following steps:
  - a. Identify the test asset or resource by scrolling through the list.



**Note:** You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents the test asset or resource
	AFT Suite
	API Suite
	Compound Test
	JMeter Test
	JUnit Test
	Postman resources
	Rate Schedule

	Icon	Represents the test asset or resource
		VU Schedule


- b. Search for the test asset or resource by entering any text contained in the test asset or resource name in the **Search** text box.
- c. Create a filter query by using the **New filter** option by performing the following steps:
  - i. Click **New filter**.
  - ii. Select an operator, and add a rule, or a group of rules.
  - iii. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.


You can select a parameter from the following list:

- Type
  - Test Asset Name
  - Test Asset Path
  - Last Result
  - Next Run
  - Components
- iv. Apply the filter query to filter the assets based on the query.


The test assets that match the filter criteria are displayed.

- v. Save the filter query by performing the following steps, if you want to reuse the filter query later:
  1. Click **Save**.
  2. Enter a name for the filter query.
  3. Click **Save**.
- d. Retrieve a saved filter, if you have saved filter queries earlier by performing the following steps:

 **Note:** To open the filter query, you must have created and saved a filter query.

- i. Click the **Open filters** icon .
- ii. Select the saved filter in the **Filters** dialog box.
- iii. Click **Apply** to apply the filter.

The test assets that match the filter criteria are displayed.

5. Click the **Execute** icon  in the row of the identified test asset.

The **Execute test asset** dialog box is displayed.

6. Select the version of the test resources that you want to run by completing any of the following actions:





**Note:** The test resources in the version can contain the test assets, datasets, AFT XML files, API environment tags, and other resources specific to projects created from any of the desktop clients.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

Use the following details about the version of the test resources that are displayed to identify the version that you want:

- Commit message.
- Tags labeled by the user for the version committed.
- The user who committed the version to the repository.
- Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch in the repository. If you do not select any version, then the latest version is selected for the test run.



**Notes:**

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✕ icon, then the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you had selected for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✕ icon.

7. Select the time for scheduling the test run from the following options:

- Select **Now** to initiate the test run immediately after you click **Execute**.



**Important:** Click **Execute** only after you have configured the other settings in this dialog box.

- Select **Later** and configure the date and time for scheduling a test to run at the scheduled date and time.

The default time for scheduling a run is **Now**.



**Notes:**

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

8. Enter a label for the test run that helps you to identify the test on the **Results** page.

After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

The default value for the **Label** field is null or an empty field.



**Important:** The configuration that you set for the test run in the **Execute test asset** dialog box is preserved when you run the same test again. Those changes are not visible when another user logs in to HCL OneTest™ Server. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

9. Click **Advanced** to make the following advanced configurations:

- a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

For example, you can set a maximum Java heap size.

- b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

If the test that you want to configure supports Jaeger tracing, then do the required tasks in the following scenarios:

If...	Then...
You want to obtain the test results as a Jaeger trace.	Enter <code>-history jaeger</code> in the <b>Program Arguments</b> field.
Jaeger is already set as a program argument and you want to remove the Jaeger trace for your test.	Delete the <code>-history jaeger</code> entry in the <b>Program Arguments</b> field.

If...	Then...
You want the report as a test log and as a Jaeger trace.	Enter <code>-history jaeger, testlog</code> in the <b>Program Arguments</b> field.



**Note:** The default report format is the *test log* format for the test reports.

- c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.



**Note:** You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Follow the instructions if the API Suite has an environment or secrets configured:
  - a. Click the **ENVIRONMENT** tab, if it is not already open.
  - b. Select the API test environment from the list if there are multiple environments configured in the test asset.
  - c. Select the secrets collection that contains the secrets to be used for the test run.



**Notes:**

- The test asset that was created in the desktop client and added to the Git repository must have the environments defined as part of the API test project.
- If the test asset contained secrets, then you must create those secrets in secrets collections in the project.

The default value for the environment is the environment configured in the test asset. The default value for secrets is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if you are running a test asset that contains datasets:
  - a. Click the **DATA SOURCES** tab, if it is not already open.
  - b. Consider the following information about datasets before you select a dataset:

The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.



**Remember:** You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.



**Note:** If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.



**Note:** If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.





**Important:** If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions if the test requires a variable that must be passed to the test at the test run time.

- a. Click the **VARIABLES** tab, if it is not already open.
- b. Choose one of the following methods to add the variables:

- To add new variables manually, click the **Add Variable** icon , enter the name, and value of the variable.
- To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon  and select the **Upload from local system** or **Browse from server** to select the variable file.



**Note:** You must have created a file with the variables before you can select the file.

The default value for the variables is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

13. Follow the instructions if you want to change the location for running the test:

- a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.



**Important:** You must have added remote Docker hosts that are registered with HCL OneTest™ Server to your project from the **Infrastructure** page. The remote Docker hosts are then displayed under the **Override** column.



**Notes:**

- If remote Docker hosts are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of HCL OneTest™ Server.
- If remote Docker hosts are added to your project, the added Docker hosts are displayed along with their availability status and ownership information.

- b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote Docker hosts are available in your project.
- Select the remote Docker host from the list when a remote Docker host is available in your project.
- Select **No override options**, if you selected any remote Docker host and want to revert to the **Default Cluster** to run the test asset.

If you want to run the test immediately or at the scheduled time, click **Execute**.

14. Click **Execute**.


The test run is initiated.

## Results

You have configured and either started or scheduled a test run of an API Suite.

### What to do next

You can choose to perform any of the following tasks after you have initiated or scheduled a run:

- Stop the test run at any point after the test run is initiated, from the **Execution** page. See [Stopping test runs on page 386](#).
- Cancel a scheduled test run, from the **Execution** page. See [Canceling scheduled test runs on page 389](#).
- View all the states of the test asset by clicking the **Show in the Progress page** icon  for the test asset for which you started or scheduled the run. See [Viewing the state of test assets on page 378](#).
- View the progress of the test from the **Progress** page. See [Viewing the progress of running test assets on page 381](#).
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See [Test results on page 435](#).

---

#### Related information

[Resetting the configuration settings for a test run on page 385](#)

Tests configurations and test runs

[Test run configurations on page 305](#)

## Configuring a run of a Compound Test that contains HTML tests

After you added the test resources that you created in the desktop client to the project, you can configure a Compound Test that contains HTML tests to be run on HCL OneTest™ Server.

### Before you begin

You must have completed the following tasks:

- Authored a Compound Test in HCL OneTest™ UI and must have completed the following tasks when you created the test assets:
  - Created a Compound Test under a Web UI project in your workspace.
  - Created a Functional HTML Test under a Compound Test in a Functional test project in the same workspace that contained the Web UI project.

- Used the **startBrowser(Firefox,<URL>)** attribute with the browser value set to *Firefox* in the Functional test script.



**Restriction:** Functional test scripts that contain the **startApp()** or **callScript()** attributes are treated as non-HTML scripts and are not supported to be used in test scripts in the HTML tests.

- Ran the Compound Test that contains the HTML tests on HCL OneTest™ UI.
- Committed the projects that include the Web UI project (with the Compound Test) and the Functional test (with the HTML tests in a Compound Test) to the remote repository.
- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See [Managing members and their roles in a team space on page 576](#).
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See [Managing access to server projects on page 588](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the test assets or resources that you added from the Git repository, and then click **Execution**.
3. Select the branch of the repository that contains the test assets or resources that you want to run.



All test assets in the selected branch are displayed on the **Execution** page.

4. Identify the test asset or resource that you want to run by performing any of the following steps:
  - a. Identify the test asset or resource by scrolling through the list.



**Note:** You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents the test asset or resource
	AFT Suite
	API Suite
	Compound Test
	JMeter Test
	JUnit Test
	Postman resources
	Rate Schedule

	Icon	Represents the test asset or resource
		VU Schedule


- b. Search for the test asset or resource by entering any text contained in the test asset or resource name in the **Search** text box.
- c. Create a filter query by using the **New filter** option by performing the following steps:
  - i. Click **New filter**.
  - ii. Select an operator, and add a rule, or a group of rules.
  - iii. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.


You can select a parameter from the following list:

- Type
  - Test Asset Name
  - Test Asset Path
  - Last Result
  - Next Run
  - Components
- iv. Apply the filter query to filter the assets based on the query.


The test assets that match the filter criteria are displayed.

- v. Save the filter query by performing the following steps, if you want to reuse the filter query later:
  1. Click **Save**.
  2. Enter a name for the filter query.
  3. Click **Save**.
- d. Retrieve a saved filter, if you have saved filter queries earlier by performing the following steps:

 **Note:** To open the filter query, you must have created and saved a filter query.

- i. Click the **Open filters** icon .
- ii. Select the saved filter in the **Filters** dialog box.
- iii. Click **Apply** to apply the filter.

The test assets that match the filter criteria are displayed.

5. Click the **Execute** icon  in the row of the identified test asset.

The **Execute test asset** dialog box is displayed.

6. Select the version of the test resources that you want to run by completing any of the following actions:





**Note:** The test resources in the version can contain the test assets, datasets, AFT XML files, API environment tags, and other resources specific to projects created from any of the desktop clients.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

Use the following details about the version of the test resources that are displayed to identify the version that you want:

- Commit message.
- Tags labeled by the user for the version committed.
- The user who committed the version to the repository.
- Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch in the repository. If you do not select any version, then the latest version is selected for the test run.



**Notes:**

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✕ icon, then the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you had selected for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✕ icon.

7. Select the time for scheduling the test run from the following options:

- Select **Now** to initiate the test run immediately after you click **Execute**.



**Important:** Click **Execute** only after you have configured the other settings in this dialog box.

- Select **Later** and configure the date and time for scheduling a test to run at the scheduled date and time.

The default time for scheduling a run is **Now**.

**Notes:**

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

8. Enter a label for the test run that helps you to identify the test on the **Results** page.

After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

The default value for the **Label** field is null or an empty field.



**Important:** The configuration that you set for the test run in the **Execute test asset** dialog box is preserved when you run the same test again. Those changes are not visible when another user logs in to HCL OneTest™ Server. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

9. Click **Advanced** to make the following advanced configurations:

- Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

For example, you can set a maximum Java heap size.

- Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

If the test that you want to configure supports Jaeger tracing, then do the required tasks in the following scenarios:

If...	Then...
You want to obtain the test results as a Jaeger trace.	Enter <code>-history jaeger</code> in the <b>Program Arguments</b> field.
Jaeger is already set as a program argument and you want to remove the Jaeger trace for your test.	Delete the <code>-history jaeger</code> entry in the <b>Program Arguments</b> field.

If...	Then...
You want the report as a test log and as a Jaeger trace.	Enter <code>-history jaeger, testlog</code> in the <b>Program Arguments</b> field.



**Note:** The default report format is the *test log* format for the test reports.

- c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.



**Note:** You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Follow the instructions if you are running a test asset that contains datasets:

- a. Click the **DATA SOURCES** tab, if it is not already open.
- b. Consider the following information about datasets before you select a dataset:

The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.



**Remember:** You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

- c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.



**Note:** If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.



**Note:** If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.



**Important:** If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions in this step if the test requires a variable that must be passed to the test at the test run time.

You must configure the supported browser by using a variable if the test has a browser configured, which is different from the one that is supported by HCL OneTest™ Server.


a. Click the **VARIABLES** tab, if it is not already open.

b. Choose one of the following methods to add the variables:

- To add new variables manually, click the **Add Variable** icon , enter the name, and value of the variable.

i. Enter `RTW_WebUI_Browser_Selection` as the name.

ii. Enter `Firefox` as the value, if you want to use Firefox as the browser and override the browser that is specified in the HTML test.

- To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon  and select the **Upload from local system** or **Browse from server** to select the variable file.



**Note:** You must have created a file with the variables before you can select the file.

The default value for the variables is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions if you want to change the location for running the test:

- a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.



**Important:** You must have added remote Docker hosts that are registered with HCL OneTest™ Server to your project from the **Infrastructure** page. The remote Docker hosts are then displayed under the **Override** column.



**Notes:**

- If remote Docker hosts are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of HCL OneTest™ Server.
- If remote Docker hosts are added to your project, the added Docker hosts are displayed along with their availability status and ownership information.

- b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote Docker hosts are available in your project.
- Select the remote Docker host from the list when a remote Docker host is available in your project.
- Select **No override options**, if you selected any remote Docker host and want to revert to the **Default Cluster** to run the test asset.

If you want to run the test immediately or at the scheduled time, click **Execute**.

13. Click **Execute**.


The test run is initiated.

## Results

You have configured and either started or scheduled a test run of a Compound Test that contains traditional HTML tests.

## What to do next

You can choose to perform any of the following tasks after you have initiated or scheduled a run:

- Stop the test run at any point after the test run is initiated, from the **Execution** page. See [Stopping test runs on page 386](#).
- Cancel a scheduled test run, from the **Execution** page. See [Canceling scheduled test runs on page 389](#).
- View all the states of the test asset by clicking the **Show in the Progress page** icon  for the test asset for which you started or scheduled the run. See [Viewing the state of test assets on page 378](#).
- View the progress of the test from the **Progress** page. See [Viewing the progress of running test assets on page 381](#).
- Monitor the test from the **Progress** page. See [Monitoring a test run](#).
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See [Test results on page 435](#).

---

### Related information

[Resetting the configuration settings for a test run on page 385](#)

### Tests configurations and test runs

[Test run configurations on page 305](#)

## Configuring a run of a Compound Test that contains Web UI tests

After you added the test resources that you created in the desktop client to the project, you can configure a Compound Test that contains Web UI tests to be run on HCL OneTest™ Server.

### Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in [Prerequisites to running tests on page 271](#) if they apply to the test you want to configure for a run.
- Read about the considerations that you must take into account before you configure a test run to run on a remote agent. See [Test run considerations for running tests on remote agents on page 282](#).
- Created Web UI tests in HCL OneTest™ UI and added the test asset to the project repository on HCL OneTest™ Server.
- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See [Managing members and their roles in a team space on page 576](#).
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See [Managing access to server projects on page 588](#).









1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the test assets or resources that you added from the Git repository, and then click **Execution**.
3. Select the branch of the repository that contains the test assets or resources that you want to run.

All test assets in the selected branch are displayed on the **Execution** page.

4. Identify the test asset or resource that you want to run by performing any of the following steps:
  - a. Identify the test asset or resource by scrolling through the list.



**Note:** You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents the test asset or resource
	AFT Suite
	API Suite
	Compound Test
	JMeter Test
	JUnit Test
	Postman resources
	Rate Schedule
	VU Schedule

- b. Search for the test asset or resource by entering any text contained in the test asset or resource name in the **Search** text box.
- c. Create a filter query by using the **New filter** option by performing the following steps:
  - i. Click **New filter**.
  - ii. Select an operator, and add a rule, or a group of rules.
  - iii. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result

- Next Run
  - Components
- iv. Apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.


- v. Save the filter query by performing the following steps, if you want to reuse the filter query later:

1. Click **Save**.
2. Enter a name for the filter query.
3. Click **Save**.


- d. Retrieve a saved filter, if you have saved filter queries earlier by performing the following steps:



**Note:** To open the filter query, you must have created and saved a filter query.

- i. Click the **Open filters** icon .
- ii. Select the saved filter in the **Filters** dialog box.
- iii. Click **Apply** to apply the filter.

The test assets that match the filter criteria are displayed.

5. Click the **Execute** icon  in the row of the identified test asset.

The **Execute test asset** dialog box is displayed.

6. Select the version of the test resources that you want to run by completing any of the following actions:



**Note:** The test resources in the version can contain the test assets, datasets, AFT XML files, API environment tags, and other resources specific to projects created from any of the desktop clients.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

Use the following details about the version of the test resources that are displayed to identify the version that you want:

- Commit message.
- Tags labeled by the user for the version committed.
- The user who committed the version to the repository.
- Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.



- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch in the repository. If you do not select any version, then the latest version is selected for the test run.



**Notes:**

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✕ icon, then the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you had selected for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✕ icon.

7. Select the time for scheduling the test run from the following options:

- Select **Now** to initiate the test run immediately after you click **Execute**.



**Important:** Click **Execute** only after you have configured the other settings in this dialog box.

- Select **Later** and configure the date and time for scheduling a test to run at the scheduled date and time.

The default time for scheduling a run is **Now**.




**Notes:**

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

8. Enter a label for the test run that helps you to identify the test on the **Results** page.

After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

The default value for the **Label** field is null or an empty field.

 **Important:** The configuration that you set for the test run in the **Execute test asset** dialog box is preserved when you run the same test again. Those changes are not visible when another user logs in to HCL OneTest™ Server. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

9. Click **Advanced** to make the following advanced configurations:

- a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

For example, you can set a maximum Java heap size.

- b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

If the test that you want to configure supports Jaeger tracing, then do the required tasks in the following scenarios:

If...	Then...
You want to obtain the test results as a Jaeger trace.	Enter <code>-history jaeger</code> in the <b>Program Arguments</b> field.
Jaeger is already set as a program argument and you want to remove the Jaeger trace for your test.	Delete the <code>-history jaeger</code> entry in the <b>Program Arguments</b> field.
You want the report as a test log and as a Jaeger trace.	Enter <code>-history jaeger, testlog</code> in the <b>Program Arguments</b> field.



**Note:** The default report format is the *test log* format for the test reports.

- c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.



**Note:** You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Follow the instructions if you are running a test asset that contains datasets:

a. Click the **DATA SOURCES** tab, if it is not already open.

b. Consider the following information about datasets before you select a dataset:

The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.



**Remember:** You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.




**Note:** If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.



**Note:** If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.



 **Important:** If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.


If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if the test requires a variable that must be passed to the test at the test run time.

a. Click the **VARIABLES** tab, if it is not already open.

b. Choose one of the following methods to add the variables:

- To add new variables manually, click the **Add Variable** icon , enter the name, and value of the variable.
- To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon  and select the **Upload from local system** or **Browse from server** to select the variable file.

 **Note:** You must have created a file with the variables before you can select the file.


The default value for the variables is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions if you are running a compound test that has static agents configured:

a. Click the **LOCATION** tab, if it is not already open.

The static agents that are configured in the test asset are listed under the **Host** column. The information about the availability and capabilities of the agent are displayed.

 **Note:** You must have added agents to your project from the **Infrastructure** page for the agents to be displayed under the **Override** column.

b. Select the agent where you want to run the test asset in the following scenarios:

If...	Then...	Action
The agent that is specified in the test assets is available and no other agents are added to the project.	The agent that is specified in the test assets is displayed in the <b>Override</b> column.	Select the agent in the <b>Host</b> that is also displayed in the <b>Override</b> column as the location to run the test.

If...	Then...	Action
The agent that is specified in the test assets is not available and no other agents are added to the project.	There is no agent displayed for override in the <b>Override</b> column.	<p>You cannot run the test on the agent.</p> <p>You must wait until the agent is available to run the test on the agent.</p>
The agent that is specified in the test assets is available and there are other agents that are added to the project.	<p>Agents that have capabilities that match with the agent capabilities specified in the test assets are listed in the <b>Override</b> column as follows:</p> <ul style="list-style-type: none"> <li>▪ Agents with capabilities that best match the capabilities in the agent in the asset are at the top of the list.</li> <li>▪ Agents with capabilities that do not match with the capabilities in the agent in the asset are listed subsequently.</li> </ul> <p>You can find details of both agents for the matching capabilities when you hover over the agent in the <b>Override</b> column.</p>	<p>Perform any of the following actions:</p> <ul style="list-style-type: none"> <li>▪ Select the agent in the <b>Host</b> that is also displayed in the <b>Override</b> column as the location to run the test.</li> <li>▪ Select an agent in the <b>Override</b> column as the location to run the test.</li> </ul>

The default value for the agents is null or an empty field if no agents were configured in the test asset. If the test asset contains agents that are configured, then the default agent is the first item to be displayed on the list of agents listed in the increasing alphabetical order.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

13. Follow the instructions if you want to change the location for running the test:

- a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.



**Important:** You must have added remote Docker hosts that are registered with HCL OneTest™ Server to your project from the **Infrastructure** page. The remote Docker hosts are then displayed under the **Override** column.

**Notes:**

- If remote Docker hosts are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of HCL OneTest™ Server.
- If remote Docker hosts are added to your project, the added Docker hosts are displayed along with their availability status and ownership information.

b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote Docker hosts are available in your project.
- Select the remote Docker host from the list when a remote Docker host is available in your project.
- Select **No override options**, if you selected any remote Docker host and want to revert to the **Default Cluster** to run the test asset.

If you want to run the test immediately or at the scheduled time, click **Execute**.

14. Click **Execute**.


The test run is initiated.

**Results**

You have configured and either started or scheduled a test run of a Compound Test that contains Web UI tests.

**What to do next**

You can choose to perform any of the following tasks after you have initiated or scheduled a run:

- Stop the test run at any point after the test run is initiated, from the **Execution** page. See [Stopping test runs on page 386](#).
- Cancel a scheduled test run, from the **Execution** page. See [Canceling scheduled test runs on page 389](#).
- View all the states of the test asset by clicking the **Show in the Progress page** icon  for the test asset for which you started or scheduled the run. See [Viewing the state of test assets on page 378](#).
- View the progress of the test from the **Progress** page. See [Viewing the progress of running test assets on page 381](#).
- Monitor the test from the **Progress** page. See [Monitoring a test run](#).
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See [Test results on page 435](#).

---

Related information

[Resetting the configuration settings for a test run on page 385](#)

Tests configurations and test runs

[Test run configurations on page 305](#)

## Configuring a run of a Compound Test that contains mobile tests

After you added the test resources that you created in the desktop client to the project, you can configure a Compound Test that contains mobile tests to be run on HCL OneTest™ Server.

### Before you begin

You must have completed the following tasks:

- Created the mobile tests in HCL OneTest™ UI and added the test asset to the project repository on HCL OneTest™ Server.
- Created a variable file if you want to import the variables file. The file must contain the details of the Appium server that is connected to the remote agent or the device cloud to which the mobile device is connected.
- Read [Considerations for using Jaeger traces in reports on page 277](#) when you configure a run and you want Jaeger to report the test results for the test.
- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See [Managing members and their roles in a team space on page 576](#).
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See [Managing access to server projects on page 588](#).

### About this task

You can run the mobile tests that are contained in a Compound Test after you create the tests in HCL OneTest™ UI, and then add them to your project on HCL OneTest™ Server. You can then run the mobile tests on the mobile devices that are connected to any of the following agents or clouds:

- Remote agents on which the Appium server is installed.
- The BitBar Cloud.
- The Perfecto Mobile cloud.

You must provide the details of the server or the mobile cloud to which the mobile devices are connected as variables in [Step 11 on page 346](#). You can either enter the variables or use the file in which you entered the variables. You must refer to the following tables for the variables that are required for a successful run:

**Table 5. Variables for the Appium server**

<b>Name of the Variable</b>	<b>Action for the Value field</b>
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the Appium server.
appium.server.host	Specify the host name or IP address of the Appium server.
appium.server.port	Specify the port on the Appium server that is configured to communicate with HCL OneTest™ Server.

**Table 6. Variables for the BitBar cloud**

<b>Name of the Variable</b>	<b>Value</b>
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the BitBar Cloud.
bitbar.apikey	Specify the user token generated for your BitBar account to authenticate your connection with the BitBar Cloud.
bitbar.host	Specify the host name of the BitBar Cloud instance.
bitbar.project	Specify the name of the project that contains the recorded test.
bitbar.testrun	Specify a name for the test run that must be displayed in the BitBar dashboard for the test run.

**Table 7. Variables for the Perfecto mobile cloud**

<b>Name of the Variable</b>	<b>Value</b>
Mobile_Device_Selection	Specify the name of the mobile device that is connected to the Perfecto Mobile cloud.



**Table 7. Variables for the Perfecto mobile cloud (continued)**

Name of the Variable	Value
perfecto.securitytoken	Specify the user token generated for your Perfecto account to authenticate your connection with the Perfecto Mobile cloud.
perfecto.host	Specify the host name of the Perfecto Mobile cloud instance.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the test assets or resources that you added from the Git repository, and then click **Execution**.
3. Select the branch of the repository that contains the test assets or resources that you want to run.



All test assets in the selected branch are displayed on the **Execution** page.

4. Identify the test asset or resource that you want to run by performing any of the following steps:
  - a. Identify the test asset or resource by scrolling through the list.



**Note:** You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents the test asset or resource
	AFT Suite
	API Suite
	Compound Test
	JMeter Test
	JUnit Test
	Postman resources
	Rate Schedule

	Icon	Represents the test asset or resource
		VU Schedule


- b. Search for the test asset or resource by entering any text contained in the test asset or resource name in the **Search** text box.
- c. Create a filter query by using the **New filter** option by performing the following steps:
  - i. Click **New filter**.
  - ii. Select an operator, and add a rule, or a group of rules.
  - iii. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.


You can select a parameter from the following list:

- Type
  - Test Asset Name
  - Test Asset Path
  - Last Result
  - Next Run
  - Components
- iv. Apply the filter query to filter the assets based on the query.


The test assets that match the filter criteria are displayed.

- v. Save the filter query by performing the following steps, if you want to reuse the filter query later:
  1. Click **Save**.
  2. Enter a name for the filter query.
  3. Click **Save**.
- d. Retrieve a saved filter, if you have saved filter queries earlier by performing the following steps:

 **Note:** To open the filter query, you must have created and saved a filter query.

- i. Click the **Open filters** icon .
- ii. Select the saved filter in the **Filters** dialog box.
- iii. Click **Apply** to apply the filter.

The test assets that match the filter criteria are displayed.

5. Click the **Execute** icon  in the row of the identified test asset.

The **Execute test asset** dialog box is displayed.

6. Select the version of the test resources that you want to run by completing any of the following actions:



**Note:** The test resources in the version can contain the test assets, datasets, AFT XML files, API environment tags, and other resources specific to projects created from any of the desktop clients.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

Use the following details about the version of the test resources that are displayed to identify the version that you want:

- Commit message.
- Tags labeled by the user for the version committed.
- The user who committed the version to the repository.
- Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch in the repository. If you do not select any version, then the latest version is selected for the test run.



**Notes:**

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✕ icon, then the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you had selected for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✕ icon.

7. Select the time for scheduling the test run from the following options:

- Select **Now** to initiate the test run immediately after you click **Execute**.



**Important:** Click **Execute** only after you have configured the other settings in this dialog box.

- Select **Later** and configure the date and time for scheduling a test to run at the scheduled date and time.

The default time for scheduling a run is **Now**.

**Notes:**

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

8. Enter a label for the test run that helps you to identify the test on the **Results** page.

After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

The default value for the **Label** field is null or an empty field.



**Important:** The configuration that you set for the test run in the **Execute test asset** dialog box is preserved when you run the same test again. Those changes are not visible when another user logs in to HCL OneTest™ Server. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

9. Click **Advanced** to make the following advanced configurations:

- a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

For example, you can set a maximum Java heap size.

- b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

If the test that you want to configure supports Jaeger tracing, then do the required tasks in the following scenarios:

If...	Then...
You want to obtain the test results as a Jaeger trace.	Enter <code>-history jaeger</code> in the <b>Program Arguments</b> field.
Jaeger is already set as a program argument and you want to remove the Jaeger trace for your test.	Delete the <code>-history jaeger</code> entry in the <b>Program Arguments</b> field.

If...	Then...
You want the report as a test log and as a Jaeger trace.	Enter <code>-history jaeger, testlog</code> in the <b>Program Arguments</b> field.



**Note:** The default report format is the *test log* format for the test reports.

- c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.



**Note:** You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Follow the instructions if you are running a test asset that contains datasets:

- a. Click the **DATA SOURCES** tab, if it is not already open.
- b. Consider the following information about datasets before you select a dataset:

The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.



**Remember:** You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

- c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.



**Note:** If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.



**Note:** If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.





**Important:** If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Perform the following steps to provide the variables that specify the server or cloud to which the mobile device is attached. You can either enter the variables that must be passed to the test at the test run time or import the file that contains the variables.

- a. Click the **VARIABLES** tab, if it is not already open.

- b. Choose one of the following methods to add the variables:

- To add new variables manually, click the **Add Variable** icon , enter the name, and value of the variable.
- To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon  and select the **Upload from local system** or **Browse from server** to select the variable file.



**Note:** You must have created a file with the variables before you can select the file.

12. Click **Execute**.


The test run is initiated.

## Results

You have configured and either started or scheduled a test run of a Compound Test that contains mobile tests.

## What to do next

You can choose to perform any of the following tasks after you have initiated or scheduled a run:

- Stop the test run at any point after the test run is initiated, from the **Execution** page. See [Stopping test runs on page 386](#).
- Cancel a scheduled test run, from the **Execution** page. See [Canceling scheduled test runs on page 389](#).
- View all the states of the test asset by clicking the **Show in the Progress page** icon  for the test asset for which you started or scheduled the run. See [Viewing the state of test assets on page 378](#).
- View the progress of the test from the **Progress** page. See [Viewing the progress of running test assets on page 381](#).
- Monitor the test from the **Progress** page. See [Monitoring a test run](#).
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See [Test results on page 435](#).

---

### Related information

[Resetting the configuration settings for a test run on page 385](#)

### Tests configurations and test runs

[Test run configurations on page 305](#)

## Configuring a run of a Compound Test that contains performance tests

After you added the test resources that you created in the desktop client to the project, you can configure a Compound Test that contains performance tests to be run on HCL OneTest™ Server.

### Before you begin

You must have completed the following tasks:

- Created performance tests under a Compound Test in HCL OneTest™ Performance.
- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See [Managing members and their roles in a team space on page 576](#).
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See [Managing access to server projects on page 588](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the test assets or resources that you added from the Git repository, and then click **Execution**.
3. Select the branch of the repository that contains the test assets or resources that you want to run.









All test assets in the selected branch are displayed on the **Execution** page.

4. Identify the test asset or resource that you want to run by performing any of the following steps:

- a. Identify the test asset or resource by scrolling through the list.



**Note:** You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents the test asset or resource
	AFT Suite
	API Suite
	Compound Test
	JMeter Test
	JUnit Test
	Postman resources
	Rate Schedule
	VU Schedule

- b. Search for the test asset or resource by entering any text contained in the test asset or resource name in the **Search** text box.
- c. Create a filter query by using the **New filter** option by performing the following steps:
- Click **New filter**.
  - Select an operator, and add a rule, or a group of rules.
  - Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result
- Next Run
- Components

- Apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.


- Save the filter query by performing the following steps, if you want to reuse the filter query later:




1. Click **Save**.
  2. Enter a name for the filter query.
  3. Click **Save**.
- d. Retrieve a saved filter, if you have saved filter queries earlier by performing the following steps:



**Note:** To open the filter query, you must have created and saved a filter query.

- i. Click the **Open filters** icon .
- ii. Select the saved filter in the **Filters** dialog box.
- iii. Click **Apply** to apply the filter.

The test assets that match the filter criteria are displayed.

5. Click the **Execute** icon  in the row of the identified test asset.

The **Execute test asset** dialog box is displayed.

6. Select the version of the test resources that you want to run by completing any of the following actions:



**Note:** The test resources in the version can contain the test assets, datasets, AFT XML files, API environment tags, and other resources specific to projects created from any of the desktop clients.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

Use the following details about the version of the test resources that are displayed to identify the version that you want:

- Commit message.
- Tags labeled by the user for the version committed.
- The user who committed the version to the repository.
- Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch in the repository. If you do not select any version, then the latest version is selected for the test run.



**Notes:**



- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✕ icon, then the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you had selected for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✕ icon.

7. Select the time for scheduling the test run from the following options:

- Select **Now** to initiate the test run immediately after you click **Execute**.



**Important:** Click **Execute** only after you have configured the other settings in this dialog box.

- Select **Later** and configure the date and time for scheduling a test to run at the scheduled date and time.

The default time for scheduling a run is **Now**.



**Notes:**

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

8. Enter a label for the test run that helps you to identify the test on the **Results** page.

After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

The default value for the **Label** field is null or an empty field.



**Important:** The configuration that you set for the test run in the **Execute test asset** dialog box is preserved when you run the same test again. Those changes are not visible when another user logs in to HCL OneTest™ Server. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

9. Click **Advanced** to make the following advanced configurations:

- a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

For example, you can set a maximum Java heap size.

- b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

If the test that you want to configure supports Jaeger tracing, then do the required tasks in the following scenarios:

If...	Then...
You want to obtain the test results as a Jaeger trace.	Enter <code>-history jaeger</code> in the <b>Program Arguments</b> field.
Jaeger is already set as a program argument and you want to remove the Jaeger trace for your test.	Delete the <code>-history jaeger</code> entry in the <b>Program Arguments</b> field.
You want the report as a test log and as a Jaeger trace.	Enter <code>-history jaeger, testlog</code> in the <b>Program Arguments</b> field.



**Note:** The default report format is the *test log* format for the test reports.

- c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.



**Note:** You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Follow the instructions if you are running a test asset that contains datasets:

- a. Click the **DATA SOURCES** tab, if it is not already open.
- b. Consider the following information about datasets before you select a dataset:

The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.



**Remember:** You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.



**Note:** If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.



**Note:** If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.





**Important:** If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if the test requires a variable that must be passed to the test at the test run time.

- a. Click the **VARIABLES** tab, if it is not already open.
- b. Choose one of the following methods to add the variables:

- To add new variables manually, click the **Add Variable** icon , enter the name, and value of the variable.
- To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon  and select the **Upload from local system** or **Browse from server** to select the variable file.



**Note:** You must have created a file with the variables before you can select the file.

The default value for the variables is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions if you are running a test that has static agents configured:

- a. Click the **LOCATION** tab, if it is not already open.

The static agents that are configured in the test asset are listed under the **Host** column. The information about the availability of the agent is displayed.



**Note:** You must have added agents to your project from the **Infrastructure** page for the agents to be displayed under the **Override** column.

- b. Select the agent where you want to run the test asset.

You can select the same agent that is configured in the test asset. Alternatively, you can override the agent with any other agent added to the project by selecting it from the list in the **Override** column.


The default value for the agents is null or an empty field if no agents were configured in the test asset. If the test asset contains agents that are configured, then the default agent is the first item to be displayed on the list of agents listed in the increasing alphabetical order.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

13. Follow the instructions if you want to change the location for running the test:

- a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.

 **Important:** You must have added remote Docker hosts that are registered with HCL OneTest™ Server to your project from the **Infrastructure** page. The remote Docker hosts are then displayed under the **Override** column.

 **Notes:**

- If remote Docker hosts are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of HCL OneTest™ Server.
- If remote Docker hosts are added to your project, the added Docker hosts are displayed along with their availability status and ownership information.

b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote Docker hosts are available in your project.
- Select the remote Docker host from the list when a remote Docker host is available in your project.
- Select **No override options**, if you selected any remote Docker host and want to revert to the **Default Cluster** to run the test asset.

If you want to run the test immediately or at the scheduled time, click **Execute**.

14. Click **Execute**.


The test run is initiated.

## Results

You have configured and either started or scheduled a test run of a Compound Test that contains performance tests.

## What to do next

You can choose to perform any of the following tasks after you have initiated or scheduled a run:

- Stop the test run at any point after the test run is initiated, from the **Execution** page. See [Stopping test runs on page 386](#).
- Cancel a scheduled test run, from the **Execution** page. See [Canceling scheduled test runs on page 389](#).
- View all the states of the test asset by clicking the **Show in the Progress page** icon  for the test asset for which you started or scheduled the run. See [Viewing the state of test assets on page 378](#).
- View the progress of the test from the **Progress** page. See [Viewing the progress of running test assets on page 381](#).
- Monitor the test from the **Progress** page. See [Monitoring a test run](#).
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See [Test results on page 435](#).

---

Related information

[Resetting the configuration settings for a test run on page 385](#)

Tests configurations and test runs

[Test run configurations on page 305](#)

## Configuring a JMeter test run

After you added the test resources that you created in JMeter to the server project, you can configure a JMeter test run on HCL OneTest™ Server.

### Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in [Test run considerations for JMeter tests on page 278](#) before you configure a JMeter test run.
- Ensured that the JMeter extension on HCL OneTest™ Server is enabled before you configure a test run.
- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See [Managing members and their roles in a team space on page 576](#).
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See [Managing access to server projects on page 588](#).

### About this task



**Important:** You can run the JMeter tests only on the default cluster of HCL OneTest™ Server. You cannot run the JMeter tests on remote agents or remote Docker hosts that are registered with HCL OneTest™ Server.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the test assets or resources that you added from the Git repository, and then click **Execution**.
3. Select the branch of the repository that contains the test assets or resources that you want to run.









All test assets in the selected branch are displayed on the **Execution** page.

4. Identify the test asset or resource that you want to run by performing any of the following steps:
  - a. Identify the test asset or resource by scrolling through the list.



**Note:** You can also identify the type of the asset from the icon that represents the test type as shown in the following table:



Icon	Represents the test asset or resource
	AFT Suite
	API Suite
	Compound Test
	JMeter Test
	JUnit Test
	Postman resources
	Rate Schedule
	VU Schedule

- b. Search for the test asset or resource by entering any text contained in the test asset or resource name in the **Search** text box.
- c. Create a filter query by using the **New filter** option by performing the following steps:
  - i. Click **New filter**.
  - ii. Select an operator, and add a rule, or a group of rules.
  - iii. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result
- Next Run
- Components

- iv. Apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.


- v. Save the filter query by performing the following steps, if you want to reuse the filter query later:
  1. Click **Save**.
  2. Enter a name for the filter query.
  3. Click **Save**.

- d. Retrieve a saved filter, if you have saved filter queries earlier by performing the following steps:






**Note:** To open the filter query, you must have created and saved a filter query.

- i. Click the **Open filters** icon .
- ii. Select the saved filter in the **Filters** dialog box.
- iii. Click **Apply** to apply the filter.

The test assets that match the filter criteria are displayed.

5. Click the **Execute** icon  in the row of the identified test asset.

The **Execute test asset** dialog box is displayed.

6. Select the version of the test resources that you want to run by completing any of the following actions:
  - Expand the list in the **Version** field, find the version of the test resources, and then select the version.

Use the following details about the version of the test resources that are displayed to identify the version that you want:

- Commit message.
- Tags labeled by the user for the version committed.
- The user who committed the version to the repository.
- Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

The version that matches the search criteria is displayed and it is selected for the test run.

7. Select the time for scheduling the test run from the following options:

- Select **Now** to initiate the test run immediately after you click **Execute**.



**Important:** Click **Execute** only after you have configured the other settings in this dialog box.

- Select **Later** and configure the date and time for scheduling a test to run at the scheduled date and time.

The default time for scheduling a run is **Now**.



**Notes:**



- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

8. Enter a label for the test run that helps you to identify the test on the **Results** page.

After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

The default value for the **Label** field is null or an empty field.



**Important:** The configuration that you set for the test run in the **Execute test asset** dialog box is preserved when you run the same test again. Those changes are not visible when another user logs in to HCL OneTest™ Server. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

9. Click **Advanced** to make the following advanced configurations:

- a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

For example, you can set a maximum Java heap size.

- b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

For example, enter **-J** to override any JMeter property that is defined in the properties file.



**Note:** You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Click **Execute**.


The test run is initiated.

## Results

You have configured and either started or scheduled a test run of a JMeter test.

## What to do next

You can choose to perform any of the following tasks after you have initiated or scheduled a run:

- Stop the test run at any point after the test run is initiated, from the **Execution** page. See [Stopping test runs on page 386](#).
- Cancel a scheduled test run, from the **Execution** page. See [Canceling scheduled test runs on page 389](#).
- View all the states of the test asset by clicking the **Show in the Progress page** icon  for the test asset for which you started or scheduled the run. See [Viewing the state of test assets on page 378](#).
- View the progress of the test from the **Progress** page. See [Viewing the progress of running test assets on page 381](#).
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See [Test results on page 435](#).

---

#### Related information

[Resetting the configuration settings for a test run on page 385](#)

#### Tests configurations and test runs

[Test run configurations on page 305](#)

## Configuring a JUnit test run

After you added the test resources that contain the Maven project with the JUnit tests to the server project, you can configure a JUnit test run on HCL OneTest™ Server.

### Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in [Test run considerations for JUnit tests on page 279](#) before you configure a JUnit test run.
- Ensured that the JUnit extension on HCL OneTest™ Server is enabled before you configure a test run.
- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See [Managing members and their roles in a team space on page 576](#).
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See [Managing access to server projects on page 588](#).

### About this task



**Important:** You can run the JUnit tests only on the default cluster of HCL OneTest™ Server. You cannot run the JUnit tests on remote agents or remote Docker hosts that are registered with HCL OneTest™ Server.

After the test runs, you can view the results of the JUnit test in the following formats:

- Surefire
- Jaeger trace
- JUnit XML Report









1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the test assets or resources that you added from the Git repository, and then click **Execution**.
3. Select the branch of the repository that contains the test assets or resources that you want to run.

All test assets in the selected branch are displayed on the **Execution** page.

4. Identify the test asset or resource that you want to run by performing any of the following steps:
  - a. Identify the test asset or resource by scrolling through the list.



**Note:** You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents the test asset or resource
	AFT Suite
	API Suite
	Compound Test
	JMeter Test
	JUnit Test
	Postman resources
	Rate Schedule
	VU Schedule

- b. Search for the test asset or resource by entering any text contained in the test asset or resource name in the **Search** text box.
- c. Create a filter query by using the **New filter** option by performing the following steps:
  - i. Click **New filter**.
  - ii. Select an operator, and add a rule, or a group of rules.
  - iii. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
  - Test Asset Name
  - Test Asset Path
  - Last Result
  - Next Run
  - Components
- iv. Apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.


- v. Save the filter query by performing the following steps, if you want to reuse the filter query later:

1. Click **Save**.
2. Enter a name for the filter query.
3. Click **Save**.

- d. Retrieve a saved filter, if you have saved filter queries earlier by performing the following steps:



**Note:** To open the filter query, you must have created and saved a filter query.

- i. Click the **Open filters** icon .
- ii. Select the saved filter in the **Filters** dialog box.
- iii. Click **Apply** to apply the filter.

The test assets that match the filter criteria are displayed.

5. Click the **Execute** icon  in the row of the identified test asset.

The **Execute test asset** dialog box is displayed.

6. Select the version of the test resources that you want to run by completing any of the following actions:
- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

Use the following details about the version of the test resources that are displayed to identify the version that you want:

- Commit message.
- Tags labeled by the user for the version committed.
- The user who committed the version to the repository.
- Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

The version that matches the search criteria is displayed and it is selected for the test run.

7. Select the time for scheduling the test run from the following options:

- Select **Now** to initiate the test run immediately after you click **Execute**.



**Important:** Click **Execute** only after you have configured the other settings in this dialog box.

- Select **Later** and configure the date and time for scheduling a test to run at the scheduled date and time.

The default time for scheduling a run is **Now**.



**Notes:**

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

8. Enter a label for the test run that helps you to identify the test on the **Results** page.

After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

The default value for the **Label** field is null or an empty field.



**Important:** The configuration that you set for the test run in the **Execute test asset** dialog box is preserved when you run the same test again. Those changes are not visible when another user logs in to HCL OneTest™ Server. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

9. Click **Advanced** to make the following advanced configurations:

- a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

You can enter a variable that points to the *MAVEN\_OPTS* property.

- b. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

For example, if you have created the JUnit tests by using JDK V12, then you must enter the variable as **JDKOTHER** and the value as **jdkv12**, so that the test run uses the JDK in the path `/data/junit-ext/jdks/jdkv12`.



**Note:** You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Click **Execute**.


The test run is initiated.

## Results

You have configured and either started or scheduled a test run of a JUnit test.

## What to do next

You can choose to perform any of the following tasks after you have initiated or scheduled a run:

- Stop the test run at any point after the test run is initiated, from the **Execution** page. See [Stopping test runs on page 386](#).
- Cancel a scheduled test run, from the **Execution** page. See [Canceling scheduled test runs on page 389](#).
- View all the states of the test asset by clicking the **Show in the Progress page** icon  for the test asset for which you started or scheduled the run. See [Viewing the state of test assets on page 378](#).
- View the progress of the test from the **Progress** page. See [Viewing the progress of running test assets on page 381](#).
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See [Test results on page 435](#).

---

### Related information

[Resetting the configuration settings for a test run on page 385](#)

### Tests configurations and test runs

[Test run configurations on page 305](#)

## Configuring a Postman test run

After you added the test resources that you created and exported from Postman to the server project, you can configure a run for Postman collections on HCL OneTest™ Server.

### Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in [Test run considerations for Postman tests on page 280](#) before you configure a test run for the Postman collections.
- Ensured that the Postman extension on HCL OneTest™ Server is enabled before you configure a test run.
- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See [Managing members and their roles in a team space on page 576](#).
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See [Managing access to server projects on page 588](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the test assets or resources that you added from the Git repository, and then click **Execution**.
3. Select the branch of the repository that contains the test assets or resources that you want to run.

All test assets in the selected branch are displayed on the **Execution** page.

4. Identify the test asset or resource that you want to run by performing any of the following steps:









**Note:** You can either run the entire Postman collection or any of the folders within the collection.




- a. Identify the test asset or resource by scrolling through the list.



**Note:** You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents the test asset or resource
	AFT Suite
	API Suite
	Compound Test
	JMeter Test
	JUnit Test
	Postman resources



 <b>Icon</b>	<b>Represents the test asset or resource</b>
	Rate Schedule
	VU Schedule

- b. Search for the test asset or resource by entering any text contained in the test asset or resource name in the **Search** text box.
- c. Create a filter query by using the **New filter** option by performing the following steps:
  - i. Click **New filter**.
  - ii. Select an operator, and add a rule, or a group of rules.
  - iii. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.


You can select a parameter from the following list:


- Type
  - Test Asset Name
  - Test Asset Path
  - Last Result
  - Next Run
  - Components
- iv. Apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

- v. Save the filter query by performing the following steps, if you want to reuse the filter query later:
  1. Click **Save**.
  2. Enter a name for the filter query.
  3. Click **Save**.

- d. Retrieve a saved filter, if you have saved filter queries earlier by performing the following steps:

 **Note:** To open the filter query, you must have created and saved a filter query.

- i. Click the **Open filters** icon .
- ii. Select the saved filter in the **Filters** dialog box.
- iii. Click **Apply** to apply the filter.

The test assets that match the filter criteria are displayed.

5. Click the **Execute** icon  in the row of the identified test asset.

The **Execute test asset** dialog box is displayed.

6. Select the version of the test resources that you want to run.



**Restriction:** You can only select the latest version of the test asset in the selected branch in the repository.

7. Select the time for scheduling the test run from the following options:

- Select **Now** to initiate the test run immediately after you click **Execute**.



**Important:** Click **Execute** only after you have configured the other settings in this dialog box.

- Select **Later** and configure the date and time for scheduling a test to run at the scheduled date and time.

The default time for scheduling a run is **Now**.



**Notes:**

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

8. Enter a label for the test run that helps you to identify the test on the **Results** page.

After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

The default value for the **Label** field is null or an empty field.



**Important:** The configuration that you set for the test run in the **Execute test asset** dialog box is preserved when you run the same test again. Those changes are not visible when another user logs in to HCL OneTest™ Server. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

9. Click **Advanced**, and then enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

You can enter the command options that you use when you run Postman tests by using Newman, as the program arguments.

For example, enter **--verbose**, if you want the details of the test run and each request sent.



**Note:** You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Follow the instructions if you want to change the location for running the test:

a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.



**Important:** You must have added remote Docker hosts that are registered with HCL OneTest™ Server to your project from the **Infrastructure** page. The remote Docker hosts are then displayed under the **Override** column.



**Notes:**

- If remote Docker hosts are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of HCL OneTest™ Server.
- If remote Docker hosts are added to your project, the added Docker hosts are displayed along with their availability status and ownership information.

b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote Docker hosts are available in your project.
- Select the remote Docker host from the list when a remote Docker host is available in your project.
- Select **No override options**, if you selected any remote Docker host and want to revert to the **Default Cluster** to run the test asset.

If you want to run the test immediately or at the scheduled time, click **Execute**.

11. Click **Execute**.


The test run is initiated.

## Results

You have configured and either started or scheduled a test run of a Postman test.

## What to do next

You can choose to perform any of the following tasks after you have initiated or scheduled a run:

- Stop the test run at any point after the test run is initiated, from the **Execution** page. See [Stopping test runs on page 386](#).
- Cancel a scheduled test run, from the **Execution** page. See [Canceling scheduled test runs on page 389](#).
- View all the states of the test asset by clicking the **Show in the Progress page** icon  for the test asset for which you started or scheduled the run. See [Viewing the state of test assets on page 378](#).
- View the progress of the test from the **Progress** page. See [Viewing the progress of running test assets on page 381](#).
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See [Test results on page 435](#).

---

#### Related information

[Resetting the configuration settings for a test run on page 385](#)

#### Tests configurations and test runs

[Test run considerations for running tests on remote Docker hosts on page 289](#)

[Test run configurations on page 305](#)

## Configuring a run of a Rate Schedule or VU Schedule

After you added the test resources that you created in the desktop client to the project, you can configure a Rate Schedule or VU Schedule to be run on HCL OneTest™ Server.

### Before you begin

You must have completed the following tasks:

- Read and completed the tasks mentioned in [Prerequisites to running tests on page 271](#) if they apply to the test you want to configure for a run.
- Read about the considerations that you must take into account before you configure a test run to run on a remote agent. See [Test run considerations for running tests on remote agents on page 282](#).
- Read the following topics:
  - [Test run considerations for schedules on page 276](#) if you want to configure a run for Rate Schedules or VU Schedules that have Resource Monitoring sources configured.
  - [Considerations for using Jaeger traces in reports on page 277](#) if you want to configure a run and you want Jaeger to report the test results for the test.
- Created Schedules in HCL OneTest™ Performance and added the test asset to the project repository on HCL OneTest™ Server.
- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See [Managing members and their roles in a team space on page 576](#).
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See [Managing access to server projects on page 588](#).









1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the test assets or resources that you added from the Git repository, and then click **Execution**.
3. Select the branch of the repository that contains the test assets or resources that you want to run.

All test assets in the selected branch are displayed on the **Execution** page.

4. Identify the test asset or resource that you want to run by performing any of the following steps:
  - a. Identify the test asset or resource by scrolling through the list.



**Note:** You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents the test asset or resource
	AFT Suite
	API Suite
	Compound Test
	JMeter Test
	JUnit Test
	Postman resources
	Rate Schedule
	VU Schedule

- b. Search for the test asset or resource by entering any text contained in the test asset or resource name in the **Search** text box.
- c. Create a filter query by using the **New filter** option by performing the following steps:
  - i. Click **New filter**.
  - ii. Select an operator, and add a rule, or a group of rules.
  - iii. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result

- Next Run
  - Components
- iv. Apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.


- v. Save the filter query by performing the following steps, if you want to reuse the filter query later:

1. Click **Save**.
2. Enter a name for the filter query.
3. Click **Save**.


- d. Retrieve a saved filter, if you have saved filter queries earlier by performing the following steps:



**Note:** To open the filter query, you must have created and saved a filter query.

- i. Click the **Open filters** icon .
- ii. Select the saved filter in the **Filters** dialog box.
- iii. Click **Apply** to apply the filter.

The test assets that match the filter criteria are displayed.

5. Click the **Execute** icon  in the row of the identified test asset.

The **Execute test asset** dialog box is displayed.

6. Select the version of the test resources that you want to run by completing any of the following actions:



**Note:** The test resources in the version can contain the test assets, datasets, AFT XML files, API environment tags, and other resources specific to projects created from any of the desktop clients.

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

Use the following details about the version of the test resources that are displayed to identify the version that you want:

- Commit message.
- Tags labeled by the user for the version committed.
- The user who committed the version to the repository.
- Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version that is committed, and then followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

The version that matches the search criteria is displayed and it is selected for the test run.

The default value for the version selected for the run is the latest version in the selected branch in the repository. If you do not select any version, then the latest version is selected for the test run.



**Notes:**

- If you selected a version but you do not want to use that version in the test run, you can remove the selected version by clicking the ✕ icon, then the default version is selected for the test run.
- If you repeated a test or ran the test again from the **Results** page, then the version of the test resources that you had selected for the earlier run is shown as selected. You can either retain this version or select any other version from the list. You can also remove the previous version by clicking the ✕ icon.

7. Select the time for scheduling the test run from the following options:

- Select **Now** to initiate the test run immediately after you click **Execute**.



**Important:** Click **Execute** only after you have configured the other settings in this dialog box.

- Select **Later** and configure the date and time for scheduling a test to run at the scheduled date and time.

The default time for scheduling a run is **Now**.




**Notes:**

- If you have configured some or all of the settings for the current test run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a test run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.

8. Enter a label for the test run that helps you to identify the test on the **Results** page.

After the test run completes, the label that you entered is displayed for the test under the **Labels** column on the **Results** page. After you have created a label, any member of the project can use that label.

The default value for the **Label** field is null or an empty field.

 **Important:** The configuration that you set for the test run in the **Execute test asset** dialog box is preserved when you run the same test again. Those changes are not visible when another user logs in to HCL OneTest™ Server. For example, if you created new variables on the server, those variables are available only for you when the same test is run again.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

9. Click **Advanced** to make the following advanced configurations:

- a. Enter any JVM arguments that must be passed to the test run at run time in the **JVM Arguments** field, if applicable for the test.

For example, you can set a maximum Java heap size.

- b. Enter program arguments that must be passed to the test run at run time in the **Program Arguments**, if applicable for the test.

If the test that you want to configure supports Jaeger tracing, then do the required tasks in the following scenarios:

If...	Then...
You want to obtain the test results as a Jaeger trace.	Enter <code>-history jaeger</code> in the <b>Program Arguments</b> field.
Jaeger is already set as a program argument and you want to remove the Jaeger trace for your test.	Delete the <code>-history jaeger</code> entry in the <b>Program Arguments</b> field.
You want the report as a test log and as a Jaeger trace.	Enter <code>-history jaeger, testlog</code> in the <b>Program Arguments</b> field.



**Note:** The default report format is the *test log* format for the test reports.

- c. Enter the environment variables that must be passed to the test run at run time in the **Environment Variables** field, if applicable for the test.

For example, enter the environment variables when the third-party libraries that are used in the test run refer to the environment variables for configuration.





**Note:** You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for each of the fields for the advanced settings is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

10. Follow the instructions if you are running a test asset that contains datasets:

a. Click the **DATA SOURCES** tab, if it is not already open.

b. Consider the following information about datasets before you select a dataset:

The default value for the datasets in the **DATA SOURCES** tab is null if the test asset did not have an associated dataset. If the asset had an associated dataset, the default value is the associated dataset.

You can utilize the dataset stored as an Excel or CSV file to override the original dataset associated with the Suite, test, or schedule. For example, when you have associated a dataset in `.xlsx`, `.xls`, or `.csv` format with the test or schedule in desktop clients and if you have another set of data stored in an Excel or CSV file, then you can select that dataset from the **Override** list.



**Remember:** You must have uploaded the dataset as an Excel or CSV file into the Git repository, and ensured that both the original dataset (from the test asset) and new datasets (added to the project) have the same column names.

c. Select the dataset that you want to use in the test run from any of the following options:

- Select the dataset that is displayed as the default dataset when the test asset contains a single dataset.




**Note:** If there is only one dataset in the test asset, then that dataset is displayed as the default dataset.

- Select the dataset from the list.



**Note:** If there are multiple datasets in the test asset, the datasets are listed in their increasing alphabetical order.

- Select the dataset from the **Override** list to override the dataset that was associated with the test in the desktop client.



 **Important:** If the test contains an encrypted dataset, the Project Owner must classify it in the **DATA SECURITY** tab on the Project page before you can select it. You must have added datasets to your project from the **Dataset** page for the datasets to be displayed in the **Override** list.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

11. Follow the instructions if the test requires a variable that must be passed to the test at the test run time.

a. Click the **VARIABLES** tab, if it is not already open.

b. Choose one of the following methods to add the variables:

- To add new variables manually, click the **Add Variable** icon , enter the name, and value of the variable.
- To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon  and select the **Upload from local system** or **Browse from server** to select the variable file.



**Note:** You must have created a file with the variables before you can select the file.

The default value for the variables is null or an empty field.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

12. Follow the instructions if you are running a test that has static agents configured:

a. Click the **LOCATION** tab, if it is not already open.

The static agents that are configured in the test asset are listed under the **Host** column. The information about the availability of the agent is displayed.



**Note:** You must have added agents to your project from the **Infrastructure** page for the agents to be displayed under the **Override** column.

b. Select the agent where you want to run the test asset.

You can select the same agent that is configured in the test asset. Alternatively, you can override the agent with any other agent added to the project by selecting it from the list in the **Override** column.


The default value for the agents is null or an empty field if no agents were configured in the test asset. If the test asset contains agents that are configured, then the default agent is the first item to be displayed on the list of agents listed in the increasing alphabetical order.

If you want to run the test immediately or at the scheduled time, click **Execute**, or continue with the next step.

13. Follow the instructions if you want to change the location for running the test:

- a. Click the **LOCATION** tab, if it is not already open.

The **Default Cluster** is the default location where the test runs, and it is listed under the **Host** column. The information about the availability of the default location is displayed.

 **Important:** You must have added remote Docker hosts that are registered with HCL OneTest™ Server to your project from the **Infrastructure** page. The remote Docker hosts are then displayed under the **Override** column.



**Notes:**

- If remote Docker hosts are not added to your project, the option **No override options** is displayed as the default value and the test runs in the Kubernetes cluster of HCL OneTest™ Server.
- If remote Docker hosts are added to your project, the added Docker hosts are displayed along with their availability status and ownership information.

- b. Select the location where you want to run the test asset from the following options:

- Select the **Default Cluster** when no remote Docker hosts are available in your project.
- Select the remote Docker host from the list when a remote Docker host is available in your project.
- Select **No override options**, if you selected any remote Docker host and want to revert to the **Default Cluster** to run the test asset.

If you want to run the test immediately or at the scheduled time, click **Execute**.


14. Follow the instructions if you want to override the Resource Monitoring labels in the test asset with the Resource Monitoring labels that you created in HCL OneTest™ Server:

- a. Click the **Resource Monitoring** tab.



**Note:**

The **Resource Monitoring** tab displays only if you enabled the **Resource Monitoring from Service** option in HCL OneTest™ Performance when you created the test asset.

- b. Click  , and press the `Ctrl + Space bar` keys, or enter the initial letter of a label to select a label in the list.



**Note:**

You can select or add labels only if you added the labels in the **Resource Monitoring Sources** page in HCL OneTest™ Server.

15. Click **Execute**.


The test run is initiated.

## Results

You have configured and either started or scheduled a test run of a Rate Schedule or VU Schedule.

## What to do next

You can choose to perform any of the following tasks after you have initiated or scheduled a run:

- Stop the test run at any point after the test run is initiated, from the **Execution** page. See [Stopping test runs on page 386](#).
- Cancel a scheduled test run, from the **Execution** page. See [Canceling scheduled test runs on page 389](#).
- View all the states of the test asset by clicking the **Show in the Progress page** icon  for the test asset for which you started or scheduled the run. See [Viewing the state of test assets on page 378](#).
- View the progress of the test from the **Progress** page. See [Viewing the progress of running test assets on page 381](#).
- Monitor the test from the **Progress** page. See [Monitoring a test run](#).
- View the results, reports, and logs of the test from the **Results** page after the test completes the run. See [Test results on page 435](#).

---

### Related information

[Resetting the configuration settings for a test run on page 385](#)

### Tests configurations and test runs

[Test run configurations on page 305](#)

## Running tests by using Data Fabrication

Starting from V10.0.2, if a test or schedule is associated with a dataset, you can replace the dataset at run time with the schema created from the **Data Fabrication** page.

### Before you begin

You must have completed the following tasks:

- Created a dataset and associated it with a test in desktop clients and added to the Git repository.
- Created a project in HCL OneTest™ Server. See [Test assets and a server project on page 580](#).

- Configured the repository that contains the test assets in your project. See [Adding repositories to a server project on page 583](#).
- Created a schema from the **Data Fabrication** page. See [Schema fabrication on page 161](#).

### About this task

In HCL OneTest™ Server, from the **Data Fabrication** page, you can create a schema that can be used while running a test or schedule that is associated with a dataset. For example, after the recording, a test is generated that captures interactions between the client and the server. When you run this test, it uses the same data that you used during recording. For example, if you want to test the application with random test data, you can override the dataset being used in the test or schedule with a schema that you created from the **Data Fabrication** page of HCL OneTest™ Server.

Whenever you generate test data to perform application testing, the generated data is different. You can produce the same set of data multiple times by setting the seed value. For example, you have used seed value as 1 to generate test data. When you run the test or schedule by providing the seed value as 1, the same set of data is used for testing instead of different data.



**Note:** You must ensure that the original dataset that is associated with a test or schedule and a schema that you have created have the same column names.

1. Initiate the test run on the **Execution** page.
2. Create or pick a label for the test run to identify the run.

After the run completes, the label is displayed against the run on the **Results** page.

3. Click the **DATA SOURCES** tab and perform the following sub-steps:
  - a. Select a schema to override the dataset that was associated with a test or schedule in the desktop client.



**Note:** If the dataset and schema have the same column names, only then the **Override** drop-down list shows the existing schema names.

- b. Enter the number of records that you want to generate in the **Number of rows** field.
- c. Optional. Enter a seed value in the **Seed value (optional)** field.

The seed value acts as an instance of random data that can be repeated if required. By default, the seed value is blank. Alternatively, you can use the up-down control button to increment or decrement the **Number of rows** and **Seed value**.



**Note:** You must provide a value in the **Number of rows** field. Otherwise, you cannot run the test or schedule.

ENVIRONMENT	DATA SOURCES	VARIABLES	LOCATION														
	<table border="1"> <thead> <tr> <th>Data source</th> <th>Override</th> </tr> </thead> <tbody> <tr> <td>PBW_1002_DS_Encrypt <i>PT10_199/Datasets/PBW_1002_DS_En.csv</i></td> <td>Excel_Schema/Excel_Group</td> </tr> <tr> <td></td> <td>           Set override options (required)         </td> </tr> <tr> <td></td> <td> <table border="1"> <thead> <tr> <th>Number of rows</th> <th>Seed value (optional)</th> <th></th> </tr> </thead> <tbody> <tr> <td>5</td> <td>1</td> <td>Set</td> </tr> </tbody> </table> </td> </tr> </tbody> </table>	Data source	Override	PBW_1002_DS_Encrypt <i>PT10_199/Datasets/PBW_1002_DS_En.csv</i>	Excel_Schema/Excel_Group		Set override options (required)		<table border="1"> <thead> <tr> <th>Number of rows</th> <th>Seed value (optional)</th> <th></th> </tr> </thead> <tbody> <tr> <td>5</td> <td>1</td> <td>Set</td> </tr> </tbody> </table>	Number of rows	Seed value (optional)		5	1	Set		
Data source	Override																
PBW_1002_DS_Encrypt <i>PT10_199/Datasets/PBW_1002_DS_En.csv</i>	Excel_Schema/Excel_Group																
	Set override options (required)																
	<table border="1"> <thead> <tr> <th>Number of rows</th> <th>Seed value (optional)</th> <th></th> </tr> </thead> <tbody> <tr> <td>5</td> <td>1</td> <td>Set</td> </tr> </tbody> </table>	Number of rows	Seed value (optional)		5	1	Set										
Number of rows	Seed value (optional)																
5	1	Set															
<a href="#">⚙️ Advanced Settings</a>		<input type="button" value="Cancel"/>	<input type="button" value="Execute"/>														

4. Click **Set**, and then **Execute**.

### What to do next

You can monitor a test run and check the logs to ensure that the data is taken from the schema instead of the dataset when running the test.


#### Related information

[Test assets and a server project on page 580](#)

## Management of running tests

Find information about the tasks that you can perform on a test that you configured for a run either while it runs or after it completes the run.

### Viewing the state of test assets


After you initiate or schedule a test run on the **Execution** page, you can view the state of the test assets on the **Progress** page by clicking the **Show in the Progress page** icon  that is enabled for that test asset on the **Execution**

page. You can view all the states of the test asset from the past hour such as `In transition`, `Running`, `Scheduled`, `Completed`, `Or Stopped`.

### Before you begin

You must have initiated runs of the test assets in your project from the **Execution** page.











**Note:** The **Show in the Progress page** icon  is enabled for the test asset only if a run of the test asset is either started or scheduled.

1. Click **Execution**, if you are not already on the **Execution** page.
2. Identify the test asset or resource that you want to run by performing any of the following steps:
  - a. Identify the test asset or resource by scrolling through the list.



**Note:** You can also identify the type of the asset from the icon that represents the test type as shown in the following table:

Icon	Represents the test asset or resource
	AFT Suite
	API Suite
	Compound Test
	JMeter Test
	JUnit Test
	Postman resources
	Rate Schedule
	VU Schedule

- b. Search for the test asset or resource by entering any text contained in the test asset or resource name in the **Search** text box.
- c. Create a filter query by using the **New filter** option by performing the following steps:
  - i. Click **New filter**.
  - ii. Select an operator, and add a rule, or a group of rules.
  - iii. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:

- Type
- Test Asset Name
- Test Asset Path
- Last Result
- Next Run
- Components

iv. Apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.


v. Save the filter query by performing the following steps, if you want to reuse the filter query later:

1. Click **Save**.
2. Enter a name for the filter query.
3. Click **Save**.

d. Retrieve a saved filter, if you have saved filter queries earlier by performing the following steps:



**Note:** To open the filter query, you must have created and saved a filter query.

- i. Click the **Open filters** icon .
- ii. Select the saved filter in the **Filters** dialog box.
- iii. Click **Apply** to apply the filter.

The test assets that match the filter criteria are displayed.

3. Click the **Show in the Progress page** icon  in the **Actions** column of the test asset.

The **Progress** page is displayed with the following details:

- The different states of the test asset runs that were initiated or completed in the past hour are displayed as follows:
  - In transition
  - Running
  - Completed
  - Stopped by User
  - Canceled
- The ID of the selected test asset is applied as a filter and is displayed in the filter row.

The applied filter is removed along with the displayed items for this filter when you perform any of the following actions on the **Progress** page:

- You click **Hide inactive**, and then click **Show all**.
- You visit any other page in the Navigation pane and return to the **Progress** page.




## Results

You have viewed all the states of a specific test asset runs that were initiated during the past hour.

### What to do next

You can choose to perform any of the following tasks from the **Progress** page:

- Stop test assets that are running. See [Stopping test runs on page 386](#).
- Cancel scheduled runs. See [Canceling scheduled test runs on page 389](#).
- Monitor tests (such as the Schedules, Compound Test, and AFT Suites). See [Monitoring a test run](#).
- View the results or the execution log of the test asset from the **Actions** column by clicking the **Open action menu** icon . See [Checking logs on page 384](#).
- View the results, reports, and logs of the completed test run from the **Results** page. See [Test results on page 435](#).

## Viewing the progress of running test assets

After you initiate or schedule a test run on the **Execution** page, you can view the progress of the test assets that are running or scheduled to run, from the **Progress** page. You can also view the test assets that have completed their run, are stopped or canceled, during the past hour.

### Before you begin

You must have initiated runs of the test assets in your project from the **Execution** page.

1. Open your project and click **Progress**.

Any one of the following views is displayed:

- No test assets are displayed in the **Progress** page if there are no test assets that are running, scheduled to run, or have completed their run in the past hour.
- Test assets in your project that are in any of the following states are listed:
  - Running
  - Scheduled
  - Completed
  - Stopped
  - Canceled
- No test assets are displayed but the option **Show all** is enabled. You can click **Show all** to display the test assets that ran in the past hour. Test assets that were hidden from the display when **Hide inactive** was clicked are displayed.



**Note:** Test assets are automatically removed from the display after *60 minutes* of being added to the **Progress** page. You can hide the test assets from the display by clicking **Hide inactive**, so that you can



view only the test assets that were added to the **Progress** page in the past hour. You cannot use the **Progress** page to view the history of test assets that were run from the **Execution** page.

2. Identify the test asset by performing any of the following actions:

- Identify the test asset or resource by scrolling through the list.
- Search for the test asset by entering any text contained in the test asset name in the **Search** text box.
- Create a filter query by using the **New filter** option by performing the following steps:
  - a. Click **New filter**.
  - b. Select an operator, and add a rule, or a group of rules.
  - c. Add or enter the relevant parameters and either select or enter the condition and the criteria for the condition.

You can select a parameter from the following list:


- Type
  - Name
  - Started by
  - Start time
  - End time
  - Status
  - Verdict
- d. Apply the filter query to filter the assets based on the query.

The test assets that match the filter criteria are displayed.

- e. Save the filter query by performing the following steps, if you want to reuse the filter query later:
  - i. Click **Save**.
  - ii. Enter a name for the filter query.
  - iii. Click **Save**.
- Retrieve a saved filter, if you have saved filter queries earlier by performing the following steps:



**Note:** To open the filter query, you must have created and saved a filter query.

- a. Click the **Open filters** icon .
- b. Select the saved filter in the **Filters** dialog box.
- c. Click **Apply** to apply the filter.




The test assets that match the filter criteria are displayed.

- Sort the test assets by the user who started the test by clicking the **Started by** column header, and find the test asset listed against your name.

- Sort the test assets by the type of the test assets by clicking the **Type** column header, and find the test asset type you ran or scheduled.
- Click **Hide inactive** to only display the test assets that are running or scheduled to run. This action removes all completed, canceled, stopped, or failed test assets from the display.



**Note:** The items in the rows are sorted in an ascending or descending order when the column header is clicked. The following icons show the status of sorting in a column:

-  items in the column are not sorted.
-  items in the column are sorted in the ascending order or with the earliest record on top.
-  items in the column are sorted in the descending order or with the latest record on top.

You can view the following details of the test assets on the **Progress** page:

Column header	Description
Type	Displays the icon for the type of the test asset.
Name	Displays the name of the test asset added to the project from the Git repository.
Started by	Displays the name of the user who started the run of the test asset.
Start Time	Displays the start time of the test asset run.
End Time	Displays the end time of the test asset run.
Status	<p>Displays the state of the test asset progressively.</p> <p>For example, after the run is in the <code>Initiated</code> state, it moves on to the <code>Running</code> state and ends in its final state depending on the verdict of the test run as either <code>Completed</code>, <code>Failed</code>, or <code>Inconclusive</code>.</p> <p>If the test run is stopped or a scheduled run is canceled, the status displayed is <code>Stopped by User</code> for a stopped run or <code>Canceled</code> for a canceled test.</p>

From the **Actions** column, you can perform the following operations on your test asset:


- Stop a run.
- Cancel a scheduled run.
- Monitor a VU Schedule, Rate Schedule, Compound Test, or an AFT Suite.
- View the execution log.
- View the result of the test asset on the **Results** page.

## Results

You have viewed the progress of the test assets that you ran from the **Execution** page.

### What to do next

You can choose to perform any of the following tasks from the **Progress** page:

- Stop test assets that are running. See [Stopping test runs on page 386](#).
- Cancel scheduled runs. See [Canceling scheduled test runs on page 389](#).
- Monitor tests (such as the Schedules, Compound Test, and AFT Suites). See [Monitoring a test run](#).
- View the results or the execution log of the test asset from the **Actions** column by clicking the **Open action menu** icon . See [Checking logs on page 384](#).
- View the results, reports, and logs of the completed test run from the **Results** page. See [Test results on page 435](#).

## Checking logs

To verify how the test ran or to debug test run failure, you can check the `Test Log` and `Execution log`.

### About this task

The `Test Log` displays the interaction between HCL OneTest™ Server and the application or system under test. After the test run completes, the verdict of the run can be *Pass*, *Fail*, or *Inconclusive*. If the verdict of the run is *Pass*, the `Test Log` is available in the **Results** page.



The screenshot shows a test run summary for 'cmptest-1'. The test is marked as 'Complete' with a 'Pass' status. The test path is 'WebUIProjectCustomCode/CompoundTests/cmptest-1.testsuite'. Below the summary, there are two panels: 'DETAILS' and 'Reports'. The 'DETAILS' panel shows Git information: Git Repo (git@github.com:paper-mountain/one-test-server-sample.git), Git Branch (master), and Git Path (WebUIProjectCustomCode/CompoundTests/cmptest-1.testsuite). The 'Reports' panel contains three links: 'Statistics Report', 'Test Log', and 'Mobile and Web UI Report'.

DETAILS		Reports	
Git Repo	git@github.com:paper-mountain/one-test-server-sample.git	<a href="#">Statistics Report</a>	
Git Branch	master	<a href="#">Test Log</a>	
Git Path	WebUIProjectCustomCode/CompoundTests/cmptest-1.testsuite	<a href="#">Mobile and Web UI Report</a>	

The `Execution log` displays the console messages of the run time process that runs the test. This log is useful in determining the cause of the failure if the verdict of the run is *Fail* or *Inconclusive*. You can view the `Execution log` from the **Progress** page.

### Viewing the `Test Log`

1. Go to the **Results** page and identify the test that you ran.
2. Click the test so that the **Reports** panel is displayed.
3. Click the **Test Log** in the **Reports** panel.

The `Test Log` is displayed in a browser window.

## Viewing the Execution log

4. Go to the **Progress** page and identify the test that is in the *Running* state.
5. Click the **Open action menu** icon.
6. Click **Execution log**.

The `Execution log` is displayed in a browser window.

### Related information

[Test run configurations on page 305](#)

Monitoring a test run

## Resetting the configuration settings for a test run

When you are configuring a test run either as a first-run or when repeating the run, and you do not want to proceed with the settings configured or saved for the test run, you can reset the configuration settings. Resetting the configuration reverts all the settings to their default values for the test run.

### Before you begin

You must be a member of the project with the *Owner* or *Tester* role to run the tests.

You must ensure that you have the **Execute test asset** dialog box is displayed before you proceed with resetting the configuration settings for the test run.

### About this task

You can reset the configuration settings when you are configuring a test run either for its first run or when you are repeating the test run.

1. Click **Reset** in the **Execute test asset** dialog box at any point when you want to reset the configuration settings for the test run.



#### Notes:

- If you have configured some or all of the settings for the current test run and you do not want to continue with those settings, clicking **Reset** resets the settings to their default values.
- If you are repeating a test run and do not want to use the saved settings from a previous run, clicking **Reset** reverts all the saved settings to their default values.

The settings in the **Execute test asset** dialog box are reset to their default values.

2. Use the following table to find the default values for each of the settings in the **Execute test asset** dialog box:

Window or Tab	Field or Setting	Default value is
<b>Execute test asset</b>	Scheduling the test run	<b>Now</b> is selected.
<b>Execute test asset</b>	Label for settings	Null or empty field.
<b>Advanced</b>	<b>JVM Arguments</b>	Null or empty field.
	<b>Program Arguments</b>	Null or empty field.
	<b>Environment Variables</b>	Null or empty field.
<b>ENVIRONMENT</b> tab	API test environment	The environment configured in the test asset.
	Secrets collection	Null or empty field.
<b>DATA SOURCES</b> tab	<b>Override</b> list	Null or empty field if the asset has no dataset.
		The dataset in the test asset if the asset has one dataset.
		The first dataset on the list of datasets, which are listed in the increasing alphabetical order, if the asset has multiple datasets.
<b>VARIABLES</b> tab		Null or empty field.
<b>LOCATION</b> tab	Agents	Null or empty field if the asset has no agents configured.
		The first agent on the list of agents, which are listed in the increasing alphabetical order, if the asset has multiple agents configured.
	Docker host	Internal Docker host if no remote Docker hosts are added.
		<b>No override options</b> if other Docker hosts are saved in previous runs.

### What to do next

You can complete configuring the settings that you want for the test run. See [Test run configurations on page 305](#).

## Stopping test runs

You might want to stop a test run or multiple tests that are running when you realized that you did not configure all the settings or you want to change a few settings for the test runs.

### Before you begin


You must have initiated a run for one or multiple test assets from the **Execution** page.



### About this task

You can stop either a running test or multiple running tests from the **Progress** page. You can stop the test if it is in either the *In transition* or *Running* state. You cannot stop the test run if the test has already completed its run.

1. Go to the **Progress** page and identify the tests in any of the following ways that you want to stop:

You can select the tests that satisfy the following conditions:


- The test must be either in the *In transition* or *Running* state.
  - The **Stop execution** icon  in the **Actions** column must be enabled for the test.
  - The checkbox must be enabled for the test.
2. Perform the actions in the following table:

If...	Action...
You want to stop any test that is running	Perform any of the following actions: <ul style="list-style-type: none"> <li>◦ Click the <b>Stop execution</b> icon  in the <b>Actions</b> column of the selected test.</li> <li>◦ Click the checkbox in the row of the selected test, and then click <b>Stop selected</b>.</li> </ul> The <b>Stop execution</b> dialog box is displayed.
You want to stop multiple tests that are running	Perform any of the following actions: <ul style="list-style-type: none"> <li>◦ Click the checkbox in the row of the selected tests, and then click <b>Stop selected</b>.</li> <li>◦ Click the <b>Select all</b> checkbox in the header row, and then click <b>Stop selected</b>.</li> </ul> <div style="text-align: center;">  <b>Note:</b> All running tests are selected when you click the <b>Select all</b> checkbox.         </div> The <b>Stop execution</b> dialog box is displayed.

3. Perform the following steps in the **Stop execution** dialog box that is displayed:

- a. Set the timeout period for stopping the test run. Enter a numeric value and select the unit from the options available as *Seconds*, *Minutes*, or *Hours*.

The time out period is the time during which the test run is allowed to stop on its own and after the timeout period, the test is forced to stop abruptly.


 **Important:** If the timeout period is not set, a value of *30 seconds* is considered as the default timeout period.


b. Perform any of the following actions:

- Keep the **Capture results** option selected, if you want the results to be captured for the test.

The captured results for the test are available to view from the **Results** page.

- Clear this option if you do not want to capture the results.


 **Note:** The **Capture results** option is selected as the default option.

 **Important:** The results are always captured for API suites even if you clear the **Capture results** option.

c. Perform any of the following actions:


- Keep the **Execute 'finally-block', if present** option selected, if you want the *finally-block* code in the test script, if present, to be run before the test run is stopped.
- Clear this option if you do not want to run the *finally-block* code in the test script.

 **Note:** The option **Execute 'finally-block', if present** is selected as the default option.

 **Restriction:** The *tear-down* steps equivalent to the **'finally-block'** code, if present in API suites are not controllable at test runtime. The **Execute 'finally-block', if present** option has no impact on API suites.

d. Click **Stop execution**.

A notification is displayed that the running tests are stopped.

 **Note:** If the test run completes before you can configure the options for stopping the test run, a notification is displayed in the **Stop execution** dialog box that the test asset has completed its run.


## Results

You have successfully stopped a running test or stopped multiple running tests. The stopped test runs are displayed on the **Progress** page with the status as *Stopped by User*. Stopped tests are not considered in the count of tests executed that are displayed on the **Overview** page.

## What to do next



You can re-initiate the test run from the **Execution** page or from the **Results** page by completing the configurations that you want for the test.

You can view the results or the execution log of the stopped test from the **Actions** column on the **Progress** page by clicking the **Open action menu** icon .

#### Related information

[Test run configurations on page 305](#)

[Viewing the progress of running test assets on page 381](#)

[Canceling scheduled test runs on page 389](#)

## Canceling scheduled test runs

You might want to cancel a scheduled test run or multiple scheduled test runs when you realized that you did not configure all the settings for the runs, want to change a few settings, or do not want those tests to run.

### Before you begin


You must have scheduled runs for one or multiple test assets from the **Execution** page.


### About this task


You can cancel either one scheduled test or multiple scheduled tests from the **Progress** page.

1. Go to the **Progress** page and identify the tests in any of the following ways that you want to cancel:

You can select the tests that satisfy the following conditions:

- The test must be in the `Scheduled` state.
  - The **Cancel** icon  in the **Actions** column must be enabled for the test.
  - The checkbox must be enabled for the test.
2. Perform the actions in the following table:

If...	Action...
You want to cancel any test run that is scheduled	Perform any of the following actions: <ul style="list-style-type: none"> <li>◦ Click the <b>Cancel</b> icon  in the <b>Actions</b> column of the selected test.</li> <li>◦ Click the checkbox in the row of the selected test, and then click <b>Stop selected</b>.</li> </ul> The <b>Cancel scheduled execution</b> dialog box is displayed.
You want to stop multiple test runs that are scheduled	Perform any of the following actions:

If...	Action...
	<ul style="list-style-type: none"> <li>◦ Click the checkbox in the row of the selected tests, and then click <b>Stop selected</b>.</li> <li>◦ Click the <b>Select all</b> checkbox in the header row, and then click <b>Stop selected</b>.</li> </ul> <p style="text-align: center;"> <b>Note:</b> All scheduled tests are selected when you click the <b>Select all</b> checkbox.</p> <p>The <b>Cancel scheduled execution</b> dialog box is displayed.</p>

- Click **Yes** in the **Cancel scheduled execution** dialog box.

#### Result

.

A notification is displayed that the scheduled test runs are canceled.

#### Results

You have successfully canceled a scheduled test or multiple scheduled tests. The canceled test runs are displayed on the **Progress** page with the status as *Canceled*. The canceled test runs are not displayed on the **Results** page and are not considered in the count of tests executed that are displayed on the **Overview** page.

#### What to do next

You can re-initiate the test run from the **Execution** page by completing the configurations that you want for the test.

---

#### Related information

[Test run configurations on page 305](#)

[Viewing the progress of running test assets on page 381](#)

[Stopping test runs on page 386](#)

## Management of virtualized services



**Disclaimer:** This release contains access to the Virtual services that virtualize the Istio based services feature in HCL OneTest™ Server as a Tech Preview. The Tech Preview is intended for you to view the capabilities for virtualized services offered by HCL OneTest™ Server, and to provide your feedback to the product team. You are permitted to use the information only for evaluation purposes and not for use in a production



environment. HCL provides the information without obligation of support and "as is" without warranty of any kind.

You can find information about the tasks that you can perform on and manage the virtualized services that run on HCL OneTest™ Server. You can start or stop the virtual services that are connected to HCL OneTest™ Server. You can view the routing rules and usage statistics of virtualized services, agents, or intercepts that are connected to HCL OneTest™ Server.

You must create stubs or virtual services in HCL OneTest™ API for the following types of services. You must commit the virtual service resources to a remote repository and then add the repository to your project on HCL OneTest™ Server before you can run the virtual services on HCL OneTest™ Server:

- Virtual services that utilize the WebSphere® MQ transport.
- Virtual services that utilize the HTTP transport.
- Virtual services that virtualize the Istio based services. You can run virtual services for the following types of requests received or sent by the Istio service mesh:
  - Requests received by services in the Istio service mesh.
  - Requests sent from namespaces in the Istio service mesh to external services that are not in the Istio service mesh.



**Important:** You can run virtual services only in the **Default Cluster** location of HCL OneTest™ Server. You cannot run virtual services on a remote Docker host.

## Working with virtual services

You can perform the following tasks on virtual services on HCL OneTest™ Server:

- Read about the considerations before you configure a run of the virtual services. See [Prerequisites for running virtual services on page 392](#).
- Read about the considerations before you configure a run of the Istio based services that are virtualized. See [Prerequisites for running virtualized Istio based services in HCL OneTest Server on page 394](#).
- Set up the HTTP proxy, if you have configured stubs in the test assets to use an HTTP proxy to route requests. See [Setting up the HTTP proxy on page 395](#).
- View all intercepts that are registered with HCL OneTest™ Server including those that you have not configured. See [Viewing intercepts that are registered with a team space on HCL OneTest Server on page 397](#).
- View virtual service resources in the test assets that are in the repositories added to a project on HCL OneTest™ Server. [Viewing virtual service resources on page 398](#).
- Configure runs of virtual services. See [Configuring a run of a virtual service on page 401](#).
- Configure runs of HTTP virtual services to run without using proxies. See [Running HTTP virtual services without using proxies on page 413](#).
- Run Istio stubs that were created in HCL OneTest™ API to virtualize services in the Kubernetes cluster on which HCL OneTest™ Server is installed.

- View running instances of virtual services. See [Viewing running instances of virtual services on page 418](#).
- View configurations of a running virtual service instance. See [Viewing configurations of running instances of virtual services on page 423](#).
- Modify the configurations of a running virtual service instance. See [Modifying configurations of running instances of virtual services on page 428](#).
- View details of the usage statistics of the virtual services. See [Viewing usage statistics of virtual services on page 431](#).
- View routing rules of the intercepts. See [Viewing routing rules of the virtual services on page 430](#).
- Stop a running virtual service instance. See [Stopping virtual services on page 434](#).

## Prerequisites for running virtual services


You can find information about the prerequisite tasks that you must complete before you configure a run of the virtual services on HCL OneTest™ Server.

## Prerequisites for running HTTP virtual services

You can find information about the tasks that you must complete before you configure a run of the HTTP virtual services on HCL OneTest™ Server.

### Conditions for running of HTTP virtual services

Before you run the HTTP virtual services from HCL OneTest™ Server, you must check for the following conditions and perform the actions indicated:

If...	Then...
<p>You want to route the HTTP traffic via the HTTP proxy either to the virtual service or live systems, based on routing rules.</p>	<p>You must set up the HTTP proxy. See <a href="#">Setting up the HTTP proxy on page 395</a>.</p>
<p>You are working with HCL OneTest™ Server V10.1.3, which is installed on Ubuntu.</p>	<p>HTTP virtual services are exposed via host names are of the following form:</p> <pre data-bbox="829 1459 1414 1497">in-<code>&lt;unique_id&gt;</code>.&lt;INGRESS_DOMAIN&gt;</pre> <p> <b>Note:</b> The DNS used by a computer that runs a client application or the HTTP proxy that routes traffic to the virtual service needs to resolve such host names to the IP address of the ingress domain.</p> <p>You must perform the following actions that depend on the method you have used to set up the ingress domain:</p>

If...	Then...
	<ul style="list-style-type: none"> <li>• If you used <code>nip.io</code> to form the ingress domain, then that virtual services host names are automatically resolved to the IP address of the ingress domain.  For example, if you used <code>nip.io</code> to form the ingress domain as follows: <pre>10.1.2.3.nip.io</pre>The virtual services host names are resolved by <code>nip.io</code> to the IP address as <code>10.1.2.3</code>.</li> <li>• If your ingress domain does not use <code>nip.io</code> then you must ensure that the DNS in use by the client application or the HTTP proxy can resolve host names of the virtual services to the IP address of the ingress domain. For example, if the ingress domain is as follows: <pre>myhost.mycom.com</pre>The DNS must resolve <code>*.myhost.mycom.com</code> to the IP address of <code>myhost.mycom.com</code>.</li> </ul>

## HTTPS virtual service endpoints

When you want to run an HTTP virtual service where the HTTP transport settings have **Use SSL** enabled, the virtual service endpoint that is exposed for the virtual service instance expects an HTTPS request. The exposed endpoint is managed by the same ingress controller or the HAProxy that is used for the server endpoints. The SSL/TLS termination occurs at the endpoint before the request is routed to the virtual service instance and results in the following implications:

- The **Virtualization (Server) SSL** options that are specified in the HTTP transport have no effect.
- The certificate served by the exposed virtual service endpoint is the same as that of server.

## Collection of usage metrics

If you want HCL OneTest™ Server to display the usage statistics of the virtual service instances, you must install the Prometheus server at the time of server installation.

---

Related information

[Setting up the HTTP proxy on page 395](#)

[Running HTTP virtual services without using proxies on page 413](#)

## Prerequisites for running virtualized Istio based services in HCL OneTest™ Server



**Disclaimer:** This release contains access to the Virtual services that virtualize the Istio based services feature in HCL OneTest™ Server as a Tech Preview. The Tech Preview is intended for you to view the capabilities for virtualized services offered by HCL OneTest™ Server, and to provide your feedback to the product team. You are permitted to use the information only for evaluation purposes and not for use in a production environment. HCL provides the information without obligation of support and "as is" without warranty of any kind.

You can find information about the tasks that you must complete before you run virtualized Istio based services in HCL OneTest™ Server.

### Configuring virtualized Istio based services in HCL OneTest™ Server

When you intend to virtualize services in the Istio mesh in HCL OneTest™ Server, you can create stubs in HCL OneTest™ API.

When you create the Istio stubs in HCL OneTest™ API, you must ensure that you are aware of the fully qualified domain name (FQDN) or the short name of the service that you want to virtualize in the Kubernetes cluster that hosts HCL OneTest™ Server.

The FQDN of the service is as follows:

```
<service-name>.<namespace>.<svc>.<cluster-name>
```

The short name of the service is as follows:

```
<service-name>
```

For example, if you want to virtualize the HTTP traffic in the service named as *reviews* that runs in a namespace named as *bookinfo*, and in the resource named as *svc* in the Kubernetes cluster named as *mycluster*, then the FQDN of the service is as follows:

```
reviews.bookinfo.svc.mycluster
```

The short name of the service is as follows:

```
reviews
```

You must enter the host of the service that you want to virtualize, in the **Host** field of the physical transport in HCL OneTest™ API. You can provide either the FQDN or the short name of the service.



**Note:** The pass-through action in the stubs is not supported.

### Considerations for running of stubs that virtualize Istio services

Before you run the stubs that virtualize services in namespaces contained in the HCL OneTest™ Server Kubernetes cluster, you must install HCL OneTest™ Server with the *demo* configuration.

When you install HCL OneTest™ Server by using the *demo* configuration, then HCL OneTest™ Server is preconfigured to use the **bookinfo** namespace as the virtualization namespace.



**Note:** You can change or add virtualization namespaces by adding the helm installation argument `execution.istio.namespaces` along with a list of namespaces in a helm command.

After you author the virtualized Istio based services, you must commit the virtual service resources to a repository and add the repository to the project on HCL OneTest™ Server. You can configure a run of the virtualized Istio service. See [Configuring a run of a virtual service on page 401](#).

## Collection of usage metrics

If you want HCL OneTest™ Server to display the usage statistics of the virtual service instances, you must install the Prometheus server at the time of server installation.

---

Related reference

[Installation of the server software on page 36](#)

## Setting up the HTTP proxy

When you want to run HTTP virtual services on HCL OneTest™ Server, you can set up an HTTP proxy to route requests to the virtual service. You must configure the HTTP proxy with the HCL OneTest™ Server URL and offline user token generated from HCL OneTest™ Server so that the HTTP proxy can register with HCL OneTest™ Server.

### Before you begin

You must have completed the following tasks:

- Installed the proxy component from HCL® Quality Server of the same version as that of HCL OneTest™ Server.
- Read the following procedures:
  - Generating offline user tokens. See [Managing access to HCL OneTest Server on page 565](#).
  - Viewing the *Team Space ID* as the URL alias of the team space. See [Viewing the configuration of a team space on page 573](#).

### About this task

You must edit the `registration.xml` file that is located in the `<QS_install_directory>\httptcp` directory for the following attributes:


- **server base-url**
- **security-token**
- **team-space-id**

You must enter the HCL OneTest™ Server URL, offline user token, and the ID of the team space to which the proxies must register as the values for the attributes. After you restart the proxy, the HTTP proxy registers with the team space on HCL OneTest™ Server. You can view the registered intercepts from the **Infrastructure > Agents and Intercepts** page in your team space.


1. Perform the following steps in HCL OneTest™ Server:

a. Log in to HCL OneTest™ Server.

The **Team Space Dashboard** page is displayed.

b. Generate the offline user token by clicking the **User** icon , and then click **Create Token**.

c. Copy the offline user token to a text file.

d. Click the **Settings** icon  of the team space in the **My Team Spaces** page.

The **Team Space Configuration** page is displayed.

e. Copy the text that is displayed under **URL alias** as the *Team Space ID*.

2. Perform the following steps to edit the `registration.xml` file that is located in the

`<QS_install_directory>\httptcp` directory on the computer on which you have installed HCL® Quality Server:

a. Open the `registration.xml` file by using a text editor.

b. Enter the values for the following attributes:



**Note:** If an entry exists, you must either comment out that entry and add a new one or replace the contents with the values indicated.

Attribute	Value
<b>server base-url</b>	URL of HCL OneTest™ Server
<b>security-token</b>	Offline user token generated from HCL OneTest™ Server
<b>team-space-id</b>	ID of the team space to which the proxies must register

c. Save and close the file.

d. Restart the HTTP proxy.

3. Perform the following steps in HCL OneTest™ Server to view the HTTP proxies that are registered in the team space:



- a. Log in to HCL OneTest™ Server and open the team space that contains your project.
- b. Click **Infrastructure > Agents and Intercepts** to view the registered intercepts.

The HTTP proxies that you configured with your offline user token and *Team Space ID* are displayed.

### Results

You have successfully configured the HTTP proxies and viewed that they are registered in the team space on HCL OneTest™ Server.

### What to do next

You can perform the following tasks:

- Read for the prerequisite tasks that you must perform before you can run the HTTP stubs. See [Prerequisites for running HTTP virtual services on page 392](#).
- Run the virtual service resource. See [Configuring a run of a virtual service on page 401](#).
- View the routing rules for the intercept. See [Viewing routing rules of the virtual services on page 430](#).
- View the routing rules for the intercept in a running instance of the virtual service. See [Viewing configurations of running instances of virtual services on page 423](#).

## Viewing intercepts that are registered with a team space on HCL OneTest™ Server

After you install the HTTP proxies and configure them with your offline user token, the proxies register with HCL OneTest™ Server in a team space as intercepts and you can view the registered intercepts on the **Agents and Intercepts** page for the team space.

### Before you begin

You must have completed the following tasks:

- Completed the prerequisite tasks before you configure a run of the HTTP virtual services. See [Prerequisites for running HTTP virtual services on page 392](#).
- Read the prerequisite information about virtualizing the Istio based services. See [Prerequisites for running virtualized Istio based services in HCL OneTest Server on page 394](#).
- Installed the HTTP proxies of the same version as the version of HCL OneTest™ Server.
- Configured the HTTP proxy with your offline user token so that the proxy can register as an intercept with HCL OneTest™ Server. See [Setting up the HTTP proxy on page 395](#).
- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See [Managing members and their roles in a team space on page 576](#).
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See [Managing access to server projects on page 588](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Click **Infrastructure > Agents and Intercepts** in the navigation pane.

The **Agents and Intercepts** page is displayed.

You can view the agents, intercepts, or Docker hosts that are registered in the team space on HCL OneTest™ Server.

3. You can view the intercepts that you own in any of the following ways:

- Search for the intercept by entering the name of the intercept in the **Search** field.



**Note:** You can enter either the full name or any text that is in the name. The search is enabled for case-sensitive text that you can enter.

- Sort the **Type** column to show the items sorted with the intercepts in the rows at the top of the table.
- Sort the **Agents** column and identify the intercepts by their names or by their owner.

You can view the following details about the intercepts that are registered with HCL OneTest™ Server:

- `Any project` is displayed in the **Projects** column indicating that the intercept can be used by virtual services in any project.
- The status of the intercept is displayed in the **Status** column.

## Results

You have viewed the registered intercepts from the **Agents and Intercepts** page.

## What to do next

You can start the virtual services that use the registered intercepts. See [Configuring a run of a virtual service on page 401](#).

## Viewing virtual service resources

When project repositories contain virtual service resources and you want to view the resources so that you can start them, you can view the virtual services from the **Resources** page in your team space on HCL OneTest™ Server.


## Before you begin

You must have completed the following tasks:

- Created virtual services that use the HTTP or IBM® WebSphere® MQ transport for tests in HCL OneTest™ API, committed the test resources to the remote repository, and added the repository to your project.
- Been assigned a *Member* or *Project Creator* role in a team space.
- Been assigned the *Viewer*, *Tester*, or *Owner* role in the project.

## About this task

After you have committed the test assets and resources to the remote repositories and added the repositories to projects in your team space on HCL OneTest™ Server, you can go to the **Resources** page and perform the following tasks:

- View the virtual service resources that are contained in any branch of the repositories that are added to your project.
- View the components in the System Model that are associated with the virtual service resources.
- Create or use saved filters for viewing virtual service resources.
- Run or start an instance of the virtual service resource.
- View the number of instances of the virtual service resources that are running.
- View all instances of a running virtual service on the **Instances** page by clicking the **Show in instances page** icon .

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the virtual service resources in the test assets by clicking **Projects > My Projects > project\_name**.





The **Overview** page is displayed.

3. Click **Virtualization > Resources** in the navigation pane.

The **Resources** page is displayed.

4. Complete the steps for the task that you want to perform as listed in the following table:

Task	Action
Viewing the virtual service resources that are contained in a branch of any of the repositories that are added to your project.	<p>Select the branch of the repository that contains the virtual services that you want to run from the list in the <b>Branch</b> field.</p> <p>All virtual services in the selected branch are displayed on the <b>Resources</b> page.</p>
Viewing the components that are associated with the virtual service resource.	<p>The components that are associated with the virtual service resource are displayed in the <b>Components</b> column.</p> <p>Click the component in the <b>Components</b> column to view the virtual resource on the <b>System Model</b> page.</p>
Creating or using saved filters for viewing specific instances of virtual service resources.	<p>To create a query, perform the following actions:</p> <ol style="list-style-type: none"> <li>a. Click <b>New filter</b>.</li> <li>b. Create a rule with an appropriate operator.</li> <li>c. Select criteria such as <i>Name, Path, Instance number, Instance state, or Components</i>.</li> <li>d. Select the condition, and then enter the value for the criteria.</li> <li>e. Apply the filter query.</li> </ol> <p>The virtual services that match the filter criteria are displayed.</p> <ol style="list-style-type: none"> <li>f. Save the filter query for retrieving it from the saved filters list.</li> </ol>

Task	Action
	<p>To use a saved filter query, perform the following actions:</p> <p> <b>Note:</b> To open the filter query, you must have created and saved a filter query.</p> <ol style="list-style-type: none"> <li>Click the <b>Open filters</b> icon .</li> <li>Select the saved filter.</li> <li>Apply the filter.</li> </ol> <p>The virtual services that match the filter criteria are displayed.</p>
Running an instance of the virtual service resource.	<p>After you identify the virtual service that you want to run, perform the following actions:</p> <ol style="list-style-type: none"> <li>Click the <b>Execute</b> icon  in the row of the identified virtual service.</li> </ol> <p>The <b>Execute virtual service</b> dialog box is displayed.</p> <ol style="list-style-type: none"> <li>Run the virtual service with the settings that are configured in the resource by clicking <b>Execute</b> or modify any of the settings, and then click <b>Execute</b>.</li> </ol> <p>An instance of the virtual service starts to run. The virtual service is displayed with the state as <i>Running</i> along with the number of instances that are running in the <b>Active instances</b> column.</p>
Viewing the number of instances of the virtual service resources that are running.	<p>If any of the virtual services in the project repositories are started by any project member with the <i>Tester</i> or <i>Owner</i> role, you can view the number of instances that are running or active from the <b>Resources</b> page.</p> <p>The virtual services are displayed with the state as <i>Running</i> along with the number of instances that are running, in the <b>Active instances</b> column.</p>
Viewing all instances of a running virtual service.	<p>Click the <b>Show in instances page</b> icon  in the row of the virtual service on the <b>Resources</b> page.</p>

Task	Action
	<p>The <b>Instances</b> page is displayed with all the running instances of a particular virtual service.</p> <p>You can also view the number of requests that are received by the virtual service. The number of requests is displayed as the number of hits when you hover the cursor over the text in the <b>Activity</b> column.</p>

## Results

You have viewed virtual services from the **Resources** page on HCL OneTest™ Server.

---

Related information

[Configuring a run of a virtual service on page 401](#)

## Configuring a run of a virtual service



**Disclaimer:** This release contains access to the Virtual services that virtualize the Istio based services feature in HCL OneTest™ Server as a Tech Preview. The Tech Preview is intended for you to view the capabilities for virtualized services offered by HCL OneTest™ Server, and to provide your feedback to the product team. You are permitted to use the information only for evaluation purposes and not for use in a production environment. HCL provides the information without obligation of support and "as is" without warranty of any kind.

When you have virtual service resources that are in the repositories added to your project, you must configure a run of the virtual service to start an instance of the virtual service on HCL OneTest™ Server.

### Before you begin

You must have completed the following tasks:

- Been assigned a *Member* or *Project Creator* role in a team space.
- Been assigned the *Tester* or *Owner* role in the project to run virtual services.
- Created virtual services that use the HTTP or IBM® WebSphere® MQ transport for tests in HCL OneTest™ API, committed the test resources to the remote repository, and added the repository to your project.
- Completed the prerequisite tasks before you configure a run of the HTTP stubs. See [Prerequisites for running HTTP virtual services on page 392](#).
- Read the prerequisite information about virtualizing the Istio based services before you can run the virtualized service on HCL OneTest™ Server. See [Prerequisites for running virtualized Istio based services in HCL OneTest Server on page 394](#).

## About this task

You can start the following types of stubs contained in API Suites from the **Resources** page on HCL OneTest™ Server:

- Virtual services that utilize the WebSphere® MQ transport.
- Virtual services that utilize the HTTP transport.
- Virtual services that virtualize the Istio based services. You can run virtual services for the following types of requests received or sent by the Istio service mesh:
  - Requests received by services in the Istio service mesh.
  - Requests sent from namespaces in the Istio service mesh to external services that are not in the Istio service mesh.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the virtual service resources in the test assets by clicking **Projects > My Projects > *project\_name***.

The **Overview** page is displayed.

3. Click **Virtualization > Resources** in the navigation pane.

The **Resources** page is displayed.

4. Select the branch of the repository that contains the virtual services that you want to run from the list in the **Branch** field.

All virtual services in the selected branch are displayed on the **Resources** page.

5. Identify the virtual service that you want to run by completing any of the following steps:
  - Search for the virtual service by entering the name or the path of the virtual service in the repository in the **Search** field box.
  - Create a filter query by using the **New filter** option and complete the following steps:
    - a. Create a rule with an appropriate operator.
    - b. Select criteria such as *Name*, *Path*, *Instance number*, or *Instance state*. Select the condition and enter the value for the criteria. Apply the filter query.

The virtual services that match the filter criteria are displayed.

- c. Save the filter query for retrieving it from the saved filters list.


- Retrieve a saved filter by using the **Open filters** icon  by completing the following steps:



**Note:** To open the filter query, you must have created and saved a filter query.

- a. Select the saved filter.
- b. Apply the filter.

The virtual services that match the filter criteria in the filter that is applied are displayed.

6. Click the **Execute** icon  in the row of the identified virtual service.

The **Execute virtual service** dialog box is displayed.

7. Select the version of the virtual service that is in the repository that you want to start by performing any of the following actions:

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

Use the following details about the version of the test resources that are displayed to identify the version that you want:

- Commit message.
- Tags labeled for the version committed.
- The user who committed the version to the repository.
- Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version committed followed by the versions committed previously.

- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

The version that matches the search criteria is displayed and it is selected for the test run.

8. Select the environment that was used to bind the physical and logical resource in the API project, in the **ENVIRONMENT** tab.



**Important:** The configuration that you set for the run in the **Execute virtual service** dialog box is preserved when you run the same virtual service again. The configurations that you set are not available to other members when they want to run the virtual service. For example, if you selected an environment, the same environment is selected when you run the virtual service again.

9. Enter a label, if required.

A label that you enter for the test run that helps you to identify the virtual service instance on the **Instances** page. The label that you entered is displayed for the virtual service under the **Labels** column on the **Instances** page. After you have created a label, any member of the project can use that label.

10. Follow the instructions if you want to modify the configurations for the behavior of the virtual service:

- a. Click the **BEHAVIOR** tab, if it is not already open.
- b. Configure or change the settings for the following options:



**Note:** The settings displayed are the settings that were configured for the virtual service when it was authored in HCL OneTest™ API.

Option	Description						
Performance	<p>The following settings are available for handling of requests by the virtual service:</p> <table border="1" data-bbox="574 533 1419 1654"> <thead> <tr> <th data-bbox="574 533 891 590">Option</th> <th data-bbox="891 533 1419 590">Description</th> </tr> </thead> <tbody> <tr> <td data-bbox="574 590 891 1482"><b>Optimize performance</b></td> <td data-bbox="891 590 1419 1482"> <p>If enabled, attempts are made to reduce the amount of processing between the time the virtual service receives a request and the time it sends a response. Specific optimizations depend on the message contents, as in the following examples:</p> <ul style="list-style-type: none"> <li>▪ When the virtual service receives requests, all validations are disabled, and for all XML payloads, the store and filter actions are converted to use XPath expressions.</li> <li>▪ When the virtualization sends responses, any store actions set on a message are disabled, and any XML content is collapsed when the virtual service is compiled instead of being collapsed every time that a response is sent.</li> </ul> <p>The optimization is disabled as the default action for this setting and you can enable the performance optimization if you fully understand the implications of optimizing the performance.</p> </td> </tr> <tr> <td data-bbox="574 1482 891 1654"><b>Threads</b></td> <td data-bbox="891 1482 1419 1654"> <p>The maximum number of threads used in processing requests received by the virtual service. The default number of threads is <i>10</i>.</p> </td> </tr> </tbody> </table>	Option	Description	<b>Optimize performance</b>	<p>If enabled, attempts are made to reduce the amount of processing between the time the virtual service receives a request and the time it sends a response. Specific optimizations depend on the message contents, as in the following examples:</p> <ul style="list-style-type: none"> <li>▪ When the virtual service receives requests, all validations are disabled, and for all XML payloads, the store and filter actions are converted to use XPath expressions.</li> <li>▪ When the virtualization sends responses, any store actions set on a message are disabled, and any XML content is collapsed when the virtual service is compiled instead of being collapsed every time that a response is sent.</li> </ul> <p>The optimization is disabled as the default action for this setting and you can enable the performance optimization if you fully understand the implications of optimizing the performance.</p>	<b>Threads</b>	<p>The maximum number of threads used in processing requests received by the virtual service. The default number of threads is <i>10</i>.</p>
Option	Description						
<b>Optimize performance</b>	<p>If enabled, attempts are made to reduce the amount of processing between the time the virtual service receives a request and the time it sends a response. Specific optimizations depend on the message contents, as in the following examples:</p> <ul style="list-style-type: none"> <li>▪ When the virtual service receives requests, all validations are disabled, and for all XML payloads, the store and filter actions are converted to use XPath expressions.</li> <li>▪ When the virtualization sends responses, any store actions set on a message are disabled, and any XML content is collapsed when the virtual service is compiled instead of being collapsed every time that a response is sent.</li> </ul> <p>The optimization is disabled as the default action for this setting and you can enable the performance optimization if you fully understand the implications of optimizing the performance.</p>						
<b>Threads</b>	<p>The maximum number of threads used in processing requests received by the virtual service. The default number of threads is <i>10</i>.</p>						
Operation	Specifies an operation referenced by the virtual service.						
Response time	Specifies the response time behavior for responses sent by the virtual service for the selected operation.						





Option	Description										
	<p>You can modify the value by selecting a value from the following options:</p> <table border="1" data-bbox="578 321 1412 947"> <thead> <tr> <th data-bbox="578 321 727 380">Option</th> <th data-bbox="734 321 1412 380">Description</th> </tr> </thead> <tbody> <tr> <td data-bbox="578 388 727 472">No delay</td> <td data-bbox="734 388 1412 472">Select this option for a response with no delay.</td> </tr> <tr> <td data-bbox="578 480 727 606">Minimum delay</td> <td data-bbox="734 480 1412 606">Select this option for a delay in the response by entering the required delay in milliseconds (ms).</td> </tr> <tr> <td data-bbox="578 615 727 779">Uniform distribution</td> <td data-bbox="734 615 1412 779">Select this option for a uniformly distributed response time by specifying the minimum and maximum delay in milliseconds (ms).</td> </tr> <tr> <td data-bbox="578 787 727 947">Gaussian distribution</td> <td data-bbox="734 787 1412 947">Select this option for a response time with Gaussian distribution by specifying the minimum and maximum delay in milliseconds (ms).</td> </tr> </tbody> </table>	Option	Description	No delay	Select this option for a response with no delay.	Minimum delay	Select this option for a delay in the response by entering the required delay in milliseconds (ms).	Uniform distribution	Select this option for a uniformly distributed response time by specifying the minimum and maximum delay in milliseconds (ms).	Gaussian distribution	Select this option for a response time with Gaussian distribution by specifying the minimum and maximum delay in milliseconds (ms).
Option	Description										
No delay	Select this option for a response with no delay.										
Minimum delay	Select this option for a delay in the response by entering the required delay in milliseconds (ms).										
Uniform distribution	Select this option for a uniformly distributed response time by specifying the minimum and maximum delay in milliseconds (ms).										
Gaussian distribution	Select this option for a response time with Gaussian distribution by specifying the minimum and maximum delay in milliseconds (ms).										
Passthrough	<p>Specifies the pass through behavior for the selected operation when requests are not handled within the virtual service.</p> <p>You can modify the value by selecting a value from the following options:</p> <table border="1" data-bbox="578 1161 1412 1482"> <tbody> <tr> <td data-bbox="578 1161 727 1299">Discard</td> <td data-bbox="734 1161 1412 1299">This option stops the system under test from receiving the intercepted message. This option can disrupt the calling system. For example, the system might time out while it waits for a reply.</td> </tr> <tr> <td data-bbox="578 1308 727 1392">Pass Through</td> <td data-bbox="734 1308 1412 1392">This option passes the intercepted message to the system under test, with an optional delay.</td> </tr> <tr> <td data-bbox="578 1400 727 1482">Simulate Error</td> <td data-bbox="734 1400 1412 1482">This option returns an error to the calling system. The message is not passed to the system under test.</td> </tr> </tbody> </table>	Discard	This option stops the system under test from receiving the intercepted message. This option can disrupt the calling system. For example, the system might time out while it waits for a reply.	Pass Through	This option passes the intercepted message to the system under test, with an optional delay.	Simulate Error	This option returns an error to the calling system. The message is not passed to the system under test.				
Discard	This option stops the system under test from receiving the intercepted message. This option can disrupt the calling system. For example, the system might time out while it waits for a reply.										
Pass Through	This option passes the intercepted message to the system under test, with an optional delay.										
Simulate Error	This option returns an error to the calling system. The message is not passed to the system under test.										

11. Follow the instructions, if the virtual service references datasets.

- You can use the dataset referenced in the asset.
- You can choose to override the dataset with another data source. If alternative data sources are available, select from the set of overrides available.

12. Follow the instructions if the virtual service requires a variable that must be passed at run time.

- a. Click the **VARIABLES** tab, if it is not already open.
- b. Choose one of the following methods to add the variables:
  - To add new variables manually, click the **Add Variable** icon , enter the name, and value of the variable.
  - To add new variables from your local computer or from the Git repository that is associated with your server project, click the **Upload** icon  and select the **Upload from local system** or **Browse from server** to select the variable file.



**Note:** You must have created a file with the variables before you can select the file.

13. Click **Advanced** to make the following advanced configurations:

- a. Enter any JVM arguments that must be passed at run time in the **JVM Arguments** field.



**Note:** Each JVM argument should be separated with white space.

For example, you can set a maximum Java heap size.

- b. Enter the environment variables that must be passed at run time in the **Environment Variables** field, if applicable.

For example, enter the environment variables when the third-party libraries that are used in the run refer to the environment variables for configuration.

- c. Select the stub logging level for the virtual service from the following options in the **Logging** list:


Option	Description
None	Specifies that the virtual service does not write log messages.
Normal	Specifies that the virtual service writes informational messages.
Debug	Specifies that the virtual service writes informational and debugging messages.


- d. Enter other configuration options as parameters and their values in the **Additional Configuration Parameters** fields, if applicable.

You can refer to the additional parameters that you can use for virtual services from the topic in the related links.

For example, if you want to start the virtual service to run in a new container, you can specify the following parameter and its value:

Parameter name	Value
stub.dedicated.container	true

 **Note:** Click **Add** to add additional parameters.

 **Note:** You must separate the arguments or variables with a white space when you enter them in the same line or start each argument or variable on a new line.

The default value for the fields for the advanced settings is null or an empty field.

 **Notes:**

- If you have configured some or all of the settings for the current run, and you do not want to continue with those settings, you can reset the settings by clicking **Reset**.
- If you want to repeat a run and do not want to use the saved settings from a previous run, you can reset all the saved settings to their default values by clicking **Reset**.


14. Click **Execute**.

## Results

You have started a virtual service from the **Resources** page on HCL OneTest™ Server.

## What to do next

After you start a run of a virtual service, you can view the status of the virtual service in any of the following ways:

- On the **Resources** page, the virtual services are displayed with the state as `Running` along with the number of instances that are running in the **Active instances** column.
- On the **Resources** page, you must click the **Show in instances page** icon  in the row of the virtual service to view all the running instances of a particular virtual service on the **Instances** page.
- Go to the **Instances** page by clicking **Virtualization > Instances**. You can view instances of all the virtual services that are running and are displayed with the state as `Running` under the **State** column. You can also view the requests that are received by the virtual service. The number of requests is displayed as number of hits when you hover the cursor over the text in the **Activity** column.

You can view the details, configuration settings, routing rules, usage statistics, or logs of a specific running instance of the virtual service. See [Viewing configurations of running instances of virtual services on page 423](#).

You can modify the behavior or the logging level of a running instance of the virtual service that you started before running tests on them. See [Modifying configurations of running instances of virtual services on page 428](#).

When you have completed testing the virtual service, you must stop the running virtual service. See [Stopping virtual services on page 434](#).

#### Related reference

[Additional configuration parameters for virtual services on page 408](#)


## Additional configuration parameters for virtual services




You can find information about the additional configuration parameters that you can set when you want to run virtual services on HCL OneTest™ Server.

When you use the server UI to start the virtual service, some of the parameters are handled by fields in the **Execute virtual service** dialog box.




You can enter the other parameters as a *name-value* pair in the **Additional configuration parameters** field that is displayed in the **Advanced** settings panel of the **Execute virtual service** dialog box.


The following additional configuration parameters are supported when you configure a run for a virtual service.

Requirement	Configuration parameter name	Supported values	An example value	Result of using the example value
To specify a unique identifier for use in the generated host name	<code>stub.ingress.host.identifier</code>	Any <code>&lt;custom_id&gt;</code> that you specify.   <b>Note:</b> The <code>&lt;custom_id&gt;</code> must start and end with an alphanumeric character. The characters supported include a-z, 0-9, and a hyphen. No other characters can be used in the <code>&lt;custom_id&gt;</code> . The <code>&lt;custom_id&gt;</code> must not be	<i>mystub</i>	See <a href="#">Running HTTP virtual services without using proxies on page 413</a> , to know how the identifier you specified is used.

Requirement	Configuration parameter name	Supported values	An example value	Result of using the example value
		 longer than 32 characters.		
To specify whether a virtual service instance should be run in a dedicated container.	<code>stub.dedicated.container</code>	<ul style="list-style-type: none"> <li>• <i>true</i></li> <li>• <i>false</i></li> </ul>  <b>Note:</b> The default value is <i>false</i> .	<i>true</i>	A dedicated container is used to run a single virtual service instance.
To specify whether performance optimization is enabled for the virtual service instance. The value set overrides the setting from the stub definition.  This option can be set in the <b>Behavior</b> tab from the UI.	<code>stub.performance.optimize</code>	<ul style="list-style-type: none"> <li>• <i>true</i></li> <li>• <i>false</i></li> </ul>  <b>Note:</b> The default value is <i>false</i> .	<i>true</i>	The performance optimization is enabled for the virtual service.
To specify the number of threads to be used for processing requests for the virtual service instance. The value set overrides the setting from the stub definition.  This option can be set in the <b>Behavior</b> tab from the UI.	<code>stub.worker.thread-count</code>	Any number that you specify as number of threads to use.	15	15 threads are used for processing requests for the virtual service instance.

Requirement	Configuration parameter name	Supported values	An example value	Result of using the example value
<p>To specify the stub logging level for the virtual service instance. The value set overrides the setting from the stub definition.</p> <p>This option can be set in the <b>Advanced</b> settings from the UI.</p>	<code>stub.logging.level</code>	<ul style="list-style-type: none"> <li>• <i>none</i></li> <li>• <i>normal</i></li> <li>• <i>debug</i></li> </ul>	<i>debug</i>	The virtual service writes informational and debugging messages to the container log.
<p>To specify the pass-through behaviour type of the virtual service instance. The value set overrides the setting from the stub definition.</p> <p>This option can be set in the <b>Behavior</b> tab from the UI.</p>	<code>stub.pass.through.behavior</code>	<p>Valid values depend on the stub operation transport type but can be as follows:</p> <ul style="list-style-type: none"> <li>• <i>simulate_error</i></li> <li>• <i>discard</i></li> <li>• <i>pass_through</i></li> </ul>	<i>simulate_error</i>	This option returns an error to the calling system. The message is not passed to the system under test.
<p>To specify the delay in passing through from the virtual service instance. The value set overrides the setting from the stub definition.</p>	<code>stub.pass.through.delay.period</code>	Any number of seconds in milliseconds.	<i>10000</i>	A delay of 10 seconds in passing through from the virtual service instance.
<p>To specify the status code to use in the virtual service instance response when simulating an error. The value set overrides the setting from the stub definition.</p>	<code>stub.pass.through.status.code</code>	You must specify a valid HTTP status code as the value. For example, <i>503</i> .	<i>500</i>	The HTTP error code of <i>500</i> is sent in the response if the virtual service simulates an error when passing through.

Requirement	Configuration parameter name	Supported values	An example value	Result of using the example value
<p> <b>Note:</b> Applies to HTTP stubs.</p> <p>This option can be set in the <b>Behavior</b> tab from the UI.</p>				
<p>To specify the reason phrase to use in the virtual service instance response when simulating an error. The value set overrides the setting from the stub definition.</p> <p> <b>Note:</b> Applies to HTTP stubs.</p> <p>This option can be set in the <b>Behavior</b> tab from the UI.</p>	<pre>stub.pass.through- .reason.phrase</pre>	<p>You must specify the text of the status message as the value.</p>	<p><i>my custom error msg</i></p>	<p>The text that you specify is set as the HTTP reason phrase in the response if the virtual service simulates an error when passing through.</p>
<p>To specify the type of response-time behavior that is required for the virtual service instance. The value set overrides the setting from the stub definition.</p> <p>This option can be set in the <b>Behavior</b> tab from the UI.</p>	<pre>stub.response.time- .type</pre>	<ul style="list-style-type: none"> <li>• <i>no_delay</i></li> <li>• <i>minimum_delay</i></li> <li>• <i>gaussian</i></li> <li>• <i>uniform</i></li> <li>• <i>performance_profile</i></li> </ul> <p> <b>Note:</b> <i>performance_profile</i> is only valid if the stub</p>	<p><i>no_delay</i></p>	<p>The virtual service sends a response without any delay.</p>

Requirement	Configuration parameter name	Supported values	An example value	Result of using the example value
		 operation in the stub definition is defined to use a performance profile.		
To specify the minimum time for the response time behaviour required for the virtual service instance. The value set overrides the setting from the stub definition.  This option can be set in the <b>Behavior</b> tab from the UI.	<code>stub.response.time-.minimum</code>	Any number of seconds in milliseconds.	<i>5000</i>	The virtual service sends a response after a minimum delay of 5 seconds.
To specify the maximum time for the response time behaviour required for the virtual service instance. The value set overrides the setting from the stub definition.  This option can be set in the <b>Behavior</b> tab from the UI.	<code>stub.response.time-.maximum</code>	Any number of seconds in milliseconds.	<i>20000</i>	The virtual service must send a response after a maximum delay of 20 seconds.

### Virtual services where the stub definition refers to multiple operations

When a stub definition has two or more events that refer to different operations and each operation has a different behavior, you must specify the `<operation_id>` suffixed to the configuration parameter name in the **Additional configuration parameters** field.





**Note:** The `<operation_id>` is the internal ID of the operation that is found in the **Documentation** tab when you open the operation in HCL OneTest™ API.

When the stub definition refers to multiple operations and you want to start a run of a virtual service, you must append the `<operation_id>` to the `<parameter_name>` in the **Additional configuration parameters** field.

The configuration parameter names that you must use for the virtual service, which is associated with a particular operation are as follows:

- `stub.pass.through.behavior.<operation_id>`
- `stub.pass.through.delay.period.<operation_id>`
- `stub.pass.through.status.code.<operation_id>`
- `stub.pass.through.reason.phrase.<operation_id>`
- `stub.response.time.type.<operation_id>`
- `stub.response.time.<operation_id>`
- `stub.response.time.maximum.<operation_id>`

For example, if the `<operation_id>` is `7281b750:162483a09f1:-7e68`, the parameter name for the `stub.response.time.type` option that you must specify is `stub.response.time.type.7281b750:162483a09f1:-7e68`.

---

#### Related information

[Configuring a run of a virtual service on page 401](#)

## Running HTTP virtual services without using proxies

When you run HTTP virtual services on HCL OneTest™ Server, the endpoint on which the virtual service is exposed externally uses a host name that includes a generated ID. If a client application is required to call the virtual service directly via its external host name and not via the HTTP proxy, then you might want to determine the host name to configure the client first before you start the virtual service.

### Before you begin

You must have completed the following tasks:

- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See [Managing members and their roles in a team space on page 576](#).
- Ensured that you are assigned a role as a *Project Owner* or *Tester* in the project. See [Managing access to server projects on page 588](#).
- Created a project in a team space on HCL OneTest™ Server and added the repository that contains the HTTP virtual services.
- Completed the prerequisite tasks before you start HTTP virtual services. See [Prerequisites for running HTTP virtual services on page 392](#).
- Selected the branch of the Git repository to view test resources on the **Execution** page.

### About this task

When you installed HCL OneTest™ Server on Ubuntu, the default HTTP virtual service endpoints are of the following form:

```
in-<unique_id>.<ingress_domain>
```

For example, with an ingress domain as **10.1.2.3.nip.io**, a virtual service endpoint might be as follows:

```
in-cd3a755635574c6fa7e264911301821a.10.1.2.3.nip.io
```

When you installed HCL OneTest™ Server on OpenShift, the default HTTP virtual service endpoints are of the following form:

```
<ingress_domain_first_label>-<unique_id>.<remainder_of_ingress_domain>
```

For example, with an ingress domain as **ots.mycluster-123456-0000.region.containers.appdomain.cloud**, a virtual service endpoint might be as follows:

```
ots-cd3a755635574c6fa7e264911301821a.mycluster-123456-0000.region.containers.appdomain.cloud
```

The **<unique\_id>** is generated when the virtual service is started. The full host name can be viewed in the **Routing to** field of the routing rule created for the virtual service on the **Intercept Rules** page. When traffic is routed to the virtual service via the HTTP proxy, a client application is unaware of this host name. If the client application is required to call the virtual services directly, it must be aware of the host name of the virtual service.

To allow a host name to be known before you start the virtual service, you can replace the generated ID with a user defined value that is specified as an additional configuration parameter when starting the stub. This value is referred to as **<custom\_id>**.



#### Notes:

- The value of the **<custom\_id>** that you specify is used as the **<unique\_id>** for the virtual service.
- The **<custom\_id>** must start and end with an alphanumeric character. The characters supported include a-z, 0-9, and a hyphen. No other characters can be used in the **<custom\_id>**. The **<custom\_id>** must not be longer than 32 characters.



**Restriction:** The **<custom\_id>** must be unique to a virtual service that is running. You must not assign the same **<custom\_id>** to another virtual service that is running simultaneously. Doing so might result in an undefined behavior of the virtual service.

For example, on Ubuntu, if you specify the value of the **<custom\_id>** for the virtual service as `stub123-test` and HCL OneTest™ Server has an ingress domain as **10.1.2.3.nip.io**. Then the host name of the virtual service endpoint is as follows:

```
in-stub123-test.10.1.2.3.nip.io
```

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the virtual service resources in the test assets by clicking **Projects > My Projects > *project\_name***.


The **Overview** page is displayed.

3. Click **Virtualization > Resources** in the navigation pane.

The **Resources** page is displayed.

4. Select the branch of the repository that contains the virtual services that you want to run from the list in the **Branch** field.

All virtual services in the selected branch are displayed on the **Resources** page.

5. Identify the virtual service that you want to run.
6. Click the **Execute** icon  in the row of the identified virtual service.

The **Execute virtual service** dialog box is displayed.

7. Select the version of the virtual service that is in the repository that you want to start by performing any of the following actions:

- Expand the list in the **Version** field, find the version of the test resources, and then select the version.

Use the following details about the version of the test resources that are displayed to identify the version that you want:

- Commit message.
- Tags labeled for the version committed.
- The user who committed the version to the repository.
- Relative time of the commit. For example, *2 hours ago* or *3 days ago*.

The list displays the versions of the test resources committed by all users to the branch in the repository. The versions are arranged with the latest version committed followed by the versions committed previously.


- Expand the list in the **Version** field, and then search for the version that you want to select by entering a partial or the complete commit message of that version.

The version that matches the search criteria is displayed and it is selected for the test run.

8. Select the environment that was used to bind the physical and logical resource in the API project, in the **ENVIRONMENT** tab.



**Important:** The configuration that you set for the run in the **Execute virtual service** dialog box is preserved when you run the same virtual service again. The configurations that you set are not


 available to other members when they want to run the virtual service. For example, if you selected an environment, the same environment is selected when you run the virtual service again.

9. Enter a label, if required.

A label that you enter for the test run that helps you to identify the virtual service instance on the **Instances** page. The label that you entered is displayed for the virtual service under the **Labels** column on the **Instances** page. After you have created a label, any member of the project can use that label.

10. Follow the instructions if you want to modify the configurations for the behavior of the virtual service:

- a. Click the **BEHAVIOR** tab, if it is not already open.
- b. Configure or change the settings for the following options:

 **Note:** The settings displayed are the settings that were configured for the virtual service when it was authored in HCL OneTest™ API.

Option	Description	
Performance	The following settings are available for handling of requests by the virtual service:	
	Option	Description
	<b>Optimize performance</b>	If enabled, attempts are made to reduce the amount of processing between the time the virtual service receives a request and the time it sends a response. Specific optimizations depend on the message contents, as in the following examples: <ul style="list-style-type: none"> <li>▪ When the virtual service receives requests, all validations are disabled, and for all XML payloads, the store and filter actions are converted to use XPath expressions.</li> <li>▪ When the virtualization sends responses, any store actions set on a message are disabled, and any XML content is collapsed when the virtual service is compiled instead of being collapsed every time that a response is sent.</li> </ul>

Option	Description											
	<table border="1"> <thead> <tr> <th data-bbox="581 275 727 310">Option</th> <th data-bbox="898 275 1409 310">Description</th> </tr> </thead> <tbody> <tr> <td data-bbox="581 321 727 485"></td> <td data-bbox="898 321 1409 485">The optimization is disabled as the default action for this setting and you can enable the performance optimization if you fully understand the implications of optimizing the performance.</td> </tr> <tr> <td data-bbox="581 495 727 659"><b>Threads</b></td> <td data-bbox="898 495 1409 659">The maximum number of threads used in processing requests received by the virtual service. The default number of threads is <i>10</i>.</td> </tr> </tbody> </table>	Option	Description		The optimization is disabled as the default action for this setting and you can enable the performance optimization if you fully understand the implications of optimizing the performance.	<b>Threads</b>	The maximum number of threads used in processing requests received by the virtual service. The default number of threads is <i>10</i> .					
Option	Description											
	The optimization is disabled as the default action for this setting and you can enable the performance optimization if you fully understand the implications of optimizing the performance.											
<b>Threads</b>	The maximum number of threads used in processing requests received by the virtual service. The default number of threads is <i>10</i> .											
Operation	Specifies an operation referenced by the virtual service.											
Response time	<p>Specifies the response time behavior for responses sent by the virtual service for the selected operation.</p> <p>You can modify the value by selecting a value from the following options:</p> <table border="1" data-bbox="581 930 1409 1549"> <thead> <tr> <th data-bbox="581 940 727 976">Option</th> <th data-bbox="735 940 1401 976">Description</th> </tr> </thead> <tbody> <tr> <td data-bbox="581 987 727 1077">No delay</td> <td data-bbox="735 987 1401 1077">Select this option for a response with no delay.</td> </tr> <tr> <td data-bbox="581 1087 727 1203">Minimum delay</td> <td data-bbox="735 1087 1401 1203">Select this option for a delay in the response by entering the required delay in milliseconds (ms).</td> </tr> <tr> <td data-bbox="581 1213 727 1381">Uniform distribution</td> <td data-bbox="735 1213 1401 1381">Select this option for a uniformly distributed response time by specifying the minimum and maximum delay in milliseconds (ms).</td> </tr> <tr> <td data-bbox="581 1392 727 1549">Gaussian distribution</td> <td data-bbox="735 1392 1401 1549">Select this option for a response time with Gaussian distribution by specifying the minimum and maximum delay in milliseconds (ms).</td> </tr> </tbody> </table>		Option	Description	No delay	Select this option for a response with no delay.	Minimum delay	Select this option for a delay in the response by entering the required delay in milliseconds (ms).	Uniform distribution	Select this option for a uniformly distributed response time by specifying the minimum and maximum delay in milliseconds (ms).	Gaussian distribution	Select this option for a response time with Gaussian distribution by specifying the minimum and maximum delay in milliseconds (ms).
Option	Description											
No delay	Select this option for a response with no delay.											
Minimum delay	Select this option for a delay in the response by entering the required delay in milliseconds (ms).											
Uniform distribution	Select this option for a uniformly distributed response time by specifying the minimum and maximum delay in milliseconds (ms).											
Gaussian distribution	Select this option for a response time with Gaussian distribution by specifying the minimum and maximum delay in milliseconds (ms).											
Passthrough	<p>Specifies the pass through behavior for the selected operation when requests are not handled within the virtual service.</p> <p>You can modify the value by selecting a value from the following options:</p>											

Option	Description	
	Discard	This option stops the system under test from receiving the intercepted message. This option can disrupt the calling system. For example, the system might time out while it waits for a reply.
	Pass Through	This option passes the intercepted message to the system under test, with an optional delay.
	Simulate Error	This option returns an error to the calling system. The message is not passed to the system under test.

11. Click **Advanced** to make the following advanced configurations:

- a. Enter the following configuration options as parameters and their values in the **Additional Configuration Parameters** fields.

Parameter name	Value
<b>stub.ingress.host.identifier</b>	<custom_id>


12. Click **Execute**.

## Results

You have started an HTTP virtual service that does not use a proxy. The virtual service exposes the host name that contains a custom ID that you specified.

## What to do next

After you start a run of a virtual service, you can view the status of the virtual service in any of the following ways:

- On the **Resources** page, the virtual services are displayed with the state as `Running` along with the number of instances that are running in the **Active instances** column.
- On the **Resources** page, you must click the **Show in instances page** icon  in the row of the virtual service to view all the running instances of a particular virtual service on the **Instances** page.
- Go to the **Instances** page by clicking **Virtualization > Instances**. You can view instances of all the virtual services that are running and are displayed with the state as `Running` under the **State** column. You can also view the requests that are received by the virtual service. The number of requests is displayed as number of hits when you hover the cursor over the text in the **Activity** column.

## Viewing running instances of virtual services

After you start virtual services from the **Resources** page, you can view the instances of the virtual services that are running from the **Instances** page in HCL OneTest™ Server. You can view the details, configuration settings, routing rules, usage statistics, or logs of a specific running instance of the virtual service. You can also stop the running instance.

## Before you begin

You must have completed the following tasks:

- Been assigned a *Member* or *Project Creator* role in a team space.
- Been assigned the *Viewer*, *Tester*, or *Owner* role in the project so that you can view the virtual services that are running.
- Started the virtual service resources as a *Tester* or *Owner* from the **Resources** page.

## About this task

After you started the virtual services from the **Resources** page on HCL OneTest™ Server, you can go to the **Instances** page and you can perform the following tasks:

- View all running instances of the virtual services that are contained in the project repositories.
- View the components in the system model that are associated with the virtual service instances.
- Search for a specific instance of the virtual service for which you want to view the details.
- Create or use saved filters for viewing specific running instances of the virtual service.
- View the details, configuration settings, routing rules, usage statistics, or logs of a specific running instance of the virtual service.
- Change the behavior or logging level of a running instance of the virtual service before you run tests on the virtual service.
- Stop a specific running instance of the virtual service.
- Stop multiple instances or all instances of the virtual service that are running.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the virtual service resources in the test assets by clicking **Projects > My Projects > *project\_name***.

The **Overview** page is displayed.

3. Click **Virtualization > Instances** in the navigation pane.

The **Instances** page is displayed.

Any one of the following views is displayed:

- No instances of virtual services are displayed if there are no instances that are running or were stopped in the past hour.
- All running instances of virtual services in your project are displayed.
- No running instances of virtual services are displayed but the option **Show all** is enabled. You can click **Show all** to display the instances that were stopped in the past hour. Instances that were hidden from the display when **Hide inactive** was clicked are displayed.



**Note:** All instances that were stopped are automatically removed from the display after 60 minutes of being stopped. You cannot use the **Instances** page to view the history of stopped instances. You can hide the stopped instances from the display by clicking **Hide inactive**, so that you can view only the running instances.


You can find the following information about the running instances that are displayed on the **Instances** page.

- The name and path of the virtual service in the project repository are displayed in the **Name** column.
- The tags or labels added to the instance when the run was configured is displayed in the **Labels** column.
- The name of the environment associated with the operation in which the virtual service was created in HCL OneTest™ API is displayed in the **Environment** column.
- The state of the running instance is displayed as *Running* in the **State** column.
- The activity of the running instance of the virtual service based on the number of requests received in the previous hour is displayed in the **Activity** column. The activities are classified and displayed as indicated in the following table:




Activity load	Activity displayed as
Received 0 requests	None
Received 1 - 10 requests	Low
Received 11 - 100 requests	Mid
Received 101 - 1000 requests	High
Received more than 1000 requests	Very High





- The **Stop** icon  is displayed in the **Actions** column and stops the running instance of the virtual service, when clicked.


4. Complete the steps for the task that you want to perform as listed in the following table:

Task	Action
Searching for a specific instance of the virtual service for which you want to view the details or want to stop the instance.	Enter either the complete text or part of the text in the name or the path of the virtual service that is in the repository in the <b>Search</b> field.  The instances of the virtual services that match the text searched are displayed.
Viewing the component associated with the running instance of the virtual service.	You can view the name of the associated component that is displayed on the <b>Instances</b> page in the <b>Filter</b> row only if you clicked the icon  on the <b>System Model</b> page to view all running instances of the virtual services .



Task	Action
<p>Creating or using saved filters for viewing specific instances of virtual service resources.</p>	<p>To create a query, perform the following actions:</p> <ol style="list-style-type: none"> <li>Click <b>New filter</b>.</li> <li>Create a rule with an appropriate operator.</li> <li>Select criteria such as <i>Name, Path, Labels, Environment, State, or Activity</i>.</li> <li>Select the condition, and then enter the value for the criteria.</li> <li>Apply the filter query.</li> </ol> <p>The virtual services that match the filter criteria are displayed.</p> <ol style="list-style-type: none"> <li>Save the filter query for retrieving it from the saved filters list.</li> </ol> <p>To use a saved filter query, perform the following actions:</p> <p> <b>Note:</b> To open the filter query, you must have created and saved a filter query.</p> <ol style="list-style-type: none"> <li>Click the <b>Open filters</b> icon .</li> <li>Select the saved filter.</li> <li>Apply the filter.</li> </ol> <p>The virtual services that match the filter criteria are displayed.</p>
<p>Viewing the details, configuration settings, routing rules, usage statistics, or logs of a specific running instance of the virtual service.</p>	<p>After you identify the virtual service instance for which you want to view the details, perform the following actions:</p> <ol style="list-style-type: none"> <li>Click the <b>Expand</b> icon  in the row of the identified instance of the virtual service.</li> </ol> <p>The <b>Details</b> panel of the instance is displayed. You can find the following details about the instance displayed:</p> <ul style="list-style-type: none"> <li>▪ Name</li> <li>▪ Path</li> <li>▪ Labels</li> <li>▪ API Environment</li> <li>▪ Version</li> <li>▪ Started at</li> <li>▪ Started by</li> <li>▪ Associated components</li> <li>▪ Documentation</li> </ul> <ol style="list-style-type: none"> <li>Click <b>Behavior</b> to expand the panel and view the behavior that was configured for the virtual service.</li> </ol>

Task	Action
	<ul style="list-style-type: none"> <li>c. Click <b>Routing Rules</b> to expand the panel and view the routing rules that were configured in the virtual service.</li> <li>d. Click <b>Statistics</b> to expand the panel and view the graphical representation of the activities performed by the virtual service.</li> <li>e. Click <b>Diagnostics</b> to expand the panel and view the logging configuration of the virtual service and the links to the logs.</li> </ul>
<p>Changing the behavior or logging level of a running instance of the virtual service.</p>	<p>After you identify the virtual service instance for which you want to view the details, perform the following actions:</p> <ul style="list-style-type: none"> <li>a. Click the <b>Expand</b> icon  in the row of the identified instance of the virtual service.</li> </ul> <p>The <b>Details</b> panel of the instance is displayed.</p> <p>Perform any of the following actions:</p> <ul style="list-style-type: none"> <li>◦ Click <b>Behavior</b> to expand the panel.</li> </ul> <p>Perform the following steps to change the behavior configurations:</p> <ul style="list-style-type: none"> <li>a. Click the <b>Edit</b> icon .</li> <li>b. Select a different operation for the virtual service, if available.</li> <li>c. Select a different response time and enter the values for the delay that you want to set.</li> <li>d. Click <b>apply</b>.</li> </ul> <ul style="list-style-type: none"> <li>◦ Click <b>Diagnostics</b> to expand the panel.</li> </ul> <p>Perform the following steps to change the logging level configuration:</p> <ul style="list-style-type: none"> <li>a. Click the <b>Edit</b> icon .</li> <li>b. Select a different logging level.</li> <li>c. Click <b>apply</b>.</li> </ul>
<p>Stopping a specific running instance of the virtual service.</p>	<ul style="list-style-type: none"> <li>a. Click the <b>Stop</b> icon  in the row of the virtual service instance.</li> <li>b. Click <b>Ok</b> in the <b>Stop virtual service instance</b> dialog box that is displayed.</li> </ul> <p>The running virtual service instance is stopped.</p>

Task	Action
Stopping multiple running instances or all the running instances of virtual services.	<p>a. Perform any of the following actions:</p> <ul style="list-style-type: none"> <li>▪ Select the checkbox in the row of the virtual service instances that you want to stop, and then click <b>Stop selected</b>.</li> <li>▪ Select the <b>Select all</b> checkbox in the header row, and then click <b>Stop selected</b>.</li> </ul> <p> <b>Note:</b> All running instances are selected when you click the <b>Select all</b> checkbox.</p> <p>b. Click <b>Ok</b> in the <b>Stop all selected virtual service instances</b> dialog box that is displayed.</p> <p>The running virtual service instances are stopped.</p>

### Results

You have viewed running instances of the virtual services from the **Instances** page on HCL OneTest™ Server.

### What to do next

You can perform the following tasks:

- View the configurations of a running instance of the virtual service. See [Viewing configurations of running instances of virtual services on page 423](#).
- Modify the behavior or logging level of a running instance of the virtual service before you run tests on the virtual service. See [Modifying configurations of running instances of virtual services on page 428](#).
- Stop a running instance or all running instances of the virtual services. See [Stopping virtual services on page 434](#).

## Viewing configurations of running instances of virtual services

After you start virtual services from the **Resources** page, you can view the details, configuration settings, routing rules, usage statistics, or logs of a specific running virtual service instance from the **Instances** page on HCL OneTest™ Server.

### Before you begin

You must have completed the following tasks:

- Been assigned a *Member* or *Project Creator* role in a team space.
- Been assigned the *Viewer*, *Tester*, or *Owner* role in the project so that you can view the virtual services that are running.
- Started the virtual service resources as a *Tester* or *Owner* from the **Resources** page.

### About this task


After you have started the virtual services from the **Resources** page on HCL OneTest™ Server, you can go to the **Instances** page and view the information about a specific running virtual service instance.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the virtual service resources in the test assets by clicking **Projects > My Projects > *project\_name***.

The **Overview** page is displayed.

3. Click **Virtualization > Instances** in the navigation pane.

The **Instances** page is displayed with all running instances of the virtual services that are in your project.

4. Identify the running instance of the virtual service of which you want to view the details, configuration settings, and other information.
5. Click the **Expand** icon  of the instance of the virtual service.

The **Details** panel of the instance is displayed with the following details about the virtual service instance:



**Note:** The information displayed pertains to the configuration settings that were configured in the virtual service at the time it was created or run.

Option	Information displayed
Name	Displays the name of the virtual service.
Path	Displays the path of the virtual service resource in the project repository.
Labels	Displays the labels or tags that were entered for the virtual service when a run of the virtual service was configured.
API Environment	Displays the name of the environment associated with the operation in which the virtual service was created in HCL OneTest™ API or the environment selected for the run.
Version	Displays the version of the virtual service resource that is in the project repository that was started as an instance.
Started at	Displays the date and time when the virtual service instance started.
Started by	Displays the name of the project member who started the virtual service instance.


Option	Information displayed
Associated components	Displays the components in the system model that are associated with the virtual service.
Documentation	Displays the text that was entered on the <b>Documentation</b> tab of the <b>Stub Editor</b> when the virtual service was created in HCL OneTest™ API.








The other panels that you can view are as follows:





- Behavior
- Routing Rules
- Statistics
- Diagnostics

6. Expand each panel by clicking the panel to view the information in that panel.

You can view the following information in the different panels:

Panel	Details										
Behavior	<p>The <b>Behavior</b> panel of the instance is displayed when expanded with the following details about the virtual service instance:</p> <p> <b>Note:</b> The information displayed pertains to the configuration settings that were configured in the virtual service at the time it was created or run.</p> <table border="1"> <thead> <tr> <th>Option</th> <th>Information displayed</th> </tr> </thead> <tbody> <tr> <td>Performance</td> <td>Displays the optimization setting and the maximum number of threads in use by the virtual service.</td> </tr> <tr> <td>Operation</td> <td>Displays the operations referenced by the virtual service. You can select each operation to view the related response time and the pass through settings.</td> </tr> <tr> <td>Response time</td> <td>Displays the response time option that is in use for the selected operation within the virtual service.</td> </tr> <tr> <td>Passthrough</td> <td>Displays the pass through option that is in use for the selected operation within the virtual service.</td> </tr> </tbody> </table>	Option	Information displayed	Performance	Displays the optimization setting and the maximum number of threads in use by the virtual service.	Operation	Displays the operations referenced by the virtual service. You can select each operation to view the related response time and the pass through settings.	Response time	Displays the response time option that is in use for the selected operation within the virtual service.	Passthrough	Displays the pass through option that is in use for the selected operation within the virtual service.
Option	Information displayed										
Performance	Displays the optimization setting and the maximum number of threads in use by the virtual service.										
Operation	Displays the operations referenced by the virtual service. You can select each operation to view the related response time and the pass through settings.										
Response time	Displays the response time option that is in use for the selected operation within the virtual service.										
Passthrough	Displays the pass through option that is in use for the selected operation within the virtual service.										
Routing Rules	<p>The <b>Routing Rules</b> panel of the instance is displayed when expanded.</p> <p>The routing rules that relate to the virtual service are displayed.</p>										

Panel	Details											
	<b>Column</b>	<b>Description</b>										
	Activity	Indicates the routing type.										
	Target	Displays the endpoint of the target service for which requests are being routed.										
	Recipient	Displays the endpoint of the virtual service that receives the routed requests										
<p>You can expand the panel for each intercept by clicking on the expand icon . You can then find the following information:</p>												
	<b>Tab</b>	<b>Description</b>										
	Details	<p>The <b>Details</b> panel provides the following information:</p> <table border="1" data-bbox="667 947 1409 1743"> <thead> <tr> <th data-bbox="667 947 792 1003">Option</th> <th data-bbox="792 947 1409 1003">Information displayed</th> </tr> </thead> <tbody> <tr> <td data-bbox="667 1003 792 1451">Condition</td> <td data-bbox="792 1003 1409 1451"> <p>Lists the condition that causes the traffic to be routed to a virtual service.</p> <p>You can view the detailed condition of the rule defined for the intercept by clicking the <b>View</b> icon .</p> <p>The detailed condition of the rule is displayed in the <b>Full condition</b> window.</p> <p> <b>Note:</b> You can copy the detailed condition if you intend to use it elsewhere.</p> </td> </tr> <tr> <td data-bbox="667 1451 792 1545">Routing to</td> <td data-bbox="792 1451 1409 1545">Lists the virtual service to which traffic is routed.</td> </tr> <tr> <td data-bbox="667 1545 792 1640">Created by</td> <td data-bbox="792 1545 1409 1640">Lists the user who created the rule.</td> </tr> <tr> <td data-bbox="667 1640 792 1743">Created at</td> <td data-bbox="792 1640 1409 1743">Lists the date and time when the rule was created.</td> </tr> </tbody> </table>	Option	Information displayed	Condition	<p>Lists the condition that causes the traffic to be routed to a virtual service.</p> <p>You can view the detailed condition of the rule defined for the intercept by clicking the <b>View</b> icon .</p> <p>The detailed condition of the rule is displayed in the <b>Full condition</b> window.</p> <p> <b>Note:</b> You can copy the detailed condition if you intend to use it elsewhere.</p>	Routing to	Lists the virtual service to which traffic is routed.	Created by	Lists the user who created the rule.	Created at	Lists the date and time when the rule was created.
Option	Information displayed											
Condition	<p>Lists the condition that causes the traffic to be routed to a virtual service.</p> <p>You can view the detailed condition of the rule defined for the intercept by clicking the <b>View</b> icon .</p> <p>The detailed condition of the rule is displayed in the <b>Full condition</b> window.</p> <p> <b>Note:</b> You can copy the detailed condition if you intend to use it elsewhere.</p>											
Routing to	Lists the virtual service to which traffic is routed.											
Created by	Lists the user who created the rule.											
Created at	Lists the date and time when the rule was created.											

Panel	Details									
	<b>Tab</b>	<b>Description</b>								
	Activity	The <b>Activity</b> panel lists the log of the activities performed in relation to the routing rule. This panel also displays the details of the activity, date and time, and message for that activity.								
Statistics	<p>The <b>Statistics</b> panel of the instance is displayed when expanded.</p> <p>The requests received by the instance of the virtual service in the previous hour are displayed as a graph. The total requests received is indicated in the <b>Total hits</b> field and the activity load in the previous hour are displayed in the <b>Activity</b> field.</p> <p>The view can be refreshed by clicking the <b>Refresh</b> icon .</p>									
Diagnostics	<p>The <b>Diagnostics</b> panel of the instance is displayed when expanded with the following details:</p> <table border="1" data-bbox="522 865 1421 1417"> <thead> <tr> <th data-bbox="522 865 750 919">Option</th> <th data-bbox="750 865 1421 919">Description</th> </tr> </thead> <tbody> <tr> <td data-bbox="522 919 750 1018">Logging</td> <td data-bbox="750 919 1421 1018">Displays the logging level that was configured for the virtual service.</td> </tr> <tr> <td data-bbox="522 1018 750 1285">Init container log</td> <td data-bbox="750 1018 1421 1285">           Displays the link to the log of the Init container.               <b>Note:</b> This logs contains messages that relate to configuring the container which is ready to run the virtual service.         </td> </tr> <tr> <td data-bbox="522 1285 750 1417">Main container log</td> <td data-bbox="750 1285 1421 1417">Displays the link to the log of the main container. This log contains messages logged by the virtual service based on the selected logging level.</td> </tr> </tbody> </table>		Option	Description	Logging	Displays the logging level that was configured for the virtual service.	Init container log	Displays the link to the log of the Init container.   <b>Note:</b> This logs contains messages that relate to configuring the container which is ready to run the virtual service.	Main container log	Displays the link to the log of the main container. This log contains messages logged by the virtual service based on the selected logging level.
Option	Description									
Logging	Displays the logging level that was configured for the virtual service.									
Init container log	Displays the link to the log of the Init container.   <b>Note:</b> This logs contains messages that relate to configuring the container which is ready to run the virtual service.									
Main container log	Displays the link to the log of the main container. This log contains messages logged by the virtual service based on the selected logging level.									

**Results**

You have viewed the information about the configuration, routing rules, usage, or logs of the virtual service instance.

**What to do next**

You can modify the behavior or logging level configurations of the running virtual service instance before you run tests on the instance. See [Modifying configurations of running instances of virtual services on page 428](#).

## Related information

[Configuring a run of a virtual service on page 401](#)

## Modifying configurations of running instances of virtual services

After you start virtual services from the **Resources** page and before you run tests on the virtual service, you might want to modify the configurations for the behavior or the logging level of the running instance.

### Before you begin

You must have completed the following tasks:

- Been assigned a *Member* or *Project Creator* role in a team space.
- Been assigned the *Viewer*, *Tester*, or *Owner* role in the project so that you can view the virtual services that are running.
- Started the virtual service resources as a *Tester* or *Owner* from the **Resources** page.

### About this task


After you have started the virtual services from the **Resources** page on HCL OneTest™ Server, you can go to the **Instances** page and identify the running instance of the virtual service for which you want to modify the configurations. You can modify the configurations for the behavior or logging levels of the running instance of the virtual service.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the virtual service resources in the test assets by clicking **Projects > My Projects > project\_name**.

The **Overview** page is displayed.

3. Click **Virtualization > Instances** in the navigation pane.

The **Instances** page is displayed with all running instances of the virtual services that are in your project.

4. Identify the running instance of the virtual service for which you want to modify the configurations.
5. Click the **Expand** icon  of the instance that you want to modify the configurations.

The **Details** panel of the instance is displayed.

6. Click **Behavior** to expand the panel and perform the following steps to change the configurations:

- a. Click the **Edit** icon .

The options that you can modify are enabled for editing.

- b. Select a different operation for the virtual service, if available, from the list for the **Operation** option.
- c. Select a different response time from the list for the **Response time** option, and then enter the values for the delay that you want to set.

You can modify the value by selecting a value from the following options:



Option	Description
No delay	Select this option for a response with no delay.
Minimum delay	Select this option for a delay in the response by entering the required delay in milliseconds (ms).
Uniform distribution	Select this option for a uniformly distributed response time by specifying the minimum and maximum delay in milliseconds (ms).
Gaussian distribution	Select this option for a response time with Gaussian distribution by specifying the minimum and maximum delay in milliseconds (ms).

- d. Click **apply** to apply the modified settings.
7. Click **Diagnostics** to expand the panel and perform the following steps to change the logging level that is configured:

- a. Click the **Edit** icon .

The **Logging level** option is enabled for editing.

- b. Select a different logging level from the following options.

Option	Description
None	Specifies that the virtual service does not write log messages.
Normal	Specifies that the virtual service writes informational messages.
Debug	Specifies that the virtual service writes informational and debugging messages.

- c. Click **apply** to apply the modified settings.

## Results

You have viewed and modified the configurations of a running virtual service instance from the **Instances** page on HCL OneTest™ Server.

## What to do next

You can run tests that are in an API Suite in your project, which then run on the virtual service instance. See [Configuring an API Suite run on page 314](#).

## Viewing routing rules of the virtual services

When you run virtual services that are created to run on proxies or intercepts and have routing rules defined, you can view the proxies or intercepts and the routing rules from the **Intercept Rules** page.

### Before you begin

You must have completed the following tasks:

- Been assigned a *Member* or *Project Creator* role in a team space.
- Been assigned the *Viewer*, *Tester*, or *Owner* role so that you can view the routing rules defined for the virtual services from the **Intercept Rules** page.
- Started the virtual service resources as a *Tester* or *Owner* from the **Resources** page.

### About this task

After you run the virtual service resources that are in your project on HCL OneTest™ Server, you can view the routing rules for the running instances of the virtual service resources, from the **Intercept Rules** page.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the virtual service resources in the test assets by clicking **Projects > My Projects > project\_name**.

The **Overview** page is displayed.


3. Click **Virtualization > Intercept Rules** in the navigation pane.

The **Intercept Rules** page is displayed.

The agents, proxies, or intercepts that are configured in the virtual service resources that are in your project are displayed.







You can find the following information that is displayed on the **Intercept Rules** page.

Column	Description
Activity	Lists the agent or proxy.
Target	Lists a summary view of the endpoint that was recorded or virtualized.
Recipient	Lists the destination where messages sent to are captured.

4. Expand the proxy or intercept panel by clicking the **Expand** icon .

The details of the proxy or intercept are displayed in the **Details** tab in the expanded panel.

5. Click each tab to view the following information:

Tab	Description										
Details	<p>The <b>Details</b> panel provides the following information:</p> <table border="1" data-bbox="456 327 1419 963"> <thead> <tr> <th data-bbox="456 327 613 382">Option</th> <th data-bbox="613 327 1419 382">Description</th> </tr> </thead> <tbody> <tr> <td data-bbox="456 382 613 793">Condition</td> <td data-bbox="613 382 1419 793"> <p>Lists the condition that causes the traffic to be recorded or routed to a virtual service.</p> <p>You can view the detailed condition of the rule defined for the proxy or intercept by clicking the <b>View</b> icon .</p> <p>The detailed condition of the rule is displayed in the <b>Full condition</b> window.</p> <p> <b>Note:</b> You can copy the detailed condition if you intend to use it elsewhere.</p> </td> </tr> <tr> <td data-bbox="456 793 613 848">Routing to</td> <td data-bbox="613 793 1419 848">Lists the virtual service to which traffic is routed.</td> </tr> <tr> <td data-bbox="456 848 613 903">Created by</td> <td data-bbox="613 848 1419 903">Lists the user who created the rule.</td> </tr> <tr> <td data-bbox="456 903 613 963">Created at</td> <td data-bbox="613 903 1419 963">Lists the date and time when the rule was created.</td> </tr> </tbody> </table>	Option	Description	Condition	<p>Lists the condition that causes the traffic to be recorded or routed to a virtual service.</p> <p>You can view the detailed condition of the rule defined for the proxy or intercept by clicking the <b>View</b> icon .</p> <p>The detailed condition of the rule is displayed in the <b>Full condition</b> window.</p> <p> <b>Note:</b> You can copy the detailed condition if you intend to use it elsewhere.</p>	Routing to	Lists the virtual service to which traffic is routed.	Created by	Lists the user who created the rule.	Created at	Lists the date and time when the rule was created.
Option	Description										
Condition	<p>Lists the condition that causes the traffic to be recorded or routed to a virtual service.</p> <p>You can view the detailed condition of the rule defined for the proxy or intercept by clicking the <b>View</b> icon .</p> <p>The detailed condition of the rule is displayed in the <b>Full condition</b> window.</p> <p> <b>Note:</b> You can copy the detailed condition if you intend to use it elsewhere.</p>										
Routing to	Lists the virtual service to which traffic is routed.										
Created by	Lists the user who created the rule.										
Created at	Lists the date and time when the rule was created.										
Activity	<p>The <b>Activity</b> panel lists the log of the activities performed by the proxy or intercept. This panel also displays the details of the activity, date and time, and message for that activity.</p>										

## Results

You have viewed the routing rules for the virtual service, proxy, or intercept that are running or connected to HCL OneTest™ Server.

## What to do next

You can perform any of the following tasks:

- You can view the usage statistics of the virtual services that are running. See [Viewing usage statistics of virtual services on page 431](#).
- You can stop the running instances of the virtual services. See [Stopping virtual services on page 434](#).

---

Related information

[Management of virtualized services on page 390](#)

## Viewing usage statistics of virtual services

After you start virtual service resources that are in the project repositories, and then run tests on the virtual service, you can view the usage statistics of the virtual services from the **Usage** page on HCL OneTest™ Server.

## Before you begin

You must have completed the following tasks:

- Completed the prerequisite tasks before you configure a run of the HTTP virtual services. See [Prerequisites for running HTTP virtual services on page 392](#).
- Read the prerequisite information about virtualizing the Istio based services. See [Prerequisites for running virtualized Istio based services in HCL OneTest Server on page 394](#).
- Been assigned a *Member* or *Project Creator* role in a team space.
- Started the virtual service resources as a *Tester* or *Owner* from the **Resources** page.
- Been assigned the *Viewer*, *Tester*, or *Owner* role in the project so that you can view the usage details of virtual services.
- Ran API Suites with tests that run on the virtual service resources that are configured.

## About this task

After you run the virtual services that are in a project, you can view details of the usage statistics of the virtual services from the **Usage** page on HCL OneTest™ Server.



**Important:** The settings such as the **Period**, or **From** and **To** that you configure to view the usage details are retained for you during your ongoing session and the settings are retained even when you perform the following activities:

- Log out of the server and log in again.
- Log in after a server restart.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the virtual service resources in the test assets by clicking **Projects > My Projects > *project\_name***.

The **Overview** page is displayed.

3. Click **Virtualization > Usage** in the navigation pane.

The **Virtual Service Usage** page is displayed.


The total requests that were received by all virtual services on the server are displayed as a graph.

The default view displays the usage data of the virtual services that are gathered during the previous week and grouped in an hourly interval. The total requests received from when the virtual service was started are displayed.

The metric data that is reported is for the number of requests that are received by a virtual service. The requests can be requests, messages, or other calls that are received by the virtual service.

- Change the default settings to view the graph for the period for which you want to view the usage.

You can change the default view by selecting an option from the following options:

Option	Description
<i>1 Hour</i>	Displays the usage data of the virtual service that are gathered during the previous hour and grouped in a 20-second interval.
<i>1 Day</i>	Displays the usage data of the virtual service that are gathered during the previous day and grouped in an eight-minute interval.
<i>1 Week</i>	Displays the usage data of the virtual service that are gathered during the previous week and grouped in an hourly interval.   <b>Note:</b> This is the default view.
<i>Range</i>	Displays the usage data of the virtual service that are gathered during the date range specified in the <b>From</b> and <b>To</b> fields. You can select the date and time for both the <b>From</b> and <b>To</b> fields. The data is dynamically grouped in intervals that depend on the range selected.



**Note:** You can refresh the view at any time to present the graph for the latest data by clicking **Refresh**.

- View details of the virtual service by hovering the mouse over any of the data-points to view the details at that point.

The following details about the virtual service are displayed when you hover the mouse over the data-point:

- The date and time of the request received.
- The number of requests received.

## Results

You have viewed the usage details of the virtual services in your project that ran on HCL OneTest™ Server.

## What to do next

You can perform any of the following tasks:

- You can view the routing rules defined in the virtual service, agent, proxy, or intercept. See [Viewing routing rules of the virtual services on page 430](#).
- You can stop the virtual service instances that are running. See [Stopping virtual services on page 434](#).

---

### Related information

[Management of virtualized services on page 390](#)

## Stopping virtual services

After you started instances of the virtual services from the **Resources** page, you must stop the running instance from the **Instances** page when you do not want to use the instance. You can stop a single instance, multiple, or all running instances.

### Before you begin


You must have started the virtual service from the **Resources** page. See [Configuring a run of a virtual service on page 401](#).

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project that contains the virtual service resources in the test assets by clicking **Projects > My Projects > *project\_name***.


The **Overview** page is displayed.


3. Go to the **Instances** page and identify the instance of the virtual service that you want to stop.

You can select the virtual service instances that satisfy the following conditions:

- The virtual service instance must be either in the `In transition Of Running` state.
- The **Stop** icon  in the **Actions** column must be enabled for the virtual service instance.
- The checkbox icon  must be enabled for selection for the virtual service instance.

4. Perform the actions indicated in the following table:

If...	Action...
You want to stop any virtual service instance that is running	Perform any of the following actions: <ul style="list-style-type: none"> <li>◦ Click the <b>Stop</b> icon  in the <b>Actions</b> column of the virtual service instance.</li> </ul> <p>The <b>Stop virtual service instance</b> dialog box is displayed.</p> <ul style="list-style-type: none"> <li>◦ Select the checkbox in the row of the virtual service instance, and then click <b>Stop selected</b>.</li> </ul> <p>The <b>Stop all selected virtual service instances</b> dialog box is displayed.</p>

If...	Action...
You want to stop multiple virtual service instances that are running	<p>Perform any of the following actions:</p> <ul style="list-style-type: none"> <li>◦ Select the checkbox in the row of the virtual service instance, and then click <b>Stop selected</b>.</li> <li>◦ Select the <b>Select all</b> checkbox in the header row, and then click <b>Stop selected</b>.</li> </ul> <p> <b>Note:</b> All running instances are selected when you select the <b>Select all</b> checkbox.</p> <p>The <b>Stop all selected virtual service instances</b> dialog box is displayed.</p>

5. Click **Ok**.

A notification is displayed that the running virtual service instances are stopped.

## Results

You have stopped a single instance, multiple, or all running virtual service instances. The stopped virtual service instance on the **Instances** page is displayed with the status as *Stopped by User* in the **State** column.

## What to do next

You can restart the virtual service by completing the configurations that you want from the **Resources** page.

You can open and view the `Main container log` of the stopped virtual service.

## Test results

After the tests or schedules are run and completed, you can view the results and reports in HCL OneTest™ Server to analyze the verdict, the performance, and statistics. You can also re-execute tests and schedules from the **Result** page with the same commit id.

From the Results page, you can perform the following tasks:

- Searching for test results. You can filter test results by using any of the following ways:
  - By using the **Search** field to search for results by name.
  - By selecting predefined time interval from the **Selection interval** drop-down list.
  - By selecting **To** and **From date** from the **Date Interval** option.
  - By clicking verdict from the **Verdict summary** slider (*Pass, Fail, Inconclusive, or Error*).
  - By creating filter queries.
- Adding labels if you have a tester or an owner role.
- Locking test results: Locked results can be unlocked by the project owner or the project member who locked the results.

- Deleting test results.
- Comparing performance reports.
- Viewing trending reports.
- Viewing multiple reports depending on the test types: Statistics reports, Mobile and Web UI reports, Functional reports, Unified Reports, or HTML reports generated from Postman or JMeter test runs.
- Re-executing tests with the same commit id.

For more details, see the links in the next section of this page.

## Test results and reports overview

HCL OneTest™ Server is a single location for hosting the results and reports of all tests run on different desktop clients and for tests run from the server.

### Test results list

Type	Test	Started by	Started on	Duration	Verdict	Components	Labels	Actions
<input type="checkbox"/>	MultiplySuite StubTestNew/Logical/calcul...	meenal123	Apr 28, 2021 4:49 PM	00:00:23.721	Pass	serviceA, UICompOTS		+ 🔍 ⋮
<input type="checkbox"/>	AddSuite StubTestNew/Logical/calcul...	meenal123	Apr 28, 2021 4:42 PM	00:00:16.312	Fail	ServiceAB, UICompOTS		+ 🔍 ⋮

When you expand the results, the results details and reports cards are displayed.

Results, reports, and logs are generated for tests run from HCL OneTest™ Server, or from desktop clients such as HCL OneTest™ Performance, HCL OneTest™ UI, or HCL OneTest™ API.

You must have configured the desktop clients to publish reports of tests that are run from the desktop client to the HCL OneTest™ Server. For more information about the publishing procedure, refer to the links at the end of this page.

### Test result details

By using the default settings, you can view the following details about the test assets used in the test run in the Details card:


- The status of the test results, the date and time it was executed and completed.
- The details about the Git repository that contains the test assets.
- The execution location that uses the test asset for a test run.





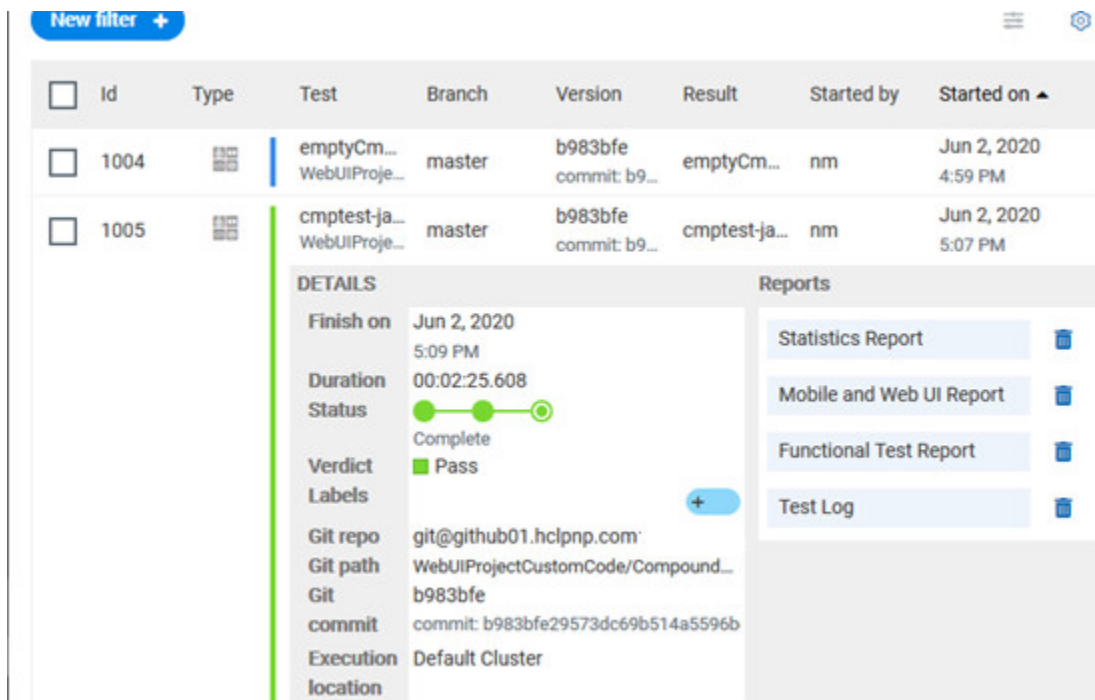
**Note:** The execution location can be the default cluster or the location that you indicate to override a docker host or the agent location when you execute a test. For more information about test configuration, see the link at the end of this page.

- The Resource Monitoring labels (override) used in a performance schedule to control Resource Monitoring sources. For more information about controlling Resource Monitoring sources, see the link at the end of this page.
- The components column displays the components of the system model that are associated with the test execution. You can click the components to view each of them.

You can configure the information displayed in the **Details** card and the **Results** columns from the **View Settings** window. You access this window by clicking the **Settings** icon .

Depending on the attribute you cleared in the **View Settings** window, the Details card displays additional information about the test results.

The selected attributes are displayed in columns of the Result view.



The screenshot shows a test results interface. At the top, there is a 'New filter +' button and a settings icon. Below is a table with columns: Id, Type, Test, Branch, Version, Result, Started by, and Started on. Two test results are listed: 1004 and 1005. Test 1005 is selected, and its details are shown in a card below. The details card is split into 'DETAILS' and 'Reports' sections.

Id	Type	Test	Branch	Version	Result	Started by	Started on
1004		emptyCm... WebUIProje...	master	b983bfe commit: b9...	emptyCm...	nm	Jun 2, 2020 4:59 PM
1005		cmptest-ja... WebUIProje...	master	b983bfe commit: b9...	cmptest-ja...	nm	Jun 2, 2020 5:07 PM

**DETAILS**

- Finish on: Jun 2, 2020 5:09 PM
- Duration: 00:02:25.608
- Status:
- Verdict: Pass
- Labels:
- Git repo: git@github01.hclpnp.com
- Git path: WebUIProjectCustomCode/Compound...
- Git commit: b983bfe
- Execution location: Default Cluster

**Reports**

- Statistics Report
- Mobile and Web UI Report
- Functional Test Report
- Test Log

## Reports

The Reports card contains the links to the test reports and the logs that are displayed in a web browser.

As a default configuration, test logs are delivered in a traditional format for the executed compound tests and schedules. You can still set the `-history jaeger` Program Argument to produce Jaeger traces when you run the tests.

The reports are generated for Compound Tests, Schedules, and Test Suites, or for other tests (Postman or JMeter tests for example). They can be run from a desktop client or from HCL OneTest™ Server.

You can run Web UI tests and SAP tests from HCL OneTest™ UI and publish them to the server from V10.1.3. You can also run and publish these tests from HCL OneTest™ Server.

The following table lists the reports that are generated for each type of test and the links to help pages.

**Table 8. Analyzing Reports**

Report	Tests type	Product	More information
Functional Test Report	API Suite	HCL OneTest™ API	<a href="#">Viewing reports</a>
	Compound Test	HCL OneTest™ UI	
	AFT Suite		
	Functional Test (for SAP test)		<a href="#">Unified Report</a>
Mobile and Web UI Report	Functional test (For Web UI tests)		
	Compound Test	HCL OneTest™ UI and HCL OneTest™ Performance	<a href="#">Mobile and web UI test result reports</a>
	AFT Suite	HCL OneTest™ UI	<a href="#">Unified Report</a>
Statistics Report	Compound Test	HCL OneTest™ UI and HCL OneTest™ Performance	<a href="#">Reports and counters</a>
	AFT Suite	HCL OneTest™ UI	<a href="#">Service Performance report</a>
	Functional test (For SAP and Web UI tests)	HCL OneTest™ UI	From the Statistic reports, you can add counters. Refer to <a href="#">Adding additional counters on a separate page</a> .
	VU Schedule	HCL OneTest™ Performance	You can also display the statistics of counters on graphs. Refer to <a href="#">Displaying counter data in tables or as graphs</a> .
Test Log	Rate Schedule		
	Compound Test	HCL OneTest™ UI and HCL OneTest™ Performance	
	AFT Suite	HCL OneTest™ UI	

**Table 8. Analyzing Reports (continued)**

Report	Tests type	Product	More information
	Functional Test (Web UI and SAP tests)	HCL OneTest™ UI	
	VU Schedule	HCL OneTest™ Performance	
	Rate Schedule		

Related information

[Publishing test results to the server from HCL OneTest API](#)

[Publishing test results to the server from HCL OneTest Performance](#)

[Publishing test results to the server from HCL OneTest UI](#)

[t\\_rm\\_add\\_labels.dita](#)

[c\\_tests\\_config\\_runs\\_p.dita](#)

## Creating search queries to filter test results

To filter the test results list, you can create a search query that you apply as a filter in HCL OneTest™ Server.

### About this task

You can use rules and group of rules in your query. This procedure explains how to create a query that contains two rules and a group of rules as an example, and how to select other filters.

1. Click **New filter** on the Result page.

Follow these steps to add a rule in your query:

2. Select **Add a rule** in your query, and enter or select the required attribute in the fields to define the rule:

- a. Select the first attribute that you want to query in the drop-down list.

**Example**

Select **Verdict**.

- b. Select the condition as *equal to (=)* or *not equal to (!=)*, *Contain* or *Not contain*.

**Example**

Select **=**.

- c. Select a value for the first attribute you selected.

The content of this field depends on the option you selected as first attribute. Enter a value for the **id**, **TypeTest**, **Version**, **Result**, **Labels**, **Branch**, **Test path**, and **Text** attributes. For the other attributes, select a value in the drop-down list.

**Example**

Select **Fail**.

d. Select either the **AND** or the **OR** operator in the first field to add a condition to compare the first rule with another rule.

e. Select **Add rule** in the drop-down list.

f. Select an attribute, a condition, and select or enter a value.

**Example**

Select **Test** as an attribute, select = as a condition operator, and enter *mytest* as a value.

Follow these steps to add a group in your query:

3. Select **Add group** in the drop-down list.



**Note:** A group is a collection of rules and can also contain other groups within a query. You can add a rule or a group within this group.

4. Select a condition operator.

**Example**

Select **Or**.

5. Select **Add rule** in the drop-down list to add a rule in the group.

6. Select the required an attribute, an operator, and enter a value for the rule.

**Example**

Select the **Type** attribute, = as an operator, and **Compound Test** as a value.

7. Select **Add rule** to add another rule in the group.

8. Select and enter the appropriate parameters as you did for the other rules.

**Example**

Select **Type** as an attribute, = as a condition, and **VU Schedule** as a value.

9. Click **Apply** if you want to apply the filter now.

You can cancel the query or save the query when the results are found.

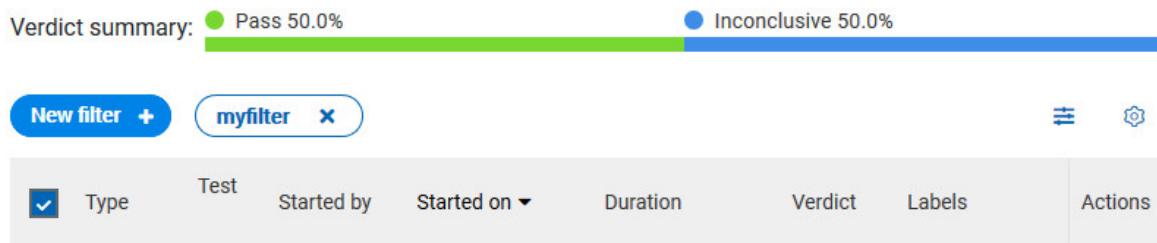
10. Click **Save**.

Proceed as follows to select your filters:

11. Enter a name for the filter and save in the dialog box that is displayed.

### Result

The filter is displayed as a favorite filter.



12. Click the icon to select the filters you created from the list of saved filters.

## Filters

Search for filter

- None
- myfilter

Delete

Apply

- Click a filter in the list of saved filters. If a filter does not display in the list, enter the name of a filter in the search field and select it.



**Note:** You can delete a filter from the list or disable filters by selecting **none** in the list.

- Apply the selected filter.

**Result**


The filter is displayed as a new favorite filter in the **Results** page.

## Comparing test reports

After you run performance tests, you can analyze the difference between two or multiple test results in HCL OneTest™ Server. For example, you can compare the performance of an application at different time slots or different milestone builds between two test results.

**About this task**

You can compare the test runs for performance tests that are in the same project or in different projects.

- Select results in the **Results** page by checking the boxes in the first column.
- Click the  icon



**Note:** When comparing multiple runs, you cannot compare multiple time-ranges or stages.



**Result**


The two or multiple test results are displayed in a browser window in the same report.

## Viewing trending reports

When the performance test runs are complete, you can view the trend of response time for an application over a period of time in HCL OneTest™ Server from trending reports. In addition to the response time, you can view the trend for the loops, transactions, and performance requirements for the application.

### About this task

The trending reports are available for performance tests only.

1. Select results in the first column in the **Results** page.
2. Click the  icon to view the trending reports.



**Note:** When comparing multiple runs, you cannot compare multiple time-ranges or stages.



### Results

You can analyze the trend information for multiple test results in the same report from a browser window.

## Re-executing tests from results

You can re-execute a test, a schedule, a collection of tests or schedules from HCL OneTest™ Server if you want to keep the same commit id.

### Before you begin

You must have executed a test or multiple tests from the **Execution** page in HCL OneTest™ Server.

Re-executing a test from the **Results** view requires a Tester or an Owner role.

## About this task

The following procedure describes how to re-execute a single test or multiple tests at a time.

Perform all the following steps from the Results page.

- Proceed as follows to re-execute one test at a time:

1. Identify the test that you want to re-execute and click the **Re-execute** button in the Actions column.

### Result

The **Execute test asset** dialog box displays the same parameters as the ones that were set to execute the initial test from the **Execution** page.

2. Select another **Version** of the test.



**Note:** If you select another version of the test, the settings configured in the dialog box and in the **Environment**, **Data Sources** and **Locations** tabs might change.

3. Modify the **Schedule** parameters.



**Note:** If you don't want to execute the test now, schedule the time when you want the test to be re-executed.

4. Click the **Environment** tab and select another **API test environment** if multiple environments were initially set for a test that is running an API Suite.

5. Click the **Data Sources** tab and select an override option if multiple data sources were defined for the initial test run.

6. Click the **Variables** tab and add a new variable if you want to re-execute a test asset with a different variable from the one configured in the asset.

7. Click the **Location** tab to override the default cluster location if the test asset has a docker host or static agents configured.

8. Click **Advanced** to modify the advanced parameters that were set for the initial test run.


9. Enter a new **Program Argument** if applicable to the test.



**Note:** If Jaeger is installed with HCL OneTest™ Server, you can enter the `-history jaeger` Program Argument to produce Jaeger traces when you run the tests.

10. Enter a new instance of **Java arguments** if applicable to the test.




11. Enter the environment variables that must be passed to the test run at runtime in the **Environment Variables** field.
12. In the **Resource Monitoring** tab, click the  , and press the Ctrl + Space keys, or enter the initial letter of a label to select a label in the list and override the current sources in the schedule.




**Note:** The **Resource Monitoring** tab is displayed for performance schedules and only if you have enabled the **Resource Monitoring from Service** option from HCL OneTest™ Performance.

The label field is grayed if you don't have any label entered in your current project. To collect data during the test execution, you must first enter labels in the Resource Monitoring sources page from HCL OneTest™ Server.

For more details about controlling Resource Monitoring sources and labels, see the links at the end of this page.

13. Click the **Re-execute**  icon to re-execute the test now or at a scheduled time.
  - Proceed as follows to re-execute multiple tests at a time:
    1. Click the checkboxes for the tests that you want to re-execute.  
You can select them one by one or select all of them.
 

**Result**  
A toolbar is displayed.
    2. Click the **Re-execute** icon  in the toolbar.
 

**Result**  
A message indicates the number of selected tests and the number of tests to be re-executed. The tests are re-executed with the same initial commit ids and parameters.
    3. Click **Execute**.

### Result

You can view the results, logs, and reports of the re-executed tests in the **Results** page.

---

#### Related information

[Adding a project on page 582](#)

[Test results and reports overview on page 436](#)

[Controlling resource monitoring sources in a schedule on page 474](#)

Tests configurations and test runs

## Resource monitoring service

When you apply load to a system under test, the amount of resources consumed by your system is increasing. If the capacity of the resources does not match the load, you can see performance issues in the results. The resource monitoring service in HCL OneTest™ Server helps you monitoring the resources of a system and establish the performance metrics of the system. Thus, you can observe the health of these resources while a schedule is running.

You can use the resource monitoring service while running a schedule to capture data, such as processor or memory usage, or to monitor the availability of hosts and services by using counters. The resource monitoring service can provide a comprehensive view of a system under test to help determine problems. Hosts and services can be servers, virtual machines, or any local host or network services. You can also monitor remote hosts and services with agents.

### Monitoring a local host and service

You can use the resource monitoring service to monitor any local host and network service. In this case, the resources of the hosts are locally monitored. This method is used for host targets that the service can reach directly. With HCL OneTest™ Server, you can also monitor performance metrics that are collected and stored by a monitoring system that is monitoring a host. You can use a Prometheus server or a data collector through an exporter, OpenMetrics exporter, for example.

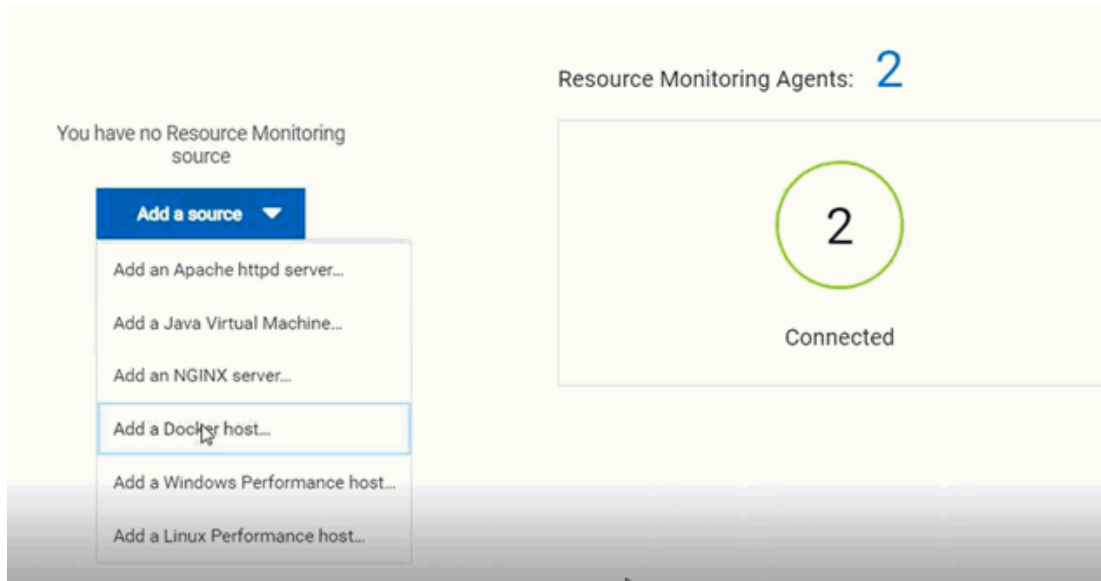
To monitor resources on a local system, you must add the source of the resource monitoring data from the resource monitoring page. Then you have to enter some connection settings and select the performance counters to monitor the sources. The metrics are exposed in a graph for each selected counter.

### Monitoring a remote host with monitoring agents

You can use the resource monitoring service to monitor remote hosts and services with agents through wider sets of data collectors, computers, and networks. The purpose is to capture CPU load, disk space, memory, and the running process for example.

Agent-based monitoring is useful when remote services are not directly accessible through the network. Agents are closer to the target that you want to monitor. You can set up the agent on an authorized host, for example, when access to an **Apache httpd** or **NGINX** server status page or a **JVM JMX** port is restricted to one or few client hosts only. The configuration task is simplified, and no security changes are required.

To monitor a host across monitoring agents, you must first set up the resource monitoring agents on the target host for which you want to collect the performance statistics. The agents establish a connection with the resource monitoring service. When the connection is set, the agent is showing up in the list of sources on the resource monitoring main page. You can select it, choose performance counters, and view the metrics statistics in a graph for each selected counter.



### Resource monitoring sources

With HCL OneTest™ Server, you can monitor resources for the following sources:

- **Apache httpd server**
- **NGINX and NGINX Plus**
- **Java Virtual Machine:** You can monitor JVM resources from a local or a remote system. Some parameters must be set in the command before running the Java Virtual Machine. For more details, see the link to the page about starting a Java Virtual Machine on this page.
- **Windows Performance host:** To monitor the performance of a Windows host, you must have installed an agent on the target Windows host.

If HCL OneTest™ Server is installed on a Linux system, and you want to check the performance of the Windows host, you must have installed an agent on the target Windows host to start monitoring its performance.

- **Linux Performance host:** To monitor the performance of a Linux host, you must have installed an agent on the target Linux host.

If HCL OneTest™ Server is installed on a Windows system and you want to check the performance of the Linux host, you must have installed the resource monitoring agent on the Linux system.

- **Docker host:** A resource monitoring agent is mandatory to get a Docker Data Collector. The Docker host source is added to the list of resource monitoring sources when the agent is installed.
- **Prometheus server:** You can monitor metrics of a host under test that are collected by a Prometheus server. Prometheus collects metrics from monitored targets by regularly requesting appropriate HTTP endpoints on these targets (called *scraping*).

The metrics data are collected through data collectors sets (groups of performance counters) that are tracking the system performance. The performance data results are stored under a Prometheus REST API so that they can be consumed by external systems.

With the resource monitoring service in HCL OneTest™ Server, you can query these performance metrics collected by Prometheus servers. 'PromQL' is the language for querying Prometheus metric data.

In the resource monitoring service, you can select default queries, or create additional ones.

A Prometheus server is required and it must be configured to scrap a set of exporters, every 15 seconds by default. Pushgateway, service discovery, and Alertmanager are optional.

- **OpenMetrics exporter:**

You can monitor the metrics of a host under test that are collected and exposed through an OpenMetrics format or a Prometheus exporter format.

The resource monitoring service in HCL OneTest™ Server is given the ability to scrap metrics exposed by an OpenMetrics or Prometheus exporter through metric counters. No Prometheus server installation is required, you only need to set up one or multiple OpenMetrics, or Prometheus exporters that fit your software or host target. Refer to the link at the end of this page to see the list of exporters.



**Note:**

- You must be a *Team Space Owner* to create, modify and remove resource monitoring sources in a team space. Any team space *Member* (not owner) can only view resource monitoring sources.
- You must be a project *Owner* to create, modify and remove resource monitoring sources in a project. Any project *Member* (not the owner) can only view resource monitoring sources.

Click the links in the next section of this page for more details on the resource monitoring tasks that you can perform in HCL OneTest™ Server.

---

Related information

[Exporters and Integrations](#)

[Starting a Java Virtual Machine on page 471](#)

## Resource monitoring capabilities

With HCL OneTest™ Server, you can monitor a host or a service that you add to a project as a source. You can select the statistics counters that are used to collect metrics and the resource usage data throughout the monitoring system. You can visualize metrics and data that are gathered during the monitoring process in HCL OneTest™ Server.

### Requirements:

- You must be a *Team Space Owner* to create, modify and remove resource monitoring sources in a team space. Any team space *Member* (not owner) can only view resource monitoring sources.
- You must be a project *Owner* to create, modify and remove resource monitoring sources in a project. Any project *Member* (not the owner) can only view resource monitoring sources.

### Supported sources

With HCL OneTest™ Server, you can monitor resources for the following sources:

- **Apache httpd server:** You can optionally use agents.
- **NGINX and NGINX Plus:** You can optionally use agents.
- **Java Virtual Machine:** You can optionally use agents.

You can monitor JVM resources from a local or a remote system. Some parameters must be set in the command before running the Java Virtual Machine. For more details, see the link to the page about starting a Java Virtual Machine on this page.

- **Windows Performance host:**

To monitor the performance of a Windows host, you must have installed an agent on the target Windows host

or on a Windows host that is configured to access the performance data of the Windows host target.

- **Linux Performance host:**

To monitor the performance of a Linux host, you must have installed an agent on the target Linux host.

- **Docker host:**

A resource monitoring agent is mandatory to get a Docker data collector. The Docker host source is added to the list of resource monitoring data collectors when the agent is installed.

- **Prometheus server:**

You can monitor metrics of a host under test that are collected by a Prometheus server. Prometheus collects metrics from monitored targets by regularly requesting appropriate HTTP endpoints on these targets (called *scraping*).

The metrics data are collected through data collectors sets (groups of monitoring counters) that are tracking the system performance. The performance data results are stored in the Prometheus database so that they can be consumed by external systems through a REST API.

In HCL OneTest™ Server, you can query these performance metrics collected by Prometheus servers. 'PromQL' is the language for querying Prometheus metric data. Refer to Prometheus PromQL documentation for information about this query language.

You can select default queries, or create additional ones.

A Prometheus server is required. It must be configured to scrap a set of exporters, every 15 seconds by default. Pushgateway, service discovery, and Alertmanager are optional.

If HCL OneTest™ Server cannot directly reach the Prometheus server, you can consider setting up an agent on an authorized host or on the same host that is running Prometheus.

- **OpenMetrics exporter:**

You can monitor the metrics of a host under test that are collected and exposed through an OpenMetrics format or a Prometheus exporter format.

HCL OneTest™ Server can scrap metrics exposed by an OpenMetrics or Prometheus exporter through metric counters. No Prometheus server installation is required, you only need to set up one or multiple OpenMetrics, or Prometheus exporters that fit your software or host target.

If HCL OneTest™ Server cannot directly reach the Prometheus server, you can consider setting up an agent on an authorized host or on the same host that is running Prometheus.

## Monitoring host resources

To monitor host resources, you must add the monitoring sources to the resource monitoring service, enter connection settings and select performance counters that are used to capture the performance statistics.

### Before you begin

You must be logged in HCL OneTest™ Server and be the owner of an existing new project.

If you monitor a remote host, it must be connected with the computer that you use to access resource monitoring service. See [Installing resource monitoring agents on remote hosts on page 463](#).

### About this task

This task applies to NGINX and NGINX Plus, Apache httpd server, Java Virtual Machine, Windows Performance host, Linux Performance host, and Docker host sources.

1. In the resource monitoring page, click **Add a source**, for example: **Add an NGINX server**.
2. In the **New server** dialog box, fill in the following connection settings:

- a. In **Target host**, enter the IP address or host name and port number of the host where the server to monitor is installed.
- b. In **Server edition**, select the appropriate server from the drop-down list.
- c. In **Path to the status page of the server**, enter the name of the page to view the status of the server.

If you select NGINX Plus (with API version 3) as a source, you must specify the name of the path to view the API root of the NGINX Plus server.

- d. In **Security**, select the following options:

**Secured with TLS/SSL**

If the application server is secured with TLS/SSL.

**Trust self-signed certificate**


To accept the server certificate.

**Do not verify host name**

To ignore verification of the host name in the certificate.

**Require credentials**

If the server requires log-in credentials, enter a **User name** and **Password**.

 **Note:** For remote hosts that are already connected to HCL OneTest™ Server through an agent, you have only the **Access target from** field enabled.

## New NGINX server

Access target from

Target host

Server edition

Path to the status page of NGINX Open Source

Security

Secured with TLS/SSL

Trust self-signed certificate

Close

Reset

Add

e. Click **Add**.

3. Select and save the resource counters to monitor. You can select them from the list where they follow the server logical organization.





## Choose Data Counters

Choose the data to be collected from **NGINX**  
**demo.nginx.com:80 (3)**

Monitoring data extracted from server status page

- Current number of active client connections
- Current number of accepted client connections
- Current number of handled client connections
- Current number of client requests
- Current number of connections where nginx is reading the request
- Current number of connections where nginx is writing the response
- Current number of idle client connections waiting for a request

Cancel

Reset

Save

For a faster selection, select the counters from the built-in sets drop-down list where they are organized by theme and save your selection.

HTTP Server Zones (8/8)

HTTP Server Zones Errors (2/2)

▼

---

Connections and Request

HTTP Server Zones

HTTP Server Zones Errors

---

Stream Server Zones

---

Stream Server Zones Errors



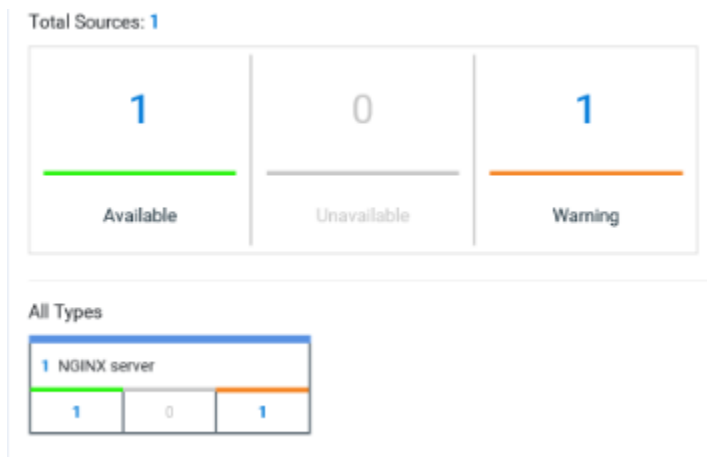
**Note:**

For Docker host, you can select different sets of counters.

- The first level of counters: Generic counters that are related to a Docker Image and all its running Docker Containers (in an exit, running or paused, created, restarted, removing, dead or transitive state).
- The second level of counters: Specific counters that are related to the existing Docker Containers.

**Result**

When the selected counters are saved, two tables are displayed in the resource monitoring main page. They contain the total number of sources you have added and the number of sources ordered by type.



**What to do next**

You can click the links in the tables to view the performance metrics of your monitored system.

## Monitoring metrics collected by a Prometheus server

You can monitor your system resources by using metrics data that are collected by a Prometheus server while executing a schedule.

### Before you begin

You must have created a project in HCL OneTest™ Server.

You must have installed and configured a Prometheus server.

Read the concept information about resource monitoring and monitoring Prometheus server source. See [Resource monitoring service on page 446](#)

### About this task

The following procedure describes how to select and create Prometheus queries in HCL OneTest™ Server. You must set a connection between HCL OneTest™ Server and the Prometheus server first to access to the Prometheus queries.

Enter the connection settings, then select queries.

1. Click **Add a source** and select **Add a Prometheus Server**.
2. Enter the connection settings in the **New server** dialog box.
  - a. In **Target host**, enter the IP address or host name and port number of the host where the Prometheus server is installed.
  - b. Enter the **Path to the API root of Prometheus** to point to the Prometheus API where the monitoring data are saved.
  - c. Enter the security parameters if required. For information about this section, see related links.
  - d. Click **Add** to close the dialog box.

### Result

The Prometheus server source is created successfully in the background and a dialog box opens to select the Prometheus data queries.

3. In the **Select Prometheus queries** dialog box, select queries that are available in the list to collect Prometheus metrics data by clicking the checkboxes.

You can reset the selection.

## Select Prometheus queries

Choose the data to be collected from Prometheus at tp-

The screenshot shows a dialog box titled "Select Prometheus queries". At the top right is a blue button labeled "Add a query" with a plus icon. Below it is a table with the following columns: "Name" (with a refresh icon) and "Actions" (with an eye icon and a copy icon). The table contains the following rows:

Name	Actions
<input checked="" type="checkbox"/> HTTP server requests count	<input type="checkbox"/> <input type="checkbox"/>
<input checked="" type="checkbox"/> HTTP server requests seconds	<input type="checkbox"/> <input type="checkbox"/>
<input checked="" type="checkbox"/> Node CPU busy seconds	<input type="checkbox"/> <input type="checkbox"/>
<input checked="" type="checkbox"/> Node disk read bytes	<input type="checkbox"/> <input type="checkbox"/>
<input checked="" type="checkbox"/> Node disk read time	<input type="checkbox"/> <input type="checkbox"/>
<input checked="" type="checkbox"/> Node disk write time	<input type="checkbox"/> <input type="checkbox"/>
<input checked="" type="checkbox"/> Node disk written bytes	<input type="checkbox"/> <input type="checkbox"/>

At the bottom of the dialog are three buttons: "Cancel", "Reset", and "Save".

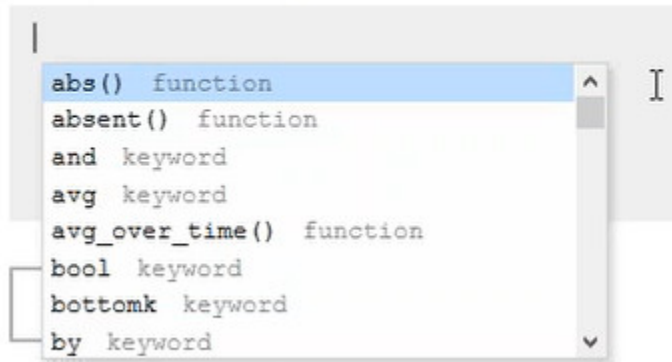
4. Follow these steps to create a query:

- Click **Add a query**.
- Enter a name for the new query in the **Create a Prometheus query** dialog box.
- Enter a PromQL query.any keyword and press Ctrl+Space in the **Type and test your PromQL query**.

### Result

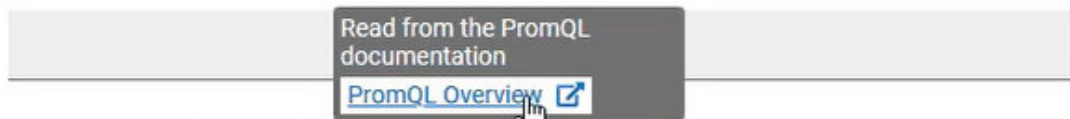
All the possible combinations of keywords and metrics related to your searched keyword are listed.

## Type and test your PromQL query ⓘ

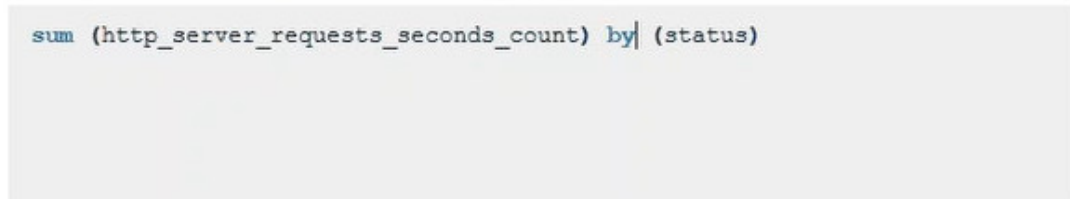



- d. Select a query in the list.
- e. Click the link that points to the 'Querying Prometheus documentation' page in the information tool tip to find other examples of queries in a web page that you can use as samples.

## Name



## Type and test your PromQL query ⓘ



- f. Click **Test query**.  
If the test is successful, a green check is displayed, otherwise you get a red check with an error message.
  - g. Click **Create**.
5. Proceed as follows to duplicate a query:
    - a. Click the **Duplicate** icon  to duplicate a standard query.
    - b. Rename the duplicated query in the **View the Prometheus query** dialog box.  
Make sure that the name you give to the query is not already used.
    - c. Modify the query type.

- d. Test the query .
- e. Click **Create**.



**Note:** You can click Create or Duplicate only if the test of the query is successful and the name of the query is not already in use.

## Create a Prometheus query


Name

JVM Memory

Type and test your PromQL query ?

```
jvm_memory_committed_bytes
```

✔

6. Proceed as follows to modify a query:
  - a. Click the **Modify** icon  .
  - b. Modify the name and type of the query.
  - c. Test the query.
  - d. Click **Update**.
  - e. Delete duplicated or created queries.
  - f. Click **Save** to close the **Select Prometheus queries** dialog box.
7. Click Save and close the **Select Prometheus queries** dialog box.

### Results

The resource monitoring page displays the total number of sources you have added to your project and the number of sources ordered by type in cards.

### What to do next

Click the links in the cards to go to the sources page and see the performance metrics for each source you added to your project. For more details, see [Viewing the performance metrics](#)

---

Related information

[Resource monitoring service on page 446](#)

[Monitoring host resources on page 450](#)

Viewing the performance metrics

## Monitoring metrics exposed by an OpenMetrics exporter

With HCL OneTest™ Server, you can monitor your system resources by using metrics data that are exposed by an OpenMetrics or a Prometheus exporter through metric counters while executing a schedule.

### Before you begin

You must have at least one exporter or software to expose monitoring metrics with an OpenMetrics or Prometheus format.

You must have created a project in HCL OneTest™ Server.

### About this task

The following procedure describes how to select and create OpenMetrics counters in HCL OneTest™ Server. You must set a connection between HCL OneTest™ Server and the OpenMetrics exporter to select counters.

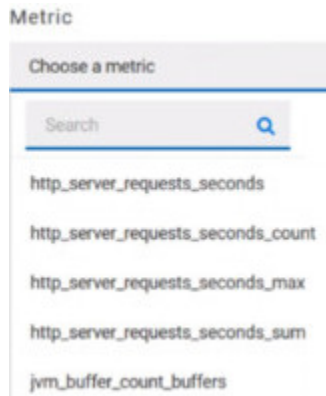
1. Click **Add a source** and select **Add an OpenMetrics exporter** .
2. In the **New OpenMetrics exporter** dialog box that opens, enter the OpenMetrics exporter details to access its exposition endpoint as follows:
  - a. In **Target host**, enter the IP address or host name and port number of the host where the server is installed.
  - b. Enter the **Path to the exposition endpoint** to point to the application where the data performance results must be stored. As the path depends on the type of exporter used, refer to the exporter's documentation and configuration to indicate the right path.
  - c. In **Security**, enter the security parameters if required.
  - d. Click **Add** to close the dialog box.

#### Result

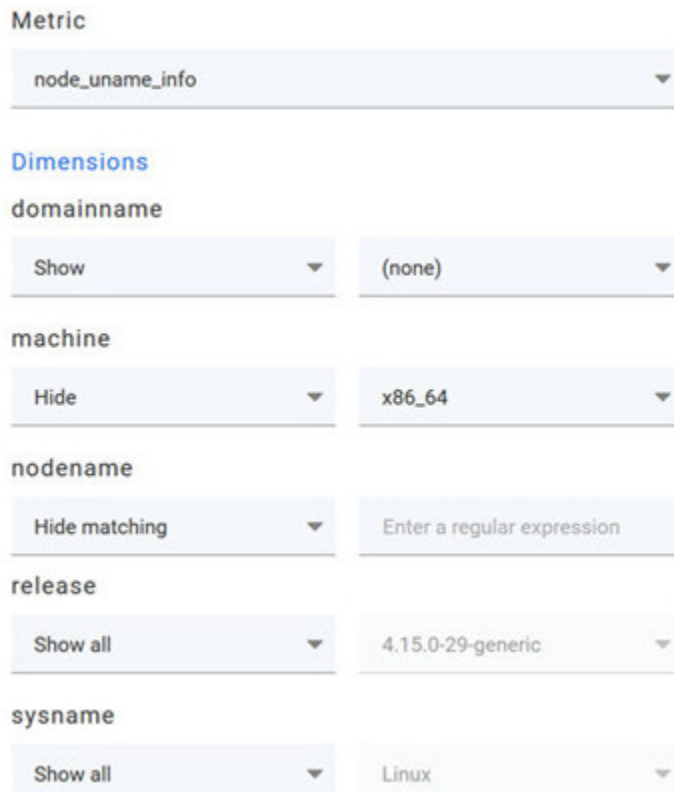
The source is created successfully in the background. A dialog box opens to select the OpenMetrics counters.

3. Click the checkboxes to select counters in the **Select OpenMetrics counters** dialog box.
4. Follow these steps to create a counter:

- a. Click **Add a counter**.
- b. Enter a name for the new counter, in the **OpenMetrics counter**.
- c. Select metrics in the drop-down list or enter a keyword in the **Dimensions** field to enable the dynamic input help. You can also use the search field that appears in the list of metrics.



- d. Select **Dimensions** from the dynamic fields when they are required.



- e. Select the appropriate OpenMetrics counter that you are looking for.
- f. Click **Create**.



## Create an OpenMetrics counter

Name

JVM Memory

Metric

jvm\_memory\_used\_bytes

Dimensions


area


Show all heap,nonheap

id


Show all 6 items selected

Cancel Create

5. Follow these steps to duplicate a counter:
  - a. Select a counter in the list, and click the **Duplicate** icon .
  - b. Modify the name of the counter and the Dimensions.
  - c. Click **Create**.
6. Proceed as follows to modify a new counter:
  - a. Click the **Modify** icon.
  - b. Modify the name, Metrics, and Dimensions of the counter in the **Modify the OpenMetrics counter**.
  - c. Click **Update**.

 **Note:** You can create, duplicate, or update a counter only if the name of the counter is not already used.

7. Click **Save** and close the **Select OpenMetrics counters** dialog box.

 **Note:** You can delete counters, but only the ones that you created or duplicated.

### Results

The resource monitoring page displays the total number of sources you have added to your project and the number of sources ordered by type in cards.

**What to do next**

Click the links in the cards to go to the sources page and see the performance metrics for each source you added to your project. For more details, see [Viewing the performance metrics](#).

---

Related information

[Resource monitoring service on page 446](#)

## Resource monitoring agents

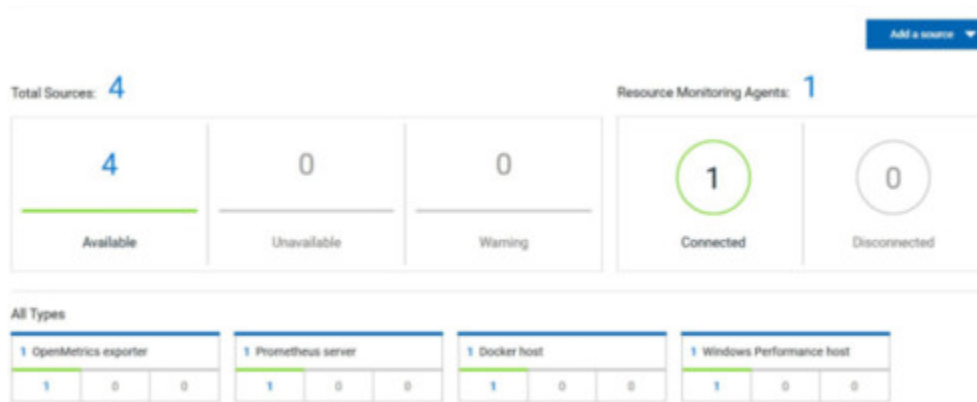
You can connect agents to the resource monitoring service from other hosts to monitor a larger set of data collectors, computers, servers, or systems, or if a server to monitor requires authorization access. To monitor these remote servers, you must configure and run the agents so that a connection is set between HCL OneTest™ Server resource monitoring service and the agents.

For testing, you would need many hosts. For example, you might have one host with the application server, another host with the database server, and some hosts to apply the user load. Due to network or firewall issues, sometimes, it becomes difficult for multiple hosts to connect to each other. resource monitoring agents are installed on the target hosts so that they can establish a connection with HCL OneTest™ Server to gather resource statistics of the target host.

The agents always try to connect with the server through the HTTPS protocol. You must install the agent and start it. From the resource monitoring service page in HCL OneTest™ Server, you can copy the command lines to download files and run the agents. When you stop the agent, the monitoring sources that you have already added persist but the live data will not be available.

You can configure your resource monitoring agent as a service so that the agent automatically runs when the host is restarted. For details, see the documentation pages about starting resource monitoring agents on Windows or Linux.

When the agent is connected to HCL OneTest™ Server, it is added to the main page of the resource monitoring service.



To monitor your host across the agent, you must add the agent as a monitoring source and select performance counters to monitor as described in the monitoring host resources help page.



**Note:**

- To connect resource monitoring agents to HCL OneTest™ Server in a team space, you must be a *Team Space Owner*. Any other team space *Member* (not owner) can only view the resource monitoring agents.
- To connect resource monitoring agents to HCL OneTest™ Server in a project, you must be a project *Owner*. Any project *Member* (not the owner) can only view the resource monitoring agents.

---

Related information

[Resource monitoring service on page 446](#)

[Monitoring host resources on page 450](#)

[Starting a resource monitoring agent as a service on Windows on page 466](#)

[Starting a resource monitoring agent as a service on Linux on page 469](#)

## Installing resource monitoring agents on remote hosts

You must install the resource monitoring agents on the target host for which you want to monitor the performance statistics. You need to run the agents to establish a connection with the resource monitoring service.

### Before you begin

- The resource monitoring service does not require access to the agent host but the agent must have reached the service host over HTTPS.
- The Java agent must have been launched from a jar file and requires a Java 8 virtual machine.
- The Docker agent must have been launched in a Docker container and requires Docker 19.03.

### About this task

This task applies to Java and Docker agents. However, the commands used to install agents are different from Java and Docker agents. You can find the commands and instructions in HCL OneTest™ Server in the resource monitoring agents page.

Before you start the Docker agents, you must build a Docker image.

1. Click **Set up Agents to extend Resource Monitoring service** link in HCL OneTest™ Server to access the agents page where you can find the instructions and commands that are to be used to install and run the agents.

[Add a source](#) ▼

Total Sources: **1**

<b>1</b>	<b>0</b>	<b>0</b>
Available	Unavailable	Warning

No agents are connected to this project

Set up agents to extend Resource Monitoring service

All Types

1 OpenMetrics exporter		
1	0	0

Extend the Resource Monitoring service with agents

- ▶ Context
- ▶ Configure the Java agent
- ▼ **Configure the Docker agent**

1. Install the Docker agent
  - [Download Dockerfile](#)
  - Alternatively, you can use one of the following commands to download the Dockerfile resource:
 

```
curl -O -J https://tp-cicd6.nonprod.hclpnp.com:/rm/agent-Dockerfile
```
  - Or:
 

```
wget --content-disposition https://tp-cicd6.nonprod.hclpnp.com:/rm/agent-Dockerfile
```
2. Build the Docker image
  - Once you have downloaded the Dockerfile on your host system, launch the following command to build a dedicated Resource Monitoring Docker image:
 

```
docker build --network=host --rm -t rmagent .
```
3. Run the Docker container

2. Expand **Configure the Java agent** and click **Download jar file** to download the Java agent, or expand **Configure the Docker agent** and click **Download Docker file** to download a Docker agent.

You can also use the `curl` and `wget` commands to download the agent without accessing the resource monitoring web UI. For more facility, use the code snippets to copy and fill in the commands with the valid offline token and the jar or docker file name.

```
curl -O -J https://hostName/rm/Agent-jar
wget --content-disposition https://hostName/rm/Agent-jar
```

- Copy the build command that is under **Build the Docker image**, paste it in your console and run it to build a resource monitoring Docker image.

```
docker build --network=host --rm -t rmagent
```



**Note:** This step applies to Docker agents only.

- Configure the command as follows to run your agent:
  - Copy the appropriate command for Windows or Linux with the code snippets.
  - Enter the path to the directory that contains the agent `.jar` file, and paste the command.

#### Example

On Windows:

```
set HCL_ONETEST_OFFLINE_TOKEN=(Enter your offline token here)
java -jar (Enter the name of the downloaded jar file here) --ServiceUrl=https://hostName/rm
--projectId=<project_id>
```

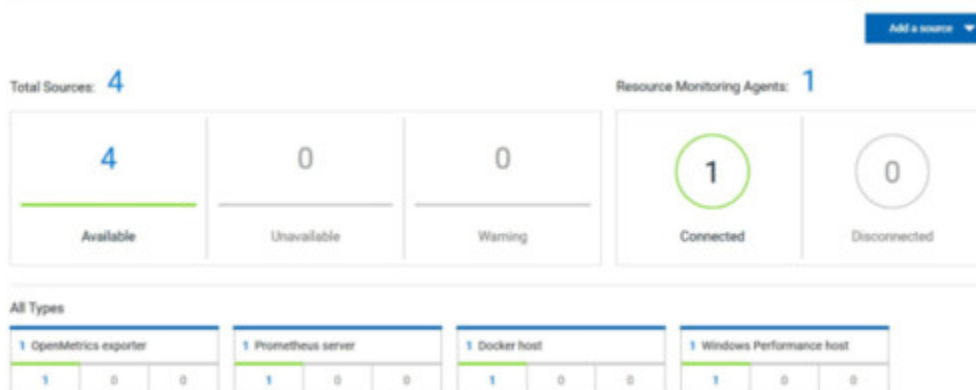
#### Example

On Linux:

```
sudo HCL_ONETEST_OFFLINE_TOKEN=(Enter your offline token here) java -jar (Enter the name of the
downloaded jar file here) --ServiceUrl=portNumber/rm --projectId=<project_id>
```

## Results

When the agent is started, the agent is connected to HCL OneTest™ Server. The agent is displayed in the list of connected resource monitoring agents with the host name and status of the agent. The agent is also added to the main page of the resource monitoring service in HCL OneTest™ Server.



## What to do next

You can add a new resource monitoring source. It can be collected from a service or from the named agents, depending on the capabilities supported by the environment.

## Starting a resource monitoring agent as a service on Windows

To ensure that the resource monitoring agent starts by itself when the host is running, you can set up the environment in such a way that the resource monitoring agent can be started as a service.

### Before you begin

You must have completed the following tasks:

- Installed Java 8 on the host.
- Added HCL OneTest™ Server to the PATH environment variable.
- Created an offline token to connect the agent securely with appropriate permissions.



**Note:** You can create an offline token from the User menu of the resource monitoring page or you can re-use your active offline token. The token expires if it is not used for a month.

To allow remote access to the performance data on Windows 10, you must have considered the following information:

- The user name and password are the same on the local and remote servers. Otherwise you must have provided remote server credentials.
  - The remote user must be a member of the Performance Monitor Users group (start `lusrmgr.msc` and add the user to this group).
  - The Remote Registry service must be running on the remote host (start `services.msc`) and must have verified the Remote Registry status).
  - File and printer sharing must be enabled on the Network Interface of the remote host that is implied in the communication with the local host.
  - You must have activated the following Windows Firewall rules so that the remote firewall does not block the access:
    - File and Printer Sharing (NB-Name-In)
    - File and Printer Sharing (NB-Session-In)
    - File and Printer Sharing (\*)
1. Download the latest release of `winsw`.  
You can choose to download `WinSW.Net2.exe` or `WinSW.Net4.exe` depending on the version of .Net framework that you already have on the host Windows.
  2. Create a folder on your local hard drive like `RMAgent-winservice`.
  3. Copy the downloaded executable to this new folder and rename the file to `RMAgent-winservice.exe`.
  4. Create a new text file in the same folder and name it `RMAgent-winservice.xml`.
  5. Copy and adapt the following content to this new `RMAgent-winservice.xml` file to set up the offline token:

```
<service>
  <id>RMAgent-winservice</id>
  <name>Resource monitoring agent</name>
  <description>This service runs the resource monitoring agent.</description>
  <executable>java</executable>
  <env name="HCL_ONETEST_OFFLINE_TOKEN" value="(Enter your offline token here)"/>

  <arguments>-jar %BASE%\RMAgent.jar --serviceUrl=https://<service-host>/rm
  --projectId=<project-id>
  --autoUpgradeDownloadThen=execute:cmd,/c,start,%BASE%\auto-upgrade.bat</arguments>
  <onfailure action="restart" delay="10 sec"/>
  <logmode>rotate</logmode>
</service>
```

**Notes:**

- Replace `<service-host>` by the host name of the host that runs HCL OneTest™ Server.
- Replace `<project-id>` which is the number that you find after `/projects/` in the browser's URL when you browse to this project.

6. Create a new text file in the same folder and name it `auto-upgrade.bat`.

7. Copy the following content to this new file to upgrade the agent .jar file automatically:

```
@echo off
for /f "tokens=*" %%a in ('dir /b /od %BASE%\com.hcl.test.rm.agent-*.jar') do set newest=%%a
%BASE%\RMAgent-winservice.exe stop
del %BASE%\RMAgent.jar
mklink %BASE%\RMAgent.jar %BASE%\%newest%
%BASE%\RMAgent-winservice.exe start
```

8. Start the command prompt as an administrator and change the directory to the newly created directory:

```
RMAgent-winservice.
```

9. Download the agent .jar file that is available from the agents page in the resource monitoring service, under the **Extend the Resource Monitoring service with agents** section and save it to the same directory.

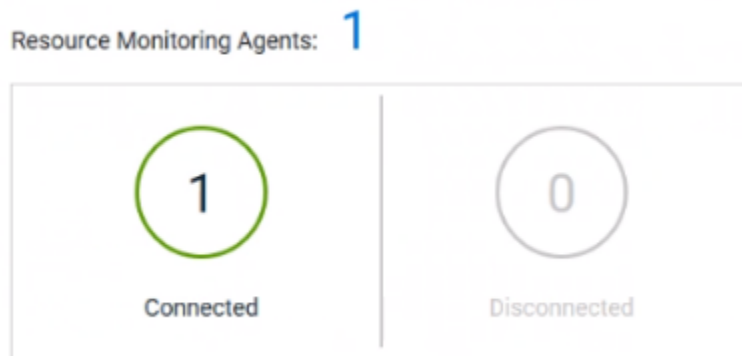
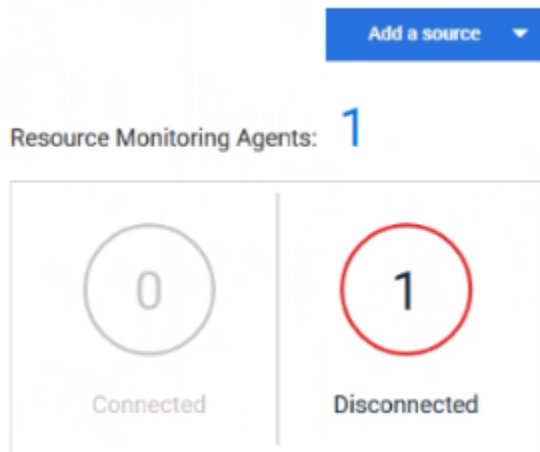
10. Enter the following command in the command prompt to create a symbolic link named `RMAgent.jar` to the agent jar file:

```
mklink RMAgent.jar com.hcl.test.rm.agent-<version-and-datetime>.jar
```

11. Enter the following commands:

```
RMAgent-winservice install
```

```
RMAgent-winservice start
```



 **Notes:**

In the Services Windows application, you can check whether the service is up and running. You can see in the resource monitoring page whether the resource monitoring agent is disconnected or connected.

The resource monitoring agents write the logs to the same folder in files named `RMAgent-winservice.out.log`, `RMAgent-winservice.err.log` and `RMAgent-winservice.wrapper.log`. If the Windows service for the agent is not started or if it is not connected to the resource monitoring service, verify the log files.

**Results**

The Windows agent is added to the main page of the resource monitoring service in HCL OneTest™ Server.

**What to do next**

You can monitor the Windows agent by selecting counters. See [Monitoring host resources on page 450](#).



Related information

[Other commands](#)

[winsw file](#)

## Starting a resource monitoring agent as a service on Linux

To ensure that the resource monitoring agent starts by itself when the host is restarted, you can set up the environment in such a way that the resource monitoring agent can be started as a service.

### Before you begin

You must have completed the following tasks:

- Installed Java 8 on the host.
- Added HCL OneTest™ Server to the PATH environment variable.
- Created an offline token to connect the agent securely with appropriate permissions.



**Note:** You can create an offline token from the User menu of the resource monitoring page or you can re-use your active offline token. The token expires if it is not used for a month.

### About this task

This topic relies on systemd services that are the default on most modern Linux distributions. Other ways may require adaptations of the instructions but the provided script will be a good basis in most cases.

1. Create a folder on your local hard drive like `/opt/RMAgent-linuxservice`.
2. Download the agent .jar file that is available from the Agents page in the resource monitoring service, below the **Extend the Resource Monitoring service with agents** section and save it to the same directory.
3. Create a new file `/etc/systemd/system/RMAgent-linuxservice.service`.
4. Add the following content to this new file:

```
[Unit]
Description = Resource monitoring agent
After = network.target

[Service]
Type = forking
ExecStart = /opt/RMAgent-linuxservice/RMAgent-linuxservice.sh start
ExecStop = /opt/RMAgent-linuxservice/RMAgent-linuxservice.sh stop
Restart = on-failure
RestartSec = 10

[Install]
WantedBy = multi-user.target
```

5. Create another file `/opt/RMAgent-linuxservice/RMAgent-linuxservice.sh`.

```

#!/bin/sh

#!/bin/sh

# Update the 3 following variables with the Server's host name, project id and offline token:
SERVICE_URL=https://<hostname>/rm
PROJECT_ID=<project-id>
export HCL_ONETEST_OFFLINE_TOKEN=<offline-token>

ARGS="--serviceUrl=$SERVICE_URL --projectId=$PROJECT_ID --autoUpgradeDownloadThen=exitFailure"
SCRIPT=$(readlink -f "$0")
RMAGENT_HOME=$(dirname "$SCRIPT")

# Ensure we're using the latest downloaded jar file
PATH_TO_JAR=`ls -t $RMAGENT_HOME/com.hcl.test.rm.agent-*.jar | head -1`
if [ -z "$PATH_TO_JAR" ]
then
cd $RMAGENT_HOME && { curl -k -O -J $SERVICE_URL/agent-jar; cd -; }
PATH_TO_JAR=`ls -t $RMAGENT_HOME/com.hcl.test.rm.agent-*.jar | head -1`
if [ -z "$PATH_TO_JAR" ]
then
echo "Start the server at $SERVICE_URL to allow download of the latest agent jar file"
echo "Exiting..."
exit 1
fi
fi
SERVICE_NAME="Resource monitoring agent"
#Pid file will reside in this script's folder
PATH_TO_PID=$RMAGENT_HOME/RMAgent-pid
#Log file will reside in this script's folder
PATH_TO_LOG=$RMAGENT_HOME/RMAgent.log

case $1 in
start)
echo "Starting $SERVICE_NAME ..."
if [ ! -f $PATH_TO_PID ]; then
nohup java -jar $PATH_TO_JAR $ARGS >> $PATH_TO_LOG 2>&1 &
echo $! > $PATH_TO_PID
echo "$SERVICE_NAME started ..."
else
echo "$SERVICE_NAME is already running ..."
fi
;;
stop)
if [ -f $PATH_TO_PID ]; then
PID=$(cat $PATH_TO_PID);
echo "$SERVICE_NAME stopping ..."
kill $PID;
echo "$SERVICE_NAME stopped ..."
rm $PATH_TO_PID
else
echo "$SERVICE_NAME is not running ..."
fi
;;
restart)
if [ -f $PATH_TO_PID ]; then

```

```

PID=$(cat $PATH_TO_PID);
echo "$SERVICE_NAME stopping ...";
kill $PID;
echo "$SERVICE_NAME stopped ...";
rm $PATH_TO_PID
echo "$SERVICE_NAME starting ..."
nohup java -jar $PATH_TO_JAR $ARGS >> $PATH_TO_LOG 2>&1 &
    echo $! > $PATH_TO_PID
echo "$SERVICE_NAME started ..."
else
echo "$SERVICE_NAME is not running ..."
fi
;;
esac

```

**Notes:**

- Replace `<service-hostname>` by the host name of the host that runs HCL OneTest™ Server.
- Replace `<project-id>` which is the number you'll find after `/projects/` in the browser's URL when browsing to this project.

**Results**

The resource monitoring agent starts automatically when the host restarts. The Linux agent is added to the main page of the resource monitoring service in HCL OneTest™ Server.

**What to do next**

You can monitor the Linux Performance host source by adding counters. See [Monitoring host resources on page 450](#).

---

Related information

[systemd services](#)

## Starting a Java Virtual Machine

To monitor the resource monitoring data from a Java Virtual Machine (JVM), you must first start the JVM. You must enter the IP address of the JVM, the IP port, and the security parameters in the command before running the Java Virtual Machine.

1. Enter The IP address of the JVM (local or remote host) and the IP port in the command that is used to run the Java Virtual Machine.
2. **Optional:** You can use the authentication security data to start the virtual machine. In this case, you must enter the name of the password file. You can also enter the name of an access file that might be needed if user privileges are required.

**Exemple**

**Parameters used to launch a JVM without security:**

```
java
-Dcom.sun.management.jmxremote
-Dcom.sun.management.jmxremote.port=9010
-Dcom.sun.management.jmxremote.local.only=false
-Dcom.sun.management.jmxremote.authenticate=false
-Dcom.sun.management.jmxremote.ssl=false
-jar MyapplicationFile.jar
```

**Parameters used to launch a JVM with authentication security:**

```
java
-Dcom.sun.management.jmxremote
-Dcom.sun.management.jmxremote.port=9010
-Dcom.sun.management.jmxremote.local.only=false
-Dcom.sun.management.jmxremote.authenticate=true
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.password.file=jmxremote.password
-Dcom.sun.management.jmxremote.access.file=jmxremote.access (this command is optional)
-jar MyapplicationFile.jar
```

**Related information**

[Resource monitoring agents on page 462](#)

[Monitoring host resources on page 450](#)

## Managing sources

After you add and connect monitoring sources to your project, you might want to modify the sources in your project and the configuration of the sources.


**Before you begin**

You must be a project owner or team space owner to add or delete sources.


**About this task**

In the page where you can view the resource monitoring usage data and metrics, you can also manage your sources.



You can modify or delete a source, add or change statistic counters, add labels to identify the sources and use them when you execute a test to collect resource monitoring results.

1. Click **Infrastructure > Resource Monitoring** in the navigation pane.  
If you have sources and agents in your project, you can see the cards with the number of available sources and agents in your project.
2. Click a link in the card of available sources to open a list of sources.
3. Perform any of the following actions for a monitoring source:
  - Click the **Delete** icon  to delete the monitoring source and click **Delete** to confirm the deletion in the dialog box.

The source is removed from the list of sources.

- Click the **Edit** icon  to change the target host settings.

You can change the target host address, the port, the path to the server status page and the security parameters.

- Click the **Selection** icon  to select additional counters or change the selected counters for other ones. The changes are reflected in the resource monitoring usage data graph.
- Click  to add a label to identify the source.
- Click **Add a source** to add a source to your project from the list of sources.

#### Related information

[Resource monitoring capabilities on page 449](#)

Viewing the performance metrics

## Creating labels


After you connect a host or service to a project, you can add monitoring labels to identify the monitoring sources that are used when you execute a schedule from HCL OneTest™ Server. You can also use labels in any test to collect resource monitoring usage data during a test run for a defined period of time.

### Before you begin

You must have connected sources to your project in HCL OneTest™ Server. See [Resource monitoring capabilities on page 449](#).

### About this task

You must be a project owner to add labels to the resource monitoring sources in a project and you must be a team space owner to add labels in a team space.

1. Click a source that is available in your project, either in the **Total Sources** cards or in the **All Types** cards from the resource monitoring page.
2. Click  under the **Labels** column of a resource monitoring source to add a label.
3. Enter a name for the label and press the Enter key.



**Note:** You can add one label or multiple labels for the same resource monitoring source. A label can contain a maximum of 200 characters.



**Tip:** You can create the same label for different sources that belong to the same logical group to monitor the whole group with only one label.

### Results

The host source is identified by a label.

### What to do next

You can use the created labels in tests to collect resource monitoring results or persistent usage data. See [Controlling resource monitoring sources in a schedule on page 474](#).

## Controlling resource monitoring sources in a schedule

In HCL OneTest™ Server, you can control the resource monitoring sources that are collected in a performance schedule and override them with the labeled resource monitoring sources that are available in your current Server project.

### Before you begin

- You must have performed the following tasks in HCL OneTest™ Performance:
  - Created a performance schedule of **Rate** or **VU** type.
  - Added resource monitoring sources, or the labels of resource monitoring sources to a schedule. For more information about the procedure, refer to [Adding resource monitoring sources to a performance schedule by using labels](#).
  - Enabled HCL OneTest™ Server as resource monitoring service only if you want to replace resource monitoring labels set in HCL OneTest™ Performance with the labels that you created in HCL OneTest™ Server.
- You must have created a project and added resource monitoring sources to your project in HCL OneTest™ Server. For more information about the procedure, see [Monitoring host resources on page 450](#).

### About this task

This procedure describes how to create resource monitoring labels in HCL OneTest™ Server and how to use them to override the source or label settings related to resource monitoring in a performance schedule. The procedure applies to the following use cases:


- To replace the resource monitoring labels that were initially set in a schedule from HCL OneTest™ Performance with the labels that you create in HCL OneTest™ Server.
- To enable resource monitoring sources that were not enabled from HCL OneTest™ Performance. In that case, you can enable resource monitoring sources from the **Execution** dialog box in HCL OneTest™ Server.
- To ignore resource monitoring sources that were set in the schedule and to change them for labels that match sources that are available in the HCL OneTest™ Server project.





**Notes:** You must be a project owner to add labels to the resource monitoring sources in a project and you must be a team space owner to add labels in a team space. You must have a project owner or a tester role access to add labels to a performance schedule in a project.

1. Click a source that is available in your project, either in the **Total Sources** cards or in the **All Types** cards from the resource monitoring page.



2. Click  under the **Labels** column of a resource monitoring source to add a label.

The screenshot shows the source monitoring table with filters for 'Sources', 'All Types', and 'Any Status'. The table has columns for Name, Status, Added, Observed, and Labels. Two rows are visible:

Name	Status	Added	Observed	Labels
> OpenMetrics at to [redacted]	available	7 hours ago	6 hours ago	SUTv2 
> Prometheus at [redacted]	available	7 hours ago	29 minutes ago	SUTv2 

3. Enter a name for the label and press the Enter key.


**Example:** Enter 'SUTv2' in three different resource monitoring sources.



**Note:** You can add one label or multiple labels for the same resource monitoring source. A label can contain a maximum of 200 characters.



**Tip:** You can create the same label for different sources that belong to the same logical group to monitor the whole group with only one label.

4. Navigate to the **Execute** view and click  on a performance schedule in the list of tests to execute a schedule.
5. Click the **Resource Monitoring** tab in the **Execute Test Asset** dialog box.



**Note:** There are different use case scenarios:



- If the **Resource Monitoring from Service** option was not enabled in HCL OneTest™ Performance (see Figure 1), no source is available in HCL OneTest™ Server project.

Figure 1:

**VU Schedule Details**

Category: **Resource Monitoring from Service**

Enable resource monitoring from service

A message indicates in the **Resource Monitoring** tab that Resource Monitoring was not enabled. You can run the schedule with the resource monitoring sources that are created and labeled in HCL OneTest™ Server (steps 1 to 6 in this procedure) by selecting labels that are in your HCL OneTest™ Server project (step 6).

- If the **Resource Monitoring from Service** option was enabled in HCL OneTest™ Performance, and data is collected from the sources you added to the performance schedule in HCL OneTest™ Performance (see Figure 2), the **Resource Monitoring** tab displays the list of these sources that are available in your HCL OneTest™ Server project.

Figure 2:

**VU Schedule Details**

Category: **Resource Monitoring from Service**

Enable resource monitoring from service

Collect from the following sources

Collect from sources matching at least one of the following labels

Title	Polling Time
<ul style="list-style-type: none"> <li>OpenMetrics exporter                             <ul style="list-style-type: none"> <li>OpenMetrics a [redacted]</li> </ul> </li> </ul>	5 Seconds
<ul style="list-style-type: none"> <li>Prometheus server                             <ul style="list-style-type: none"> <li>Prometheus at [redacted]</li> </ul> </li> </ul>	5 Seconds

To override the initial sources with the sources that you created in HCL OneTest™ Server, select the labels that are in your HCL OneTest™ Server project and that match the Server sources (step 6 in this procedure).





- If you added sources to your performance schedule by using labels in HCL OneTest™ Performance (see Figure 3), the **Resource Monitoring** tab displays the list of labels that are available in your HCL OneTest™ Server project.

Figure 3:

**VU Schedule Details**

Category: **Resource Monitoring from Service**

Enable resource monitoring from service

Collect from the following sources

Collect from sources matching at least one of the following labels


OpenM	Add...
Prom	Edit...
SUTv2	Remove

To select resource monitoring labels from your schedule, follow step 6 in this procedure.

In the list of sources or labels that is displayed in the **Resource Monitoring** tab:

- Green icons identify either labels that match the sources or sources that are found in the current project.
- Red icons indicate that the sources are not running,
- Yellow icons identify either labels that do not match the sources, or sources that are not found in the current project.

Data is collected for the sources that match labels only.

6. Click the , and press the Ctrl + Space keys, or enter the initial letter of a label to select a label in the list.

**Example:** Select 'SUTv3'.

ENVIRONMENT    DATA SOURCES    VARIABLES    LOCATION    **RESOURCE MONITORING**

Asset settings: list of labels

- ▲ L1\_ITS: not matching
- ▲ L2\_Proj\_ITS: not matching

To collect data during this execution, you can select existing labels that match sources:


SUTv3 x +

Label selection summary:

- ✔ SUTv3: matching
  - ✔ Prometheus at tp-junit1.nonprod.hclpnp.com:48090
  - ✔ OpenMetrics at tp-junit1.nonprod.hclpnp.com:48080 Defined in team space

Advanced    Reset    Cancel    Execute

You can see all the labels that you added to the resource monitoring sources in HCL OneTest™ Server.

 **Note:** You can select labels matching resource monitoring sources that are created in a team space in addition to the ones that are created in a project.

7. Select the labels that match the sources.
8. Click **Execute** to execute the performance schedule.
9. Click the **Result** view in the navigation bar of HCL OneTest™ Server when the schedule is complete.
10. Expand the performance schedule result to see the resource monitoring label name in the **Details** card.

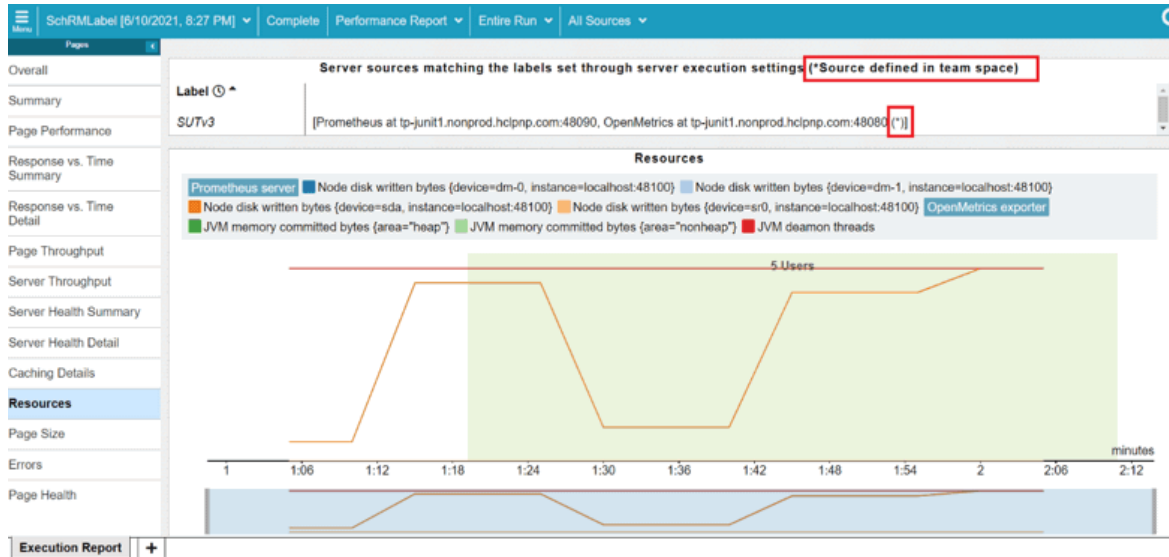
The **Details** card displays the resource monitoring labels that are used in the schedule to override the initial labels and collect resource monitoring sources (in the example, the 'SUTv3' label). For other details about the **Results** page, see the link at the end of this page.

SchedListOfRMSourcesEnabled    dilip    Sep 8, 2020 10:16 AM    00:04:25.153    Pass

DETAILS		Reports	
Id	1751	Statistics Report	
Branch	Dilip_1011	Test Log (Text)	
Result	SchedListOfRMSourcesEnabled		
Finish on	Sep 8, 2020 10:20 AM		
Status	Complete		
Git repo	https://[redacted]testingproducts/PT_SVT_TestAsset.git		
Git path	proj_RM_1011/Schedules/SchedListOfRMSourcesEnabled testsuite		
Git commit	commit sch with RM sources CRC server  d24e819		
	[redacted] committed 22 days ago (#refs/heads/Dilip_1011)		
Execution location	Default Cluster		
Resource Monitoring labels (override)	<span style="background-color: #0056b3; color: white; padding: 2px;">SUTv3</span>		

In the **Report** card, you find the link to the Statistics Report that is generated. In the **Resource** tab of the Statistics Report, you can view the information related to the Server sources matching the labels set in the schedule.

11. Click the **Statistics Report** link in the **Reports** card to see the performance metrics of the monitoring sources matching the selected labels.



Example: You can see the statistics for the sources tagged with the 'SUTv3' label.



**Note:** In the Resources report, each resource name followed by an asterisk identifies a resource monitoring source that is defined in a team space.

#### Related information

[Test results and reports overview on page 436](#)

## Configuration of a change management system

When you use any application to create and track defects (bugs), issues, or other work items, you can configure the application as a change management system on HCL OneTest™ Server. You can then create defects (bugs), issues, or other work items without the need to open your application.

You use HCL OneTest™ Server to run tests for the software that you develop and view their results. When you discover that you might want to raise defects (bugs), issues, or other types of work items for these tests, you are required to open the application that you use. You can only then, create defects (bugs), issues, or other work items in your application.

When you configure your application as a change management system on HCL OneTest™ Server, you can create defects (bugs), issues, or other work items without the need to open your application.


You can find details in the following table about the usage of the applications that you can configure as a change management system on HCL OneTest™ Server:

Issue tracking application	Usage as a change management system	Go to...
Atlassian Jira Software	You can create stories, defects, or tasks as issues in your project in Jira from HCL OneTest™ Server and also launch your project in Jira from HCL OneTest™ Server to track and manage these issues.	<a href="#">Configuration of Jira as a change management system on page 480</a>

## Configuration of Jira as a change management system

You can integrate HCL OneTest™ Server with Jira to manage and create defects after the test run is complete.

You can create multiple defects for a test result from the **Results** page of HCL OneTest™ Server, if required. After you create a defect, you can track the defect from the **Change Management System** card on the **Results** page.

When you do not configure the Jira server in your project and if you click the **Open action menu** icon  from the **Actions** column in the **Results** page, the following message is displayed:

The change management system is not configured for the project. You must contact your project owner to configure Jira for the project. If you are a project owner click [here](#).


If you are a project *Owner*, you can click the link provided in the message to open the **CHANGE MANAGEMENT SYSTEM** tab. You can then configure the Jira server for your project.

The following topics provide more details about integrating HCL OneTest™ Server with Jira and creating defects.

## Task flow for integrating Jira

You must perform these tasks in sequence as listed in the following table. The table also provides you the links to the information about the tasks.

	Tasks	More information
1	Install HCL OneTest™ Server.	<a href="#">Installation of the server software on page 36</a>
2	Create the type of tests that are supported for running on HCL OneTest™ Server.	<a href="#">Tests supported on HCL OneTest Server on page 21</a>

	Tasks	More information
3	Create a project on HCL OneTest™ Server and add repositories to the project to access the test resources available in the respective repository.	<a href="#">Test assets and a server project on page 580</a>
4	Configure Jira to your project on HCL OneTest™ Server where the test resources are available.   <b>Note:</b> You must have the project <i>Owner</i> access to configure Jira.	<a href="#">Configuring the Jira server on page 482</a>
5	Create defects for the tests available in your project.	<a href="#">Creating Jira defects on page 484</a>

Related information

[Configuration of Jira as a change management system on page 480](#)

## Generating public and consumer keys

You must first generate a public key and a consumer key for your project that is available on HCL OneTest™ Server to configure Jira.

### Before you begin

You must have completed the following tasks:

- You must have installed Jira on your computer. For more information about installing Jira, refer to [Installing Jira software](#).
- You must be able to access the Jira server from HCL OneTest™ Server.

### About this task

After you add the URL of the Jira server to your project, you can generate both the public key and the consumer key from HCL OneTest™ Server. You can then share both the keys with the administrator of Jira server.

1. Log in to HCL OneTest™ Server, if you are not already logged in.
2. Open your project.
3. Expand **Configure**, and then click **Project**.
4. Click the **Change Management System** tab, and then click **New System**.

#### Result

The **Configure Change Management System** page is displayed.

5. Enter the URL of Jira server in the **Change Management System** link field, and then click **Add**.

#### Result

A public key and a consumer key are generated and displayed on the **Configure Tracking System** page.

## Results

You have successfully generated both the public and consumer keys.

## What to do next

You can then click **Copy** to copy both the keys and save the keys in your local file. You can then share both the keys with the administrator of the Jira server to setup an application link in Jira.

The administrator of the Jira server then sets up the application link in Jira by using both the keys that you shared. After the application link set up is complete, you can then configure the Jira server for your project on HCL OneTest™ Server. See [Configuring the Jira server on page 482](#).

## Configuring the Jira server

After the administrator of the Jira server sets up the application link in Jira by using both the public and consumer keys, you can configure the Jira server for your project on HCL OneTest™ Server where the tests are available.

### Before you begin

You must have completed the following tasks:

- Generated a public key and a consumer key on HCL OneTest™ Server. See [Generating public and consumer keys on page 481](#).
- Shared both the keys with the administrator of the Jira server to set up an application link in Jira.

### About this task

When you share both the keys with the administrator of the Jira server, the administrator of the Jira server sets up the application link in Jira.

After the completion of setting up of the application link by the administrator of the Jira server, you can then open the **Change Management System** tab of your project and complete the authorization process to configure the Jira server for your project on HCL OneTest™ Server.

### Tasks performed by the administrator of the Jira server

As an administrator of the Jira server, you must perform the following actions to set up an application link on the Jira server:

1. Log in to the Jira server and open the **Configure Application Links** page.
2. Enter the URL of the application, and then click **Create new link**.
3. Verify the URL on the **Configure Application URL** page, and then click **Continue** to open the **Link applications** page.
4. Enter the information for the fields specified as follows on the **Link applications** page, and then click **Continue**:

- **Application Name:** Enter an application name. The application name must be unique. For example, `HCL`.
- **Application Type:** Select the type of application from the **Application Type** drop-down list. The default selection is **General Application**.
- Select the **Create incoming link** checkbox.



**Note:** You need not set other fields when you set up an application link to establish a connection between the Jira server and HCL OneTest™ Server.

5. Enter the information on the **Link applications** page for the fields specified as follows, and then click **Continue** to set up the application link:

- **Consumer Key:** Enter the consumer key shared by the administrator of HCL OneTest™ Server.
- **Consumer Name:** Enter an application name. For example, `HCL_1`.
- **Public Key:** Enter the public key shared by the administrator of HCL OneTest™ Server.

You have successfully set up the application link on the Jira server to establish a connection between the Jira server and HCL OneTest™ Server.

To know more about the application link, refer to the [Configuring the client application as a consumer in Jira](#) section.

1. Open your project.
2. Expand **Configure**, and then click **Project**.
3. Open the **Change Management System** tab.
4. Click the **Open action menu** button from **Jira Integration Details**, and then click **Configure**.

#### Result

The **Authorize User** page is displayed.

5. Complete the authorization process on the **Authorize User** page by performing the following actions:
  - a. Click the authorization URL of the Jira server to open the **Jira authorization** page.
  - b. Log in to your Jira server account with your valid credentials.
  - c. Click **Allow** to authorize the Jira server account to generate an access token.

#### Result

The authorization is successful.

The **Access Approved** page is displayed with a verification code.

- d. Copy the verification code, and then paste the verification code in the **Enter Verification Code** field on the **Authorize User** page.
- e. Click **Save**.

#### Result

The **Configure Change Management System** page is displayed.


6. Select a Jira server project that is available from the **Project** drop-down list, and then click **Update**.



**Note:** The change management system name and the change management system link are already displayed on the **Configure Change Management System** page.

## Results

You have successfully configured the Jira server for your project on HCL OneTest™ Server.

You can see a tick icon  next to the URL of the Jira server on the successful configuration. You can also expand the **Jira Integration Details** section to see the project name of the Jira that you selected.



**Note:** You can delete the URL of the Jira server that you no longer require in your test environment.

## What to do next

You can create defects for the tests available in your project and track the defects from the **Results** page on HCL OneTest™ Server after the test run is complete. See [Creating Jira defects on page 484](#).

## Creating Jira defects

You can create a Jira defect for a test available in your project and track the defect from HCL OneTest™ Server after the test run is complete.

### Before you begin

You must have completed the following tasks:

- Been a member of the project with the *Owner* or *Tester* role to create a defect for a test.
- Configured Jira in HCL OneTest™ Server for your project. See [Configuration of Jira as a change management system on page 480](#).



**Note:** You must have the project *Owner* access to configure Jira.

- Initiated a test run of a test available in your branch from the **Execution** page.


### About this task

After you configure Jira for your project on HCL OneTest™ Server, you can create a Jira defect for a test available in your project and track the defect from the **Results** page after the test run is complete.





**Note:** You can create a Jira defect for a test result from the **Results** page only if you have executed the test on HCL OneTest™ Server.

1. Open your project on HCL OneTest™ Server.
2. Run a test from the **Execution** page.
3. Click **Results** from the left pane to open the **Results** page.
4. Select the test result from the **Results** page.
5. Click the **Jira: Raise Defect** icon  from **Actions**.



**Important:** You can create a Jira defect for a test result only if Jira is configured to your server project. If Jira is not configured with your server project and you click the **Open action menu** button from **Actions** on the **Results** page for a test result, you can see a message is displayed on the **Results** page that states that there is no change management system configured for the project. You must then contact your project *Owner* to configure Jira for the project. Your project *Owner* can then click the link provided in the message to open the project to configure Jira for your project.

### Result

The **Raise defect in Jira** page is displayed with the fields available for your project.

6. Enter the information required for all the fields, and then click **Create**.



**Important:** You can create a story, defect, or task. Creating other issue types are not supported.

### Result

A Jira defect is created.

7. Expand the test result to see the Jira tracking ID details from the **Results** page.



**Note:** You can raise multiple defects for a test, if required.

## Results

You have created a defect for a test result from the **Results** page on HCL OneTest™ Server.

### What to do next

You can now click the Jira tracking ID from the **Results** page to open the defect and view the details. The **Reporter** field on the Jira server page displays the name of the person who has created the defect. To know more about Jira server defects and how it works, refer to the [documentation](#).

---

#### Related information

[Configuration of Jira as a change management system on page 480](#)

## Integrating with other applications

You can integrate certain applications with HCL OneTest™ Server to run tests, view the test results, and create defects.

### Integration plugin compatibility matrix

You can find information about the versions of the integration plugin that are compatible with HCL OneTest™ Server.

The following table lists the versions of the integration plugin that are required to integrate IBM® Engineering Test Management, HCL Launch, Jenkins, and UrbanCode™ Deploy with HCL OneTest™ Server.



**Note:** You must download the required version of the integration plugin from the [HCL® License & Delivery portal](#) based on the existing version of HCL OneTest™ Server. You can then integrate Engineering Test Management, Jenkins, HCL Launch, and UrbanCode™ Deploy with HCL OneTest™ Server.

HCL OneTest™ Server	10.1.0	10.1.1	10.1.2	10.1.3	10.2
Jenkins plugin	HOT-SERV-ER-Jenkins-2.0	HOT-SERV-ER-Jenkins-3.0	HOT-SERV-ER-Jenkins-4.0	HOT-SERV-ER-Jenkins-4.0	HOT-SERV-ER-Jenkins-5.0
UrbanCode™ Deploy plugin	HOT-SERV-ER-UCD-1.0	HOT-SERV-ER-UCD-1.0	HOT-SERV-ER-UCD-1.0	HOT-SERV-ER-UCD-2.0	HOT-SERV-ER-UCD-3.0
HCL™ Launch plugin	NA	HOT-SERV-ER-LAUNCH-1.0	HOT-SERV-ER-LAUNCH-2.0	HOT-SERV-ER-LAUNCH-2.0	HOT-SERV-ER-LAUNCH-2.0

### Integration with Azure DevOps

When you use Azure DevOps for continuous integration and continuous development of your application, you can run tests created for your application and available in a project on HCL OneTest™ Server, in Azure DevOps pipelines by using the *HCL OneTest Studio* extension.

#### Overview

You can use the *HCL OneTest Studio* extension to integrate HCL OneTest™ Server with Azure DevOps.

The *HCL OneTest Studio* extension enables you to select any type of test available for your project in HCL OneTest™ Server that you can add to your task for the job in the Azure DevOps pipelines.

You must have created the tests in the desktop clients and committed the test assets and test resources to a remote repository. The remote repository must be added to the project.

Depending on the type of tests you want to perform on your application, you must have created any or all of the following types of tests in the desktop clients for the application you are testing:

Type of test	Desk-top client
<ul style="list-style-type: none"> <li>Accelerated Functional Testing suites</li> <li>Compound tests</li> </ul>	HCL OneTest™ UI
<ul style="list-style-type: none"> <li>Test Suite</li> </ul>	HCL OneTest™ API
<ul style="list-style-type: none"> <li>Compound tests</li> <li>VU Schedule</li> <li>Rate Schedule</li> </ul>	HCL OneTest™ Performance

You can now follow the tasks listed in the task flow table to integrate HCL OneTest™ Server with Azure DevOps. See [Task flow for integrating Azure DevOps on page 487](#).

## Task flow for integrating HCL OneTest™ Server with Azure DevOps

The table shows the task flow for integrating HCL OneTest™ Server with Azure DevOps by using the *HCL OneTest Studio* extension. You must perform these tasks in sequence as listed in the following table. The table also provides you the links to the information about the tasks.

	Tasks	More information
1	Install HCL OneTest™ Server.	<a href="#">Installation of the server software on page 36</a>
2	Create tests in the desktop clients for the application you are testing.	<ul style="list-style-type: none"> <li><a href="#">HCL OneTest™ Performance documentation</a></li> <li><a href="#">HCL OneTest™ UI documentation</a></li> <li><a href="#">HCL OneTest™ API documentation</a></li> </ul>
3	Create a project on HCL OneTest™ Server and add the test assets that are created in the desktop clients to your project.	<a href="#">Test assets and a server project on page 580</a>
4	Create an organization and a project in Azure DevOps for running jobs in Azure DevOps pipelines.	<a href="#">Creating an organization</a>

	Tasks	More information
5	Access the <b>Visual Studio Marketplace</b> portal and search for the latest version of the <i>HCL OneTest Studio</i> extension.	<a href="#">Visual Studio Marketplace</a>
6	Install the latest version of the <i>HCL OneTest Studio</i> extension.	<a href="#">Installing the HCL OneTest Studio extension on page 488</a>
7	Extend the trust certificate if you have used an internal CA certificate.	Extending the trusted CA list
8	Run tests in an Azure DevOps pipeline.	<a href="#">Running tests in an Azure DevOps Pipeline on page 489</a>

Related information

[Integration with Azure DevOps on page 486](#)

## Installing the *HCL OneTest Studio* extension

You must install the latest version of the *HCL OneTest Studio* extension in your Azure DevOps organization before you can use the extension for running your application tests in an Azure DevOps pipeline.

### Before you begin

You must have access to the **Visual Studio Marketplace** portal to install the latest version of the *HCL OneTest Studio* extension.

### About this task

After you install the latest version of the *HCL OneTest Studio* extension from the **Visual Studio Marketplace** portal in your Azure DevOps organization, you can select the tests that you want to run for your application in an Azure DevOps pipeline by using the *HCL OneTest Studio* extension.

1. Log in to the **Visual Studio Marketplace** portal, if you are not already logged in.
2. Click the **Azure DevOps** tab.
3. Search for the *HCL OneTest Studio* extension.
4. Click the *HCL OneTest Studio* extension.
5. Click **Get it free**.

#### Result

The **Visual Studio Marketplace** portal for the *HCL OneTest Studio* extension is displayed.

6. Select the organization where you want to run your test from the **Select an Azure DevOps Organization** list.
7. Click **Install**.

#### Result

The installation is completed.

8. Click **Proceed to organization**.

**Result**

The **Organization** page in Azure DevOps is displayed.

9. Click **Organization settings > Extensions**.

**Result**

The *HCL OneTest Studio* extension is displayed as an installed extension.

**Results**

You have installed the *HCL OneTest Studio* extension in your Azure DevOps organization.

**What to do next**

You can run tests that are available in HCL OneTest™ Server as a job in an Azure DevOps pipeline. See [Running tests in an Azure DevOps Pipeline on page 489](#).

## Running HCL OneTest™ Server tests in an Azure DevOps Pipeline

After you install the *HCL OneTest Studio* extension in your organization, you can run tests that are available in HCL OneTest™ Server as a job in Azure DevOps pipelines.

**Before you begin**

You must have completed the following tasks:

- Added the tests that you created in the desktop clients for your application to the project on HCL OneTest™ Server.
- Installed the latest version of the *HCL OneTest Studio* extension in your organization. See [Installing the HCL OneTest Studio extension on page 488](#).
- Installed an agent in your pipeline. See [Azure Pipelines agents](#).
- Generated an offline user token from HCL OneTest™ Server. See [Generating an offline token on page 565](#).

**About this task**


After you add the *HCL OneTest Studio* extension in your Azure DevOps organization, you can use an existing pipeline or create a new one to add HCL OneTest™ Server test tasks. You can install an agent or use the one that you installed in your default agent pool. You can add HCL OneTest™ Server tests as tasks to your agent job, configure the task, and then run the task in the Azure DevOps pipeline.

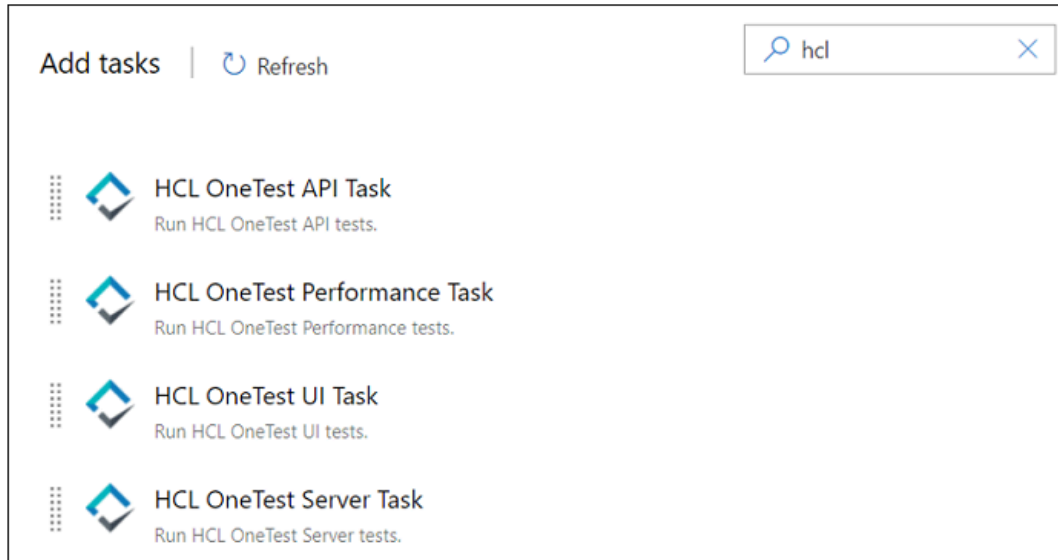
1. Open your **Organization** page in Azure DevOps and perform the following steps:
  - a. Click the project you want to use.
  - b. Initialize the repository by performing the following steps:

- i. Click **Repos** from the left pane.
- ii. Click **Initialize** from the **Initialize with a README or gitignore** section.



**Note:** Select the **Add a README** checkbox if it is not selected.

- c. Click **Pipelines** from the left pane.
  - d. Click **Create Pipeline**.
  - e. Click **Use the classic editor** to create a pipeline without YAML.
  - f. Verify the project, repository, and branch for manual and scheduled builds, and then click **Continue**.
  - g. Click **Empty job**.
2. Select **Pipeline** and complete the following steps:
- a. Change the name for the build pipeline if required.
  - b. Select the **Agent pool** for your build pipeline.  
  
You can use the agent from the default agent pool or use the one you have installed.
  - c. Select the **Agent Specification** for the agent if required.
3. Add a task to the agent job by completing the following steps:
- a. Click the **Add Task** icon  for the agent job.  
**Result**  
  
The **Add tasks** pane is displayed.
  - b. Search for the HCL tasks defined in the *HCL OneTest Studio* extension.  
**Result**  
  
The tasks that you can select are displayed.



- c. Select the **HCL OneTest™ Server Task**, and then click **Add** to add the task to the agent job.

**Result**

The selected task is added to the agent job and it is displayed with a warning that some settings require attention. You must configure the settings mentioned in [step 4 on page 491](#).

You can also remove the tasks that are not required in your job. Select the tasks in the list that you want to remove. You can then right-click the tasks, and click **Remove selected task(s)** to remove them.




4. Configure the settings by performing the following steps:

- a. Select the task version from the list if required.
- b. Follow the action for the task by referring to the following table:




**Note:** All mandatory fields are marked with an asterisk (\*) in the UI.

Field	Action
Display name	Enter a name of the task.
HCL OneTest™ Server service connection	Select the service connection from the drop-down list.  If you are selecting the HCL OneTest™ Server service connection for the first time, you must click <b>New</b> to add the following details to add an HCL OneTest™ Server service connection, and then save the connection details:

Field	Action
	<ul style="list-style-type: none"> <li>▪ <b>Server URL</b> - Enter the URL of HCL OneTest™ Server. The format of the URL is as follows: <code>https://hostname</code>.</li> <li>▪ <b>Offline Token</b> - Enter the offline token that you generated from HCL OneTest™ Server.</li> <li>▪ <b>Service connection name</b> - Enter a service connection name.</li> <li>▪ <b>Description (optional)</b> - Enter the details of the service connection if required.</li> <li>▪ Optionally, select the <b>Grant access permission to all pipelines</b> checkbox if required.</li> </ul> <p>You can save the service connection details. The service connection is available for selection from the <b>HCL OneTest™ Server service connection</b> drop-down list.</p> <p> <b>Note:</b></p> <p>You can edit or delete the service connection that you added if required. Click <b>Manage</b>, and then select the service connection from the <b>Service connections</b> list to open the service connection. You can click <b>Edit</b> to edit the service connection details. If you want to delete an existing service connection details, click the <b>Vertical ellipsis</b> icon, and then click <b>Delete</b> to delete the service connection details.</p> <p>Optionally, you can also create, edit, and delete a service connection from your Azure DevOps project dashboard. Open your project, click <b>Project settings &gt; Service connections</b>, and then perform any of the following tasks:</p> <ul style="list-style-type: none"> <li>▪ Create a new service connection.</li> <li>▪ Edit or delete an existing service connection that you already added.</li> </ul>
<p>Team Space Name</p>	<p>Enter the name of the team space that contains the project.</p> <p> <b>Note:</b> The license on the team space must be configured and you must be a member of that team space.</p>
<p>Project</p>	<p>Enter the name of the project from the available names in the team space.</p> <p> <b>Note:</b> You must be an Owner or a Tester of the project that is available in the team spaces to run the tests from the Azure DevOps pipeline.</p>
<p>Branch</p>	<p>Enter the branch where the test assets are stored.</p>
<p>Repository Link</p>	<p>Enter the repository path that is configured for the project that contains the test to run.</p>



Field	Action
File Path	Enter the path of the test in HCL OneTest™ Server that you want to run.
API Test Environment	Enter the name of the environment that is configured in the test asset for the test created in HCL OneTest™ API.  <b>Note:</b> This setting is applicable only if you are running an API suite test.

- c. Expand **Control Options** and configure the settings for your task if required.
  - d. Expand **Output Variables** and configure the settings for your task if required.
5. Select from the following options:
- a. Click **Save** to save the configured settings for the task.



**Note:** The task is not queued for a run.

You can save the task to a build pipeline and opt to run the build at a later time.

- b. Click **Save & queue** to save the configurations and queue the run in the pipeline.

#### Result

The **Run pipeline** dialog box is displayed.

6. Complete the following steps:
- a. Enter a comment for the test in the **Save comment** field.
  - b. Select the agent that you configured for the test from the **Agent pool** list.
  - c. Select the agent specification from the **Agent Specification** list if required.
  - d. Select the branch from the **Branch/tag** list.
  - e. Add the variables and demands for the task run from the **Advanced Options** pane if required.
  - f. Select the **Enable system diagnostics** checkbox for a detailed log view.
  - g. Click **Save and run**.

#### Result

The `pipeline summary` page displays the progress of the job run.

### Results

You have run the tests for the application you are testing, in the Azure DevOps pipeline.

## What to do next

You can open the job to view the task logs from the [pipeline summary page](#).

You must click the task to open the **Task** page to view the test results. The **Reports information** section on the **Task** page displays the names of the report along with its corresponding URLs. The report URLs are the HCL OneTest™ Server URLs where the reports are stored. You can access the report URLs to view the test execution information at any point of time.

You can also view the test reports and logs of the test that was run on the Azure DevOps from the **Results** page on HCL OneTest™ Server. See [Test results and reports overview on page 436](#).

## Integration with HCL® Launch

When you use HCL® Launch for automating application deployments of your application, you can run tests created for your application and available in a project HCL OneTest™ Server on from HCL® Launch by using the HCL OneTest™ Server Launch plugin.

### Overview

You can use the HCL OneTest™ Server Launch plugin to integrate HCL OneTest™ Server with HCL® Launch.

You must have created the tests in the desktop clients and committed the test assets and test resources to a remote repository. The remote repository must be added to the project.

Depending on the type of tests you want to perform on your application, you must have created any or all of the following types of tests in the desktop clients for the application you are testing:

Type of test	Desk-top client
<ul style="list-style-type: none"> <li>Accelerated Functional Testing suites</li> <li>Compound tests</li> </ul>	HCL OneTest™ UI
<ul style="list-style-type: none"> <li>Test Suite</li> </ul>	HCL OneTest™ API
<ul style="list-style-type: none"> <li>Compound tests</li> <li>VU Schedule</li> <li>Rate Schedule</li> </ul>	HCL OneTest™ Performance

You can now follow the tasks listed in the task flow table to integrate HCL OneTest™ Server with HCL® Launch. See [Task flow for integrating HCL Launch](#).

## Installing the HCL OneTest™ Server Launch plugin

You must install the latest version of the HCL OneTest™ Server Launch plugin on the HCL® Launch server to integrate HCL OneTest™ Server with HCL® Launch and run tests on the HCL® Launch server.

### Before you begin

You must have downloaded the latest version of the HCL OneTest™ Server Launch plugin from HCL License and Delivery Portal. See [HCL® License & Delivery portal](#).

1. Open the HCL® Launch dashboard.
2. Click **Settings**.
3. Click **Automation Plugins** from the **Automation** pane.
4. Click **Load Plugin**.
5. Click **Choose File** to locate and **Open** the compressed HCL OneTest™ Server Launch plugin file.



**Note:** Do not extract the HCL OneTest™ Server Launch plugin compressed file contents.

6. Click **Submit**.

### Result

The HCL OneTest™ Server Launch plugin is displayed in the **Automation Plugins** tab.

### What to do next

You can run the tests for the application that are available in your HCL OneTest™ Server project on the HCL® Launch server. See [Running tests on the HCL® Launch server on page 508](#).

## Using a custom trust store for HCL® Launch integration

You can use a custom trust store in the HCL OneTest™ Server Launch plugin file to establish a trusted and secure connection between the HCL® Launch server and HCL OneTest™ Server.

### Before you begin

You must have configured the certificate that is used by HCL OneTest™ Server as a trusted CA, and then install HCL OneTest™ Server. See [Installation of the server software on page 36](#).

### About this task

If the SSL certificate assigned to HCL OneTest™ Server is signed by an internal Certified Authority (CA), then you must download and import the CA certificate to a custom trust store. You can then use the custom trust store in the **HCL OneTest™ Server test** process step to establish a trusted and secure connection between the HCL® Launch server and HCL OneTest™ Server.



**Note:** If the internal CA certificate is already imported to the default trust store that is used by the HCL® Launch server, you need not use a custom trust store.

1. Locate the default trust store file (`cacerts` file) in your JRE directory from your computer, and then copy the file to a location of your choice on your computer.
2. Run the following command from the command prompt or terminal to import the CA certificate to your custom trust store:

```
keytool -import -trustcacerts -file <path to the downloaded CA certificate with the file extension> -alias <custom label for the certificate> -keystore <path to the trust store>
```

For example,

```
keytool -import -trustcacerts -file C:\Users\ca file.crt -alias alias1 -keystore D:\cert\cacerts
```



**Note:** The default password of the trust store is *changeit*. It remains the same for the custom trust store. If you want to change the password, you can run the following command, and then enter the new password:

```
keytool -storepasswd -keystore <path to the trust store>
```

For example, `keytool -storepasswd -keystore D:\cert\cacerts`

## Results

You have successfully imported the downloaded CA certificate to the custom trust store.

## What to do next

You can then run the tests that are available in your HCL OneTest™ Server project from the HCL® Launch server.

## Creating a component in HCL® Launch

You must create a component to include artifacts and processes. The artifacts include runnable files, images, databases, configuration instructions. Whereas the processes define the activities that components can perform.

### Before you begin

- You must be familiar with working with HCL® Launch.
- You must have been granted access to HCL® Launch.

1. Log in to HCL® Launch, if you are not already logged in.

#### Result

The HCL® Launch dashboard is displayed.

2. Click **Components**, and then click **Create Component**.
3. Enter a name for the component in the **Name** field.
4. Enter the details in the other optional fields based on your requirement, and then click **Save**.

**Result**

The component that you created is displayed.

**Results**

You have created the component in HCL® Launch.

**What to do next**

You must create a process for the component in HCL® Launch. See [Creating a process in HCL Launch on page 497](#).

---

Related information

[HCL Launch Documentation](#)

## Creating a process in HCL® Launch

You must create a process for the component to include step properties for the test that you want to run from HCL® Launch.

**Before you begin**

- You must be familiar with working with HCL® Launch.
- You must have performed the following tasks:
  - Been granted access to HCL® Launch.
  - Created a component in HCL® Launch. See [Creating a component in HCL Launch on page 496](#).

1. Log in to HCL® Launch, if you are not already logged in.

**Result**

The HCL® Launch dashboard is displayed.

2. Click **Components**.

**Result**

A list of components that are available in HCL® Launch is displayed.

3. Select the component from the list for which you want to create a process.
4. Click the **Processes** tab, and then click **Create Process**.

**Result**

The **Create Process** dialog is displayed.

5. Enter a name for the process in the **Name** field.
6. Select **Operational (No Version Needed)** from **Process Type** drop-down list.
7. Verify the **Default Working Directory** field.

The **Default Working Directory** field defines the location that the agent uses to run the process. The default value is `${p:resource/work.dir}/${p:component.name}`.

Where `${p:resource/work.dir}` is the default working directory for the agent and `${p:component.name}` is the name of the component.

8. Click **Save**.

#### **Result**

The process that you created is listed in the **Processes** tab and the **Design** tab for the process is displayed.

#### **Results**

You have created the process for the component in HCL® Launch.

#### **What to do next**

You must configure the process in HCL® Launch. See [Configuring the process on page 498](#).

---

Related information

[HCL Launch Documentation](#)

## Configuring the process

You must configure the process that you created for the component to organize the steps in the process, specify the properties of the steps, and connect them.

#### **Before you begin**

- You must be familiar with working with HCL® Launch.
- You must have performed the following tasks:
  - Been granted access to HCL® Launch.
  - Created a component in HCL® Launch. See [Creating a component in HCL Launch on page 496](#).
  - Created a process for the component in HCL® Launch. See [Creating a process in HCL Launch on page 497](#).

#### **About this task**

When you open any process to configure, the process is displayed in the process editor. The process editor lists the plugins and steps. The required **Start** and **Finish** steps represent the beginning and the end of the process and steps are automatically placed on the design area.

You must provide the values for certain fields in the properties for the selected test step to run tests from HCL® Launch.

1. Log in to HCL® Launch, if you are not already logged in.

#### **Result**

The HCL® Launch dashboard is displayed.

2. Click **Components**.

**Result**

A list of components that are available in HCL® Launch is displayed.

3. Select the component from the list in which you created the process.
4. Click the **Processes** tab.

**Result**

A list of processes that are available for the component is displayed.

5. Select the process from the list that you want to configure.

**Result**

The **Design** tab for the process is displayed.

6. Click **HCL OneTest Studio**, and then **HCL OneTest Server** from the left menu.
7. Drag the **Run HCL OneTest Server test** step, and then drop it into the design area.



**Note:** The selected test must be placed between **Start** and **Finish** steps.

8. Specify the properties for the selected test by performing the following steps:


- a. Click the **Edit** icon.

**Result**

The **Edit Properties for Run an HCL OneTest Server test** dialog is displayed.


- b. Specify the properties for the selected test step by referring to the following table:

The following table lists the required fields that you must provide to run the test from HCL® Launch:


Fields	Action
<b>Name</b>	Enter the name for the test step.
<b>HCL OneTest Server URL</b>	Enter the URL of HCL OneTest™ Server.
<b>Offline Token</b>	Enter the offline token that you generated from HCL OneTest™ Server.
<b>Team Space Name</b>	Enter the name of the team space that contains the project.   <b>Note:</b> The license for the team space must be configured and you must be a member of that team space.
<b>Project Name</b>	Enter the name of the project that are available in the team space.


Fields	Action
<b>Repository Link</b>	Enter the URL of the Git repository that you added to your project in HCL OneTest™ Server.
<b>File Path</b>	Enter the path of the test assets that you want to run.  You can find the path of the test assets from the <b>Execution</b> page in HCL OneTest™ Server

The following table lists the optional fields that you can provide to run the test from HCL® Launch:

Fields	Description
<b>Branch Name</b>	Use this field to enter the name of the branch in the Git repository where you stored test assets.
<b>Custom Trust Store Password</b>	Use this field to enter the trust store password if you have modified the password while creating the custom trust store.   <b>Note:</b> If you have not modified the trust store password, you can retain the <b>Custom Trust Store Password</b> field blank.
<b>Custom Trust Store Path</b>	Use this field to enter the file path of your trust store followed by the file name if HCL OneTest™ Server uses an internal CA certificate and you have imported the certificate to a custom trust store.
<b>Datasets</b>	Use this field to enter the path to the dataset if you want to replace the values of the dataset during a test run.  You must ensure that both original and new datasets are in the same workspace and have the same column names. When you enter a value for the <b>Datasets</b> field, you must also include the path to the dataset. The path must be in the following format:  <code>/project_name/ds_path/original_ds.csv:/project_name/ds_path/new_ds.csv</code>



Fields	Description
	 <b>Note:</b> You can override multiple datasets that are saved in a different project by adding multiple paths to the dataset separated by a semicolon.
<b>Environment</b>	Use this field to enter the environment details for the following types of test assets: <ul style="list-style-type: none"> <li>▪ APISUITE</li> <li>▪ APITEST</li> <li>▪ APISTUB</li> </ul>
<b>Labels</b>	Use this field to add labels to test results when the test run is complete. You can add multiple labels to a test result separated by a comma. For example, <i>label1, label2</i> After the test run is complete, then the values provided in the <b>Labels</b> field are displayed on the <b>Results</b> page of HCL OneTest™ Server.
<b>Secrets Collection Name</b>	Use this field to enter the name of the Secret if you created secrets collections for your project and to use Secrets at test run time. This field is mandatory only if you want to run the following types of test assets: <ul style="list-style-type: none"> <li>▪ APISUITE</li> <li>▪ APITEST</li> <li>▪ APISTUB</li> </ul>
<b>Show Hidden Properties</b>	Select this checkbox if you want to use an HTTP Proxy to connect to HCL OneTest™ Server. After you select the <b>Show Hidden Properties</b> checkbox, you can configure the HTTP Proxy by using the following fields: <ul style="list-style-type: none"> <li>▪ <b>HTTP Proxy Host:</b> Use this field to enter the hostname of the HTTP proxy that you want to connect to HCL OneTest™ Server.</li> <li>▪ <b>HTTP Proxy Port:</b> Use this field to enter the port number where the HTTP proxy is used to listen to both HTTP and HTTPS traffic.</li> </ul>

Fields	Description
	<ul style="list-style-type: none"> <li>▪ <b>HTTP Proxy User name:</b> Use this field to provide the username for the HTTP proxy.</li> <li>▪ <b>HTTP Proxy Password:</b> Use this field to enter a valid password for the user that you specified in the <b>HTTP Proxy User name</b> field.</li> </ul> <p> <b>Note:</b> You can use the <b>Show Hidden Properties</b> field only when you want to run the following types of test assets:</p> <ul style="list-style-type: none"> <li>▪ APISUITE</li> <li>▪ APITEST</li> <li>▪ APISTUB</li> </ul>
<b>Start Date</b>	<p>Use this field to enter the date and time in the following format for running tests at the scheduled date and time:</p> <p><code>yyyy/MM/dd/HH:mm</code></p> <p>When you enter a value in the <b>Start date</b> field, the status of the test displays as <code>Scheduled</code> on the <b>Overview</b> and <b>Progress</b> pages of HCL OneTest™ Server.</p>
<b>Variables</b>	<p>Use this field to enter the name of the variable and its value if your test requires variables during the test run time.</p> <p>You must enter the variables in the following format:</p> <p><code>name_of_the_variable=value_of_the_varibale</code></p> <p>You can add multiple variables to the test run separated by a semicolon.</p> <p>For example, <code>varname1=value1;varname2=value2</code></p>



**Note:** In addition to optional fields, you can also use the following fields from HCL® Launch to configure your test run:

- **Working Directory**
- **Precondition**
- **Post Processing Script**
- **Use Impersonation**
- **Auth Token Restriction**



You can accept the default values or change the values based on your requirements. For more information about these fields, see the related links.

c. Click **OK** to save the properties for the test.

9. Click **Save** in the design area.

### Results

You have configured the process for the component in HCL® Launch.

### What to do next

You must create a resource in HCL® Launch. See [Creating a resource in HCL Launch on page 503](#).

---

#### Related information

[HCL Launch Documentation](#)

[Process step preconditions](#)

[Post-processing scripts](#)

[User impersonation for process steps](#)

[Restricting authentication tokens](#)

## Creating a resource in HCL® Launch

You must create a resource to associate agents with components that you created in HCL® Launch.

### Before you begin

- You must be familiar with working with HCL® Launch.
- You must have been granted access to HCL® Launch.

1. Log in to HCL® Launch, if you are not already logged in.

#### Result

The HCL® Launch dashboard is displayed.

2. Click **Resources**, and then click **Create Top-Level Group**.

#### Result

The **Create Resource** dialog is displayed.

3. Enter a name for the resource in the **Name** field.

4. Click **Save**.

### Results

You have created the resource in HCL® Launch.

### What to do next

You must configure the resource. See [Configuring the resource on page 504](#).

---

Related information

[HCL Launch Documentation](#)

## Configuring the resource

You must configure the resource to add an agent and associate the agent with the component.

### Before you begin

- You must be familiar with working with HCL® Launch.
- You must have performed the following tasks:
  - Been granted access to HCL® Launch.
  - Created a component in HCL® Launch. See [Creating a component in HCL Launch on page 496](#).
  - Created a resource in HCL® Launch. See [Creating a resource in HCL Launch on page 503](#).

1. Log in to HCL® Launch, if you are not already logged in.

#### Result

The HCL® Launch dashboard is displayed.

2. Click **Resources**.

#### Result

A list of resources that are available in HCL® Launch is displayed.

3. Perform the following steps to add an agent to the resource:

- a. Click the resource from the list for which you want to add an agent.
- b. Click the **Actions** icon from the last column, and then click **Add Agent**.
- c. Select the agent from the drop-down list.



**Note:** The **Name** field is auto populated with the name of the agent.

- d. Click **Save**.

#### Result

The selected agent is added to the resource and you can view the status of the agent in the **Status** column.

4. Perform the following steps to add a component to the agent:

- a. Click the agent from the list for which you want to add a component.
- b. Click the **Actions** icon from the last column, and then click **Add Component**.
- c. Select the component from the drop-down list.



**Note:** The **Name** field is auto populated with the name of the component.

- d. Click **Save**.

**Result**

The selected component is added to the agent.

**Results**

You have configured the resource in HCL® Launch.

**What to do next**

You must create an application. See [Creating an application in HCL Launch on page 505](#).

Related information

[HCL Launch Documentation](#)

## Creating an application in HCL® Launch

You must create an application to fetch all the components together that you want to deploy.

**Before you begin**

- You must be familiar with working with HCL® Launch.
  - You must have been granted access to HCL® Launch.
1. Log in to HCL® Launch, if you are not already logged in.

**Result**

The HCL® Launch dashboard is displayed.

2. Click **Applications**.
3. Click **Create Applications**, and then **New Applications**.
4. Enter a name for the application in the **Name** field.

**Result**

The **Environments** page for the application that you created is displayed.

**Results**

You have created the application in HCL® Launch.

**What to do next**

You must configure the application. See [Configuring the application on page 506](#).

---

Related information

[HCL Launch Documentation](#)

## Configuring the application

You must configure the application to associate resources with environments and define processes to run test assets.

### Before you begin

- You must be familiar with working with HCL® Launch.
- You must have performed the following tasks:
  - Created a process for the component in HCL® Launch. See [Creating a process in HCL Launch on page 497](#).
  - Created a component in HCL® Launch. See [Creating a component in HCL Launch on page 496](#).
  - Created a process for the component in HCL® Launch. See [Creating a process in HCL Launch on page 497](#).
  - Configure the process for the component in HCL® Launch. See [Configuring the process on page 498](#).
  - Created a resource in HCL® Launch. See [Creating a resource in HCL Launch on page 503](#).
  - Configured the resource in HCL® Launch. See [Configuring the resource on page 504](#).
  - Created an application in HCL® Launch. See [Creating an application in HCL Launch on page 505](#).

1. Log in to HCL® Launch, if you are not already logged in.

#### Result

The HCL® Launch dashboard is displayed.

2. Click **Applications**.

#### Result

A list of applications that are available in HCL® Launch is displayed.

3. Click the application that you want to configure from the **Name** column.

#### Result

The **Environments** page for the selected application is displayed.

4. Perform the following steps to create an environment for the application that you selected:

- a. Click **Create Environment**.
  - b. Enter a name for the environment in the **Name** field.
  - c. Click **Save**.
5. Perform the following steps to configure resources to the environment:
- a. Click the environment that you created.
  - b. Click **Add Base Resources**.
 

**Result**

A list of resources that are available in HCL® Launch is displayed.
  - c. Select the checkbox to add resources to the environment.
  - d. Click **Save**.
 

**Result**

You can view the corresponding agent and the component that you added for the resource by using the **Expand** icon.
6. Perform the following steps to add the component to the application:
- a. Click **Applications**, and then select your application from the list.
  - b. Click the **Components** tab, and then **Add Components**.
  - c. Select the checkbox from the drop-down list to add components to the application.
  - d. Click **Save**.
7. Perform the following steps to create a process for the application:
- a. Click the **Processes** tab, and then **Create Process**.
  - b. Enter a name for the process in the **Name** field.
  - c. Click **Save**.
 

**Result**

The **Design** tab for the process that you created is displayed.
8. Drag the component process listed under the **Component Process Steps** option from the left navigation pane and drop it into the design area.
9. Select the component process from the drop-down list in the **Operational (No Version Needed) Process** field.
10. Click **Save**.
11. Click the **Edit** icon, and then change the name of the properties.
12. Click **OK**, and then click **Save**.

### Results

You have configured the application to run test assets from HCL® Launch.

### What to do next

You can run test assets from HCL® Launch. See [Running HCL OneTest Server tests on the HCL Launch server on page 508](#).

---

Related information

[HCL Launch Documentation](#)

## Running HCL OneTest™ Server tests on the HCL® Launch server

After you install the HCL OneTest™ Server Launch plugin on the HCL® Launch server, you can create a `process request` that contains the test for your application, and then run the test on the HCL® Launch server.

### Before you begin

You must have completed the following tasks:

- Added the tests that you created in the desktop clients for your application to the project on HCL OneTest™ Server.
- Installed the latest version of the HCL OneTest™ Server Launch plugin. See [Installing the HCL OneTest™ Server Launch plugin on page 495](#).
- Generated an offline user token from HCL OneTest™ Server. See [Generating an offline token on page 565](#).

### About this task

After you have installed the HCL OneTest™ Server Launch plugin on the HCL® Launch server, you can either use an existing component in your project or create a component. You can create a component process and select the **HCL OneTest™ Server test** step to edit the *step properties* for the test you want to run. After selecting the agent, you can create an application. You can then create an application process for the application, and then submit the application process for a run.

1. Log in to the HCL® Launch server, if you are not already logged in.
2. Click **Components** from the HCL® Launch dashboard, and then click **Create Component** to create a component.



**Note:** You can either use an existing component or create a component.

3. Create a component process in the component by performing the following steps:
  - a. Open the component that you created.
  - b. Click the **Processes** tab from the component dashboard, and then click **Create Process**.

#### Result

The **Create Process** dialog box is displayed.

- c. Enter the required values to create a component process and click **Save**.





**Note:** All mandatory fields are marked with an asterisk (\*) in the UI.

- i. Enter the process name in the **Name** field.
- ii. Select **Operational (No Version Needed)** from the **Process Type** list.



**Note:** The **Default Working Directory** field displays the folder path where the agent can download the artifacts and create temporary files.

The process that you created is listed in the **Processes** list and the **Design** tab for the process is displayed.



**Note:** The process opens in the process editor. The process editor lists the plugins and steps. The required **Start** and **Finish** steps represent the beginning and the end of the process and are automatically placed on the design area.

4. Select the process step you want to run by completing the following steps:
  - a. Search for the **HCL OneTest™ Server test** process step from the left design pane.
  - b. Select the **Run HCL OneTest™ Server test** process step and drag the test into the design area.

**Result**

The selected test is placed in between the **Start** and **Finish** steps.

5. Specify the properties for the selected test by performing the following steps:

- a. Click the **Edit** icon .

**Result**





The **Edit Properties for Run HCL OneTest™ Server test** dialog box of the selected test is displayed.

- b. Specify the properties for the selected test step by following the action in the table that follows.



**Note:** All mandatory fields are marked with an asterisk (\*) in the UI.

Field	Action required for a HCL OneTest™ Server test
Name	Enter the name of the step.
HCL OneTest™ Server URL	Enter the URL for HCL OneTest™ Server.
Offline Token	Enter the offline token that you generated from HCL OneTest™ Server.
Team Space Name	Enter the name of the team space that contains the project.

Field	Action required for a HCL OneTest™ Server test
	 <b>Note:</b> The license on the team space must be configured and you must be a member of that team space.
Project Name	<p>Enter the name of the project from the available names in the team space.</p>  <b>Note:</b> You must be an Owner or a Tester of the project that is available in the team spaces to run the tests from the HCL® Launch server.
Branch Name	Enter the name of the branch where the test assets are stored.
Repository Link	Enter the repository path for the test to run.
File Path	Enter the file path of the HCL OneTest™ Server test that you want to run.
Environment	<p>Enter the HCL OneTest™ API environment details for the test.</p>  <b>Note:</b> This field is applicable only for an API test.
Field	Optional action for a HCL OneTest™ Server test
Custom Trust Store Path	Enter the file path of your trust store followed by the file name in the <b>Custom Trust Store Path</b> field if the HCL OneTest™ Server uses an internal CA certificate and you have imported the certificate to a custom trust store. <sup>1</sup>
Custom Trust Store Password	<p>Enter the trust store password if you have modified the password while creating the custom trust store. <sup>1</sup></p>  <b>Note:</b> If you have not modified the trust store password, you can keep the <b>Custom Trust Store Password</b> field blank.
Working Directory	Specify an alternative path to the working directory for this step. <sup>1</sup>
Precondition	Specify any conditions that are to be completed before the test runs. You can edit the script by clicking the script displayed. <sup>1</sup>
Post Processing Script	Specify if you want to run any scripts after the completion of the test run. You can click <b>New</b> to add new scripts. <sup>1</sup>

1. You need not set this property for a step when you are running an HCL OneTest™ Server test.

Field	Action required for a HCL OneTest™ Server test
Use Impersonation checkbox	Select this checkbox to run the test as a different user. <sup>1</sup>
Auth Token Restriction	Set the authentication token actions by applying token restrictions.  The <b>System Default</b> is selected by default. You can add a new token restriction or edit the one already added. <sup>1</sup>

c. Click **OK** to save the properties for the test.


6. Click **Save** in the design area.

7. Click the **Resources** tab from the HCL® Launch dashboard and create a resource by clicking **Create Top-Level Group**.


#### Result

The created resource is displayed on the **Resource Tree** tab page.

8. Select the agent that runs the test by completing the following steps:

 **Important:** You must have already installed the agent on the HCL® Launch server that you want to use.

a. Select the resource displayed on the **Resource Tree** tab page.


b. Click the **Horizontal ellipsis** icon  for the selected resource.

c. Click **Add Agent**.

d. Select the agent to add to the resource, and then click **Save**.

#### Result

The selected agent is added to the resource in the **Resource Tree** pane and the status of the agent can also be viewed.

 **Important:** The agent must be **Online** for the test to run.

9. Add the component to the agent by performing the following steps:

a. Click the **Horizontal ellipsis** icon  for the agent.

b. Click **Add Component** on the list.

c. Select the component to add to the resource, and then click **Save**.

#### Result

The selected component is added to the agent for the resource in the **Resource Tree** pane.

10. Create an application by completing the following steps:

- a. Click the **Applications** tab from the HCL® Launch dashboard.
- b. Click **Create Application**.
- c. Complete the details in the **Create Application** dialog box, and then click **Save**.

**Result**

The **Environments** tab page is displayed for the created application.

- d. Click **Create Environment** to create an environment for the application that you created.
- e. Complete the details in the **Create Environment** dialog box, and then click **Save**.

**Result**

The environment that you created is displayed.

- f. Click the environment to open, and then click **Add Base Resources**.
- g. Select the resource from the list in the **Add Resource to Environment** dialog box, and then click **Save** to add the resource to the environment.

**Result**

The resource added to the environment is displayed.



**Note:** When you add a resource to an environment, the corresponding agent and the component are displayed for the resource.

- h. Click the application from the `breadcrumbs`.

**Result**

The **Environments** tab page is displayed for your application.

- i. Add the component to the application by performing the following steps:
  - i. Click the **Components** tab.
  - ii. Click **Add Component**.
  - iii. Select the component from the list in the **Add a Component** dialog box, and then click **Save**.

**Result**

The selected component is displayed on the **Components** tab page.

- j. Create a process for the application by performing the following steps:
  - i. Click the **Processes** tab.
  - ii. Click **Create Process**.
  - iii. Complete the details in the **Create an Application Process** dialog box, and then click **Save**.

**Result**

The **Design** tab page for the application process that you created is displayed.

- k. Select the component process from the left pane and drag it into the design area.



**Note:** You can click the **Edit** icon  to add the properties, if required.

- l. Click **Save** in the design area.

11. Select the application process to run the test by completing the following steps:

- a. Click **Applications** from the HCL® Launch dashboard.
- b. Click the application that you configured for a test run.
- c. Click **Request Process**.

#### Result

The **Create Deployment** page is displayed.

- d. Specify the process request by performing the following steps:
  - i. Select the application from the **Application** list.
  - ii. Select the environment from the **Environment** list.
  - iii. Select the application process from the **Process** list.
  - iv. Optionally, enter a description in the **Description** field.
  - v. Click **Next**.
  - vi. Select the component versions as required, and then click **Next**.
  - vii. Enable **Run Now** to run the `application process request`.



**Note:** You can also schedule the `application process request` at a later point of time. To do this, you can disable **Run Now**, and then specify the date, time, and the recurrence pattern to run the `application process request` at a stipulated time.

- viii. Click **Next**.
- ix. Verify the process request details, and then click **Submit Deployment**.

#### Result

The HCL® Launch dashboard shows the progress of the application process request.


#### Results

You have used the HCL OneTest™ Server Launch plugin to integrate HCL OneTest™ Server with HCL® Launch and run the test from your project on the HCL® Launch server.

After the HCL® Launch process request runs successfully, you can view the status of the completed process request displayed as follows:

- **Success:** When the test run is successful
- **Failed:** When the test run is failed

### What to do next

- You can view the details of the test run as a process from the HCL® Launch dashboard by performing the following action:
  - Expand the *step*. You can then expand the application process. You can then hover over the process, and then click the **Output Log** icon . The output log is displayed. You can verify the log details.
- You can also view the test reports and logs of the test that was run on the HCL® Launch server from the **Results** page on HCL OneTest™ Server. See [Test results and reports overview on page 436](#).

## Integration with Jenkins

When you integrate HCL OneTest™ Server with Jenkins, you can run tests created for your application and available in a project on HCL OneTest™ Server from a Jenkins server by using the HCL OneTest™ Server Jenkins plugin.

### Prerequisites

You must have installed Jenkins on your computer. For more information about installing Jenkins, refer to [Installing Jenkins](#).

You can then run the following command to start the Jenkins server to support UTF-8 character sets:

```
java -Dfile.encoding=UTF8 -jar jenkins.war
```

### Overview

You can use the HCL OneTest™ Server Jenkins plugin to integrate HCL OneTest™ Server with Jenkins.

You must have created the tests in the desktop clients and committed the test assets and test resources to a remote repository. The remote repository must be added to the project.

Depending on the type of tests you want to perform on your application, you must have created any or all of the following types of tests in the desktop clients for the application you are testing:

Type of test	Desk-top client
<ul style="list-style-type: none"> <li>• Accelerated Functional Testing suites</li> <li>• Compound tests</li> </ul>	HCL OneTest™ UI

Type of test	Desk-top client
<ul style="list-style-type: none"> <li>• Test Suite</li> </ul>	HCL OneTest™ API
<ul style="list-style-type: none"> <li>• Compound tests</li> <li>• VU Schedule</li> <li>• Rate Schedule</li> </ul>	HCL OneTest™ Performance

You can now follow the tasks listed in the task flow table to integrate HCL OneTest™ Server with Jenkins. See Task flow for integrating Jenkins.

## Task flows for running test assets from Jenkins

You can perform certain tasks to run test assets from the Jenkins **Freestyle** project.

The following table lists the task flows for running test assets from the Jenkins **Freestyle** project:

Tasks	More information
Install the HCL OneTest™ Server Jenkins plugin.	<a href="#">Installing the HCL OneTest Server Jenkins Plugin on page 515</a>
Import the CA certificate to your custom trust store.	<a href="#">Using a custom trust store on page 516</a>
Configure the <b>Freestyle</b> project.	<a href="#">Configuring the Freestyle project on page 517</a>
Run HCL OneTest™ Server test assets on Jenkins.	<a href="#">Running HCL OneTest Server tests on the Jenkins server on page 521</a>

## Installing the HCL OneTest™ Server Jenkins Plugin

You must install the latest version of the HCL OneTest™ Server plugin on the Jenkins server before you can use the plugin for running tests that are available in your HCL OneTest™ Server project on the Jenkins server.

### Before you begin

You must have downloaded the HCL OneTest™ Server Jenkins plugin 5.0 from the [HCL License and Delivery](#) portal.

1. Open the Jenkins dashboard.
2. Click **Manage Jenkins > Manage Plugins**.
3. Click **Advanced**.
4. Click **Choose File** to locate and **Open** the HCL OneTest™ Server Jenkins plugin file from the **Upload Plugin** section.

### 5. Click **Upload**.

#### **Result**

The HCL OneTest™ Server Jenkins plugin is displayed in the **Installed** tab.

#### **What to do next**

You can run the tests that are available in HCL OneTest™ Server on the Jenkins server. See [Running HCL OneTest™ Server tests on the Jenkins server on page 521](#).

## Using a custom trust store

You can use a custom trust store in the Jenkins build step of a HCL OneTest™ Server Jenkins plugin to establish a trusted and secure connection between the Jenkins server and HCL OneTest™ Server.

#### **Before you begin**

You must have configured the certificate that is used by HCL OneTest™ Server as a trusted CA, and then install HCL OneTest™ Server. See [Installation of the server software on page 36](#).

#### **About this task**

If the SSL certificate assigned to HCL OneTest™ Server is signed by an internal Certified Authority (CA), then you must download and import the CA certificate to a custom trust store. You can then use the custom trust store in the Jenkins plugin build step to establish a trusted and secure connection between the Jenkins server and HCL OneTest™ Server.



**Note:** If the internal CA certificate is already imported to the default trust store that is used by the Jenkins server, you need not use a custom trust store.



#### **Restriction:**

When you use Red Hat Enterprise Linux (RHEL) operating systems for Jenkins, you must run the Jenkins service with a user who has access to the custom trust store path to utilize the custom trust store feature. To change the Jenkins user, you must open the `/etc/sysconfig/jenkins` file and set the `JENKINS_USER` to the user who has access to the custom trust store path.

You can then run the following commands to set `JENKINS_USER` to another user who has access to the path of the custom trust store:

```
$JENKINS_USER= <username>
```

For example, `$JENKINS_USER= <user1>`



**Note:** You must ensure that the user account is available in the `/etc/passwd` file.

You can then run the following commands to change the ownership of the Jenkins folder:





```
chown -R username:username /var/lib/jenkins
```

```
chown -R username:username /var/cache/jenkins
```

```
chown -R username:username /var/log/jenkins
```

For example,

```
chown -R user1:user1 /var/lib/jenkins
```

```
chown -R user1:user1 /var/cache/jenkins
```

```
chown -R user1:user1 /var/log/jenkins
```

After the change of ownership is complete, run the following command to restart the Jenkins server:

```
/etc/init.d/jenkins restart
```

1. Locate the default trust store file (`cacerts` file) in your JRE directory from your computer, and then copy the file to a location of your choice on your computer.
2. Run the following command from the command prompt or terminal to import the CA certificate to your custom trust store:

```
keytool -import -trustcacerts -file <path to the downloaded CA certificate with the file extension> -alias <custom label for the certificate> -keystore <path to the trust store>
```

For example,

```
keytool -import -trustcacerts -file C:\Users\ca file.crt -alias alias1 -keystore D:\cert\cacerts
```



**Note:** The default password of the trust store is *changeit*. It remains the same for the custom trust store. If you want to change the password, you can run the following command, and then enter the new password:

```
keytool -storepasswd -keystore <path to the trust store>
```

For example, `keytool -storepasswd -keystore D:\cert\cacerts`

## Results

You have successfully imported the downloaded CA certificate to the custom trust store.

## What to do next

You can add the HCL OneTest™ Server tests to the Jenkins build step, and then run the tests from the Jenkins server.

## Configuring the Freestyle project


You must configure a **Freestyle** project to add a build step, and then run tests that are available in your HCL OneTest™ Server project from Jenkins.




## Before you begin



You must have completed the following tasks:


- Installed the HCL OneTest™ Server Jenkins plugin on the Jenkins server. See [Installing the HCL OneTest Server Jenkins Plugin on page 515](#).
- Created a Jenkins **Freestyle** project.
- Generated an offline user token from HCL OneTest™ Server. See [Managing access to HCL OneTest Server on page 565](#).

1. Click the **Build** tab, and then click **Add build step**.
2. Select the **Run HCL OneTest Server test** option from the drop-down list.
3. Provide the details about the test run for the fields in the following table:

Field	Description
Name	Enter a name for the Jenkins build step.
Use Custom Trust Store	<p>Select the <b>Use Custom Trust Store</b> checkbox if you have used an internal CA certificate and have imported the certificate to the custom trust store. You can then enter the file path of your trust store followed by the file name in the <b>Custom Trust Store Path</b> field.</p> <p>Select the <b>Use Custom Password for the Trust Store</b> checkbox if you have modified the trust store password, and then enter the new password in the <b>Custom Trust Store Password</b> field.</p> <p> <b>Note:</b> If you have not modified the trust store password, you must clear the <b>Use Custom Password for the Trust Store</b> checkbox.</p>
Server URL	<p>Enter the URL of HCL OneTest™ Server.</p> <p>The format of the URL is as follows: <i>https://hostname</i>.</p>
Offline Token	Enter the offline user token that you generated from HCL OneTest™ Server.
Team Space	<p>Select the name of the team space from the drop-down list.</p> <p>The <b>Team Space</b> drop-down list displays the names of the team spaces if you are an <i>Owner</i> or a <i>member</i> of the team space.</p>
Project	Select a project from the drop-down list.

Field	Description
	<p>The <b>Project</b> drop-down list displays the projects that are available in the corresponding team space of HCL OneTest™ Server.</p> <p> <b>Note:</b> The <b>Project</b> drop-down list displays the projects where you are an <i>Owner</i> or a <i>member</i> of HCL OneTest™ Server. You must be an <i>Owner</i> or a <i>Tester</i> of the project that is available in the team spaces to run the tests from the Jenkins server.</p>
Branch	<p>Select the branch from the drop-down list.</p> <p>The <b>Branch</b> drop-down list displays the branches available in the corresponding project of HCL OneTest™ Server.</p> <p> <b>Note:</b> After you select the branch from the <b>Branch</b> drop-down list, if you want to change the URL, offline user token, or the project in the <b>Build</b>, then you must close the build step. Then you must click <b>Add build step</b>, select <b>Run HCL OneTest™ Server test</b>, and then enter the details.</p>
Asset Type	<p>Select the test asset type that you want to run from the <b>Asset Type</b> drop-down list.</p> <p>The available options for the <b>Asset Type</b> field are as follows:</p> <ul style="list-style-type: none"> <li>◦ AFT Suite</li> <li>◦ API Suite</li> <li>◦ Compound Test</li> <li>◦ Rate Schedule</li> <li>◦ VU Schedule</li> </ul> <p> <b>Notes:</b></p> <ul style="list-style-type: none"> <li>◦ You cannot run the following test assets from the Jenkins server: <ul style="list-style-type: none"> <li>▪ API test</li> <li>▪ Functional test</li> <li>▪ HCL AppScan CodeSweep</li> <li>▪ JMeter Test</li> <li>▪ JUnit Test</li> </ul> </li> </ul>

Field	Description
	<ul style="list-style-type: none"> <li>▪ Performance test</li> <li>▪ Postman resources</li> </ul> <ul style="list-style-type: none"> <li>◦ The <b>Test Environment</b> field is mandatory if you select <b>APISUITE</b> as an asset type to run an API suite.</li> </ul>
Test	<p>Select the required test from the drop-down list.</p> <p>The <b>Tests</b> drop-down list displays the available test assets from the corresponding branch in the following format:</p> <pre>Project_name [Path: the path of the test assets] [Repo: URL of the Git repository that the test belongs to]</pre> <p> <b>Note:</b> After you select the test details from the <b>Tests</b> drop-down list if you want to change the URL, offline user token, or the project in the <b>Build step</b>, then you must close the build step. You must then click <b>Add build step</b>, select <b>Run HCL OneTest™ Server test</b>, and then enter the details.</p>
Test Environment	<p>This field is mandatory only if you select <b>Asset Type</b> as <b>APISUITE</b>.</p> <p>The <b>Test Environment</b> list displays the available test environments from HCL OneTest™ Server.</p> <p> <b>Note:</b> The following message is displayed when you select any other asset type apart from <b>APISUITE</b> in the <b>Asset Type</b> field:</p> <pre>You can select Test Environment only if you are running tests of type APISUITE.</pre>

 **Note:** You can run multiple tests sequentially in the same job by adding multiple build steps and providing details for the tests that you want to run.

4. Click **Save** to save the build step.

## Results

You have configured the **Freestyle** project by adding the build step.

## What to do next

You can run test assets from the Jenkins server. See [Running HCL OneTest Server tests on the Jenkins server on page 521](#).

## Running HCL OneTest™ Server tests on the Jenkins server

After you install the HCL OneTest™ Server Jenkins plugin on the Jenkins server, you can run tests that are available in your HCL OneTest™ Server project on the Jenkins server.

### Before you begin

You must have completed the following tasks:


- Added the tests that you created in the desktop clients for your application to the project on HCL OneTest™ Server.
- Installed the latest version of the HCL OneTest™ Server Jenkins plugin on the Jenkins server. See [Installing the HCL OneTest™ Server Jenkins Plugin on page 515](#).
- Generated an offline user token from HCL OneTest™ Server. See [Generating an offline token on page 565](#).
- Created a Jenkins free-style software project. See [Building a software project in Jenkins](#).




1. Login to the Jenkins server, if you are not already logged in.
2. Open your Jenkins free-style software project.
3. Click **Configure**.
4. Click the **Build** tab.
5. Click **Add build step**, and then click **Run HCL OneTest™ Server test**.



### Result

The **Run HCL OneTest™ Server test** pane is displayed.

- a. Provide details about the test run by referring to the following table.

Field	Description
Name	Enter a name for the Jenkins build step.
Use Custom Trust Store	<p>Select the <b>Use Custom Trust Store</b> checkbox if you have used an internal CA certificate and have imported the certificate to the custom trust store.</p> <p>You can then enter the file path of your trust store followed by the file name in the <b>Custom Trust Store Path</b> field.</p> <p>You can select the <b>Use Custom Password for the Trust Store</b> checkbox if you have modified the trust store password. Enter the new password.</p> <p> <b>Note:</b> If you have not modified the trust store password, you can keep the <b>Use Custom Password for the Trust Store</b> checkbox unselected.</p>
Server URL	Enter the URL of HCL OneTest™ Server. The format of the URL is as follows: <i>https://host-name</i> .

Field	Description
Offline Token	Enter the offline user token that you generated from HCL OneTest™ Server.
Team Space	Select the name of the team space from the <b>Team Space</b> drop-down list. The <b>Team Space</b> drop-down list displays the names of the team spaces if you are an Owner or a member of the team space.
Project	<p>Select a project from the <b>Project</b> drop-down list. The <b>Project</b> drop-down list displays the projects that are available in the corresponding team space of HCL OneTest™ Server.</p> <p> <b>Note:</b> The <b>Project</b> drop-down list displays the projects where you are an Owner or a member of HCL OneTest™ Server. You must be an Owner or a Tester of the project that is available in the team spaces to run the tests from the Jenkins server.</p>
Branch	<p>Select the branch from the <b>Branch</b> drop-down list. The <b>Branch</b> drop-down list displays the branches available in the corresponding project of HCL OneTest™ Server.</p> <p> <b>Note:</b> After you select the branch from the <b>Branch</b> drop-down list, if you want to change the URL, offline user token, or the project in the <b>Build</b>, then you must close the build step. Then you must click <b>Add build step</b>, select <b>Run HCL OneTest™ Server test</b>, and then enter the details.</p>
Asset Type	<p>Select the test asset type of the test that you want to run from the <b>Asset Type</b> drop-down list. The available asset types are as follows:</p> <ul style="list-style-type: none"> <li>▪ AFTSUITE</li> <li>▪ COMPOUND</li> <li>▪ VUSCHEDULE</li> <li>▪ RATESCHEDULE</li> <li>▪ APISUITE</li> </ul> <p> <b>Note:</b> The test environment is mandatory if you select <b>APISUITE</b> as an asset type to run an API suite.</p>
Test	Select the required test from the <b>Tests</b> drop-down list. The <b>Tests</b> drop-down list displays the available test assets from the corresponding branch in the selected project, test asset path, and the repository (that the test belongs to) from HCL OneTest™ Server based on the type of the test asset you selected from the <b>Asset Type</b> drop-down list.

Field	Description
	 <b>Note:</b> After you select the test details from the <b>Tests</b> drop-down list, if you want to change the URL, offline user token, or the project in the <b>Build</b> , then you must close the build step. You must then click <b>Add build step</b> , select <b>Run HCL OneTest™ Server test</b> , and then enter the details.
Test Environment	<p>This field is mandatory only if you are running an API suite test. Based on the asset type as <b>APISUITE</b> and test from the <b>Tests</b> list that you select, the <b>Test Environment</b> list displays the available test environments from HCL OneTest™ Server.</p>  <b>Note:</b> The following message is displayed when you select any other asset type apart from <b>APISUITE</b> in the <b>Asset Type</b> field: <code>You can select Test Environment only if you are running tests of type APISUITE.</code>

- b. Click **Save** to save the build step.
  - c. Optionally, you can run multiple tests sequentially in the same job by adding multiple build steps and provide details for the tests that you want to run.
6. Click **Build Now** from the left pane to run the test on the Jenkins server.

## Results

The Jenkins dashboard shows the progress of the test run.

## What to do next

After the Jenkins build completes, you can view the test results. You can click the build number from the **Build History** pane on the Jenkins dashboard. You can then click **Console Output** to view a detailed log of the build from the console output.

The `Test Result` displays the status of the completed test that you ran.

 **Note:** If you add multiple build steps to run multiple tests, multiple `Test Result` instances are displayed.

The **Reports information** section displays the names of the report along with its corresponding URLs. The report URLs are the HCL OneTest™ Server URLs where the reports are stored. You can access the report URLs to view the test execution information at any point of time.

You can also view the test reports and logs of the test that was run on the Jenkins server from the **Results** page on HCL OneTest™ Server. See [Test results and reports overview on page 436](#).

## Integration with UrbanCode Deploy

When you use UrbanCode™ Deploy (UCD) for automating application deployments of your application, you can run tests created for your application and available in a project on HCL OneTest™ Server from UCD by using the HCL OneTest™ Server UCD plugin.

### Overview

You can use the HCL OneTest™ Server UCD plugin to integrate HCL OneTest™ Server with UCD.

You must have created the tests in the desktop clients and committed the test assets and test resources to a remote repository. The remote repository must be added to the project.

Depending on the type of tests you want to perform on your application, you must have created any or all of the following types of tests in the desktop clients for the application you are testing:

Type of test	Desk-top client
<ul style="list-style-type: none"> <li>Accelerated Functional Testing suites</li> <li>Compound tests</li> </ul>	HCL OneTest™ UI
<ul style="list-style-type: none"> <li>Test Suite</li> </ul>	HCL OneTest™ API
<ul style="list-style-type: none"> <li>Compound tests</li> <li>VU Schedule</li> <li>Rate Schedule</li> </ul>	HCL OneTest™ Performance

You can now follow the tasks listed in the task flow table to integrate HCL OneTest™ Server with UCD. See Task flow for integrating UrbanCode Deploy.

## Installing the HCL OneTest™ Server UCD plugin

You must install the latest version of the HCL OneTest™ Server UCD plugin to integrate HCL OneTest™ Server with UCD and run tests on the UCD server.

### Before you begin

You must have downloaded the HCL OneTest™ Server UCD plugin 3.0 from the [HCL® License & Delivery portal](#).

1. Open the UCD dashboard.
2. Click **Settings**.



3. Click **Automation Plugins** from the **Automation** pane.
4. Click **Load Plugin**.
5. Click **Choose File** to locate and **Open** the compressed HCL OneTest™ Server UCD plugin file.



**Note:** Do not extract the HCL OneTest™ Server UCD plugin compressed file contents.

6. Click **Submit**.

#### Result

The HCL OneTest™ Server UCD plugin is displayed in the **Automation Plugins** tab.

#### What to do next

You can run the tests for the application that are available in your HCL OneTest™ Server project on the UCD server. See [Running tests on the UCD server on page 537](#).

## Using a custom trust store for UCD integration

You can use a custom trust store in the HCL OneTest™ Server UCD plugin file to establish a trusted and secure connection between the UCD server and HCL OneTest™ Server.

#### Before you begin

You must have configured the certificate that is used by HCL OneTest™ Server as a trusted CA, and then install HCL OneTest™ Server. See [Installation of the server software on page 36](#).

#### About this task

If the SSL certificate assigned to HCL OneTest™ Server is signed by an internal Certified Authority (CA), then you must download and import the CA certificate to a custom trust store. You can then use the custom trust store in the **HCL OneTest™ Server test** process step to establish a trusted and secure connection between the UCD server and HCL OneTest™ Server.



**Note:** If the internal CA certificate is already imported to the default trust store that is used by the UCD server, you need not use a custom trust store.

1. Locate the default trust store file (`cacerts` file) in your JRE directory from your computer, and then copy the file to a location of your choice on your computer.
2. Run the following command from the command prompt or terminal to import the CA certificate to your custom trust store:

```
keytool -import -trustcacerts -file <path to the downloaded CA certificate with the file extension> -alias <custom label for the certificate> -keystore <path to the trust store>
```

For example,

```
keytool -import -trustcacerts -file C:\Users\ca file.crt -alias alias1 -keystore D:\cert\cacerts
```



**Note:** The default password of the trust store is *changeit*. It remains the same for the custom trust store. If you want to change the password, you can run the following command, and then enter the new password:

```
keytool -storepasswd -keystore <path to the trust store>
```

For example, `keytool -storepasswd -keystore D:\cert\cacerts`

## Results

You have successfully imported the downloaded CA certificate to the custom trust store.

## What to do next

You can then run the tests that are available in your HCL OneTest™ Server project from the UCD server.

## Creating a component in UrbanCode™ Deploy

You must create a component to include artifacts and processes. The artifacts include runnable files, images, databases, configuration instructions. Whereas the processes define the activities that components can perform.

### Before you begin

- You must be familiar with working with UrbanCode™ Deploy.
- You must have been granted access to UrbanCode™ Deploy.

1. Log in to UrbanCode™ Deploy, if you are not already logged in.

#### Result

The UrbanCode™ Deploy dashboard is displayed.

2. Click **Components**, and then click **Create Component**.
3. Enter a name for the component in the **Name** field.
4. Enter the details in the other optional fields based on your requirement, and then click **Save**.

#### Result

The component that you created is displayed.

## Results

You have created the component in UrbanCode™ Deploy.

## What to do next

You must create a process for the component in UrbanCode™ Deploy. See [Creating a process in UrbanCode Deploy on page 527](#).

---

### Related information

[IBM UrbanCode Deploy Documentation](#)

## Creating a process in UrbanCode™ Deploy

You must create a process for the component to include step properties for the test that you want to run from UrbanCode™ Deploy.

### Before you begin

- You must be familiar with working with UrbanCode™ Deploy.
- You must have performed the following tasks:
  - Been granted access to UrbanCode™ Deploy.
  - Created a component in UrbanCode™ Deploy. See [Creating a component in UrbanCode Deploy on page 526](#).

1. Log in to UrbanCode™ Deploy, if you are not already logged in.

#### Result

The UrbanCode™ Deploy dashboard is displayed.

2. Click **Components**.

#### Result

A list of components that are available in UrbanCode™ Deploy is displayed.

3. Select the component from the list for which you want to create a process.
4. Click the **Processes** tab, and then click **Create Process**.

#### Result

The **Create Process** dialog is displayed.

5. Enter a name for the process in the **Name** field.
6. Select **Operational (No Version Needed)** from **Process Type** drop-down list.
7. Verify the **Default Working Directory** field.

The **Default Working Directory** field defines the location that the agent uses to run the process. The default value is `${p:resource/work.dir}/${p:component.name}`.

Where `${p:resource/work.dir}` is the default working directory for the agent and `${p:component.name}` is the name of the component.

8. Click **Save**.

#### Result

The process that you created is listed in the **Processes** tab and the **Design** tab for the process is displayed.

### Results

You have created the process for the component in UrbanCode™ Deploy.

### What to do next

You must configure the process in UrbanCode™ Deploy. See [Configuring the process on page 528](#).

---

Related information

[IBM UrbanCode Deploy Documentation](#)

## Configuring the process

You must configure the process that you created for the component to organize the steps in the process, specify the properties of the steps, and connect them.

### Before you begin

- You must be familiar with working with UrbanCode™ Deploy.
- You must have performed the following tasks:
  - Been granted access to UrbanCode™ Deploy.
  - Created a component in UrbanCode™ Deploy. See [Creating a component in UrbanCode Deploy on page 526](#).
  - Created a process for the component in UrbanCode™ Deploy. See [Creating a process in UrbanCode Deploy on page 527](#).
  - Generated an offline user token from HCL OneTest™ Server. See [Generating an offline token on page 565](#).
  - Added the tests assets to the project on HCL OneTest™ Server.

### About this task

When you open any process to configure, the process is displayed in the process editor. The process editor lists the plugins and steps. The required **Start** and **Finish** steps represent the beginning and the end of the process and steps are automatically placed on the design area.

1. Log in to UrbanCode™ Deploy, if you are not already logged in.

#### Result

The UrbanCode™ Deploy dashboard is displayed.

2. Click **Components**.

#### Result

A list of components that are available in UrbanCode™ Deploy is displayed.

3. Select the component from the list in which you created the process.

4. Click the **Processes** tab.

#### Result

A list of processes that are available for the component is displayed.

5. Select the process from the list that you want to configure.

#### Result

The **Design** tab for the process is displayed.

6. Click **HCL OneTest Studio**, and then **HCL OneTest Server** from the left menu.
7. Drag the **Run HCL OneTest Server test** step, and then drop it into the design area.



**Note:** The selected test must be placed between **Start** and **Finish** steps.

8. Specify the properties for the selected test by performing the following steps:


- a. Click the **Edit** icon.

**Result**



The **Edit Properties for Run HCL OneTest Server test** dialog is displayed.


- b. Specify the properties for the selected test step by referring to the following tables:

The following table lists the required fields that you must provide to run the test from UrbanCode™ Deploy:

Fields	Action
<b>Name</b>	Enter the name for the test step.
<b>HCL OneTest Server URL</b>	Enter the URL of HCL OneTest™ Server.
<b>Offline Token</b>	Enter the offline token that you generated from HCL OneTest™ Server.
<b>Team Space Name</b>	Enter the name of the team space that contains the project.   <b>Note:</b> The license for the team space must be configured and you must be a member of that team space.
<b>Project Name</b>	Enter the name of the project that are available in the team space.
<b>Repository Link</b>	Enter the URL of the Git repository that you added to your project in HCL OneTest™ Server.
<b>File Path</b>	Enter the path of the test assets that you want to run.  You can find the path of the test assets from the <b>Execution</b> page in HCL OneTest™ Server

The following table lists the optional fields that you can provide to run the test from UrbanCode™ Deploy:

Fields	Description
<b>Branch Name</b>	Use this field to enter the name of the branch in the Git repository where you stored test assets.
<b>Custom Trust Store Password</b>	Use this field to enter the trust store password if you have modified the password while creating the custom trust store.   <b>Note:</b> If you have not modified the trust store password, you can retain the <b>Custom Trust Store Password</b> field blank.
<b>Custom Trust Store Path</b>	Use this field to enter the file path of your trust store followed by the file name if HCL OneTest™ Server uses an internal CA certificate and you have imported the certificate to a custom trust store.
<b>Datasets</b>	Use this field to enter the path to the dataset if you want to replace the values of the dataset during a test run.  You must ensure that both original and new datasets are in the same workspace and have the same column names. When you enter a value for the <b>Datasets</b> field, you must also include the path to the dataset. The path must be in the following format:  <code>/project_name/ds_path/original_ds.csv:/project_name/ds_path/new_ds.csv</code>   <b>Note:</b> You can override multiple datasets that are saved in a different project by adding multiple paths to the dataset separated by a semicolon.
<b>Environment</b>	Use this field to enter the environment details for the following types of test assets: <ul style="list-style-type: none"> <li>▪ APISUITE</li> <li>▪ APITEST</li> <li>▪ APISTUB</li> </ul>
<b>Labels</b>	Use this field to add labels to test results when the test run is complete.  You can add multiple labels to a test result separated by a comma.

Fields	Description
	<p>For example, <i>label1</i>, <i>label2</i></p> <p>After the test run is complete, then the values provided in the <b>Labels</b> field are displayed on the <b>Results</b> page of HCL OneTest™ Server.</p>
<b>Secrets Collection Name</b>	<p>Use this field to enter the name of the Secret if you created secrets collections for your project and to use Secrets at test run time.</p> <p>This field is mandatory only if you want to run the following types of test assets:</p> <ul style="list-style-type: none"> <li>▪ APISUITE</li> <li>▪ APITEST</li> <li>▪ APISTUB</li> </ul>
<b>Show Hidden Properties</b>	<p>Select this checkbox if you want to use an HTTP Proxy to connect to HCL OneTest™ Server.</p> <p>After you select the <b>Show Hidden Properties</b> checkbox, you can configure the HTTP Proxy by using the following fields:</p> <ul style="list-style-type: none"> <li>▪ <b>HTTP Proxy Host:</b> Use this field to enter the hostname of the HTTP proxy that you want to connect to HCL OneTest™ Server.</li> <li>▪ <b>HTTP Proxy Port:</b> Use this field to enter the port number where the HTTP proxy is used to listen to both HTTP and HTTPS traffic.</li> <li>▪ <b>HTTP Proxy User name:</b> Use this field to provide the username for the HTTP proxy.</li> <li>▪ <b>HTTP Proxy Password:</b> Use this field to enter a valid password for the user that you specified in the <b>HTTP Proxy User name</b> field.</li> </ul> <p> <b>Note:</b> You can use the <b>Show Hidden Properties</b> field only when you want to run the following types of test assets:</p> <ul style="list-style-type: none"> <li>▪ APISUITE</li> <li>▪ APITEST</li> <li>▪ APISTUB</li> </ul>
<b>Start Date</b>	<p>Use this field to enter the date and time in the following format for running tests at the scheduled date and time:</p> <p>yyyy/MM/dd/HH:mm</p>

Fields	Description
	When you enter a value in the <b>Start date</b> field, the status of the test displays as <i>Scheduled</i> on the <b>Overview</b> and <b>Progress</b> pages of HCL OneTest™ Server.
<b>Variables</b>	<p>Use this field to enter the name of the variable and its value if your test requires variables during the test run time.</p> <p>You must enter the variables in the following format:</p> <pre>name_of_the_variable=value_of_the_varibale</pre> <p>You can add multiple variables to the test run separated by a semicolon.</p> <p>For example, <i>varname1=value1;varname2=value2</i></p>



**Note:** In addition to optional fields, you can also use the following fields from UrbanCode™

Deploy to configure your test run:

- **Working Directory**
- **Precondition**
- **Post Processing Script**
- **Use Impersonation**
- **Auth Token Restriction**

You can accept the default values or change the values based on your requirements. For more information about these fields, see the related links.

c. Click **OK** to save the properties for the test.

9. Click **Save** in the design area.

## Results

You have configured the process for the component in UrbanCode™ Deploy.

## What to do next

You must create a resource in UrbanCode™ Deploy. See [Creating a resource in UrbanCode Deploy on page 533](#).

---

Related information

[IBM UrbanCode Deploy Documentation](#)

[Process step preconditions](#)

[Post-processing scripts](#)



[User impersonation for process steps](#)

[Restricting authentication tokens](#)

## Creating a resource in UrbanCode™ Deploy

You must create a resource to associate agents with components that you created in UrbanCode™ Deploy.

### Before you begin

- You must be familiar with working with UrbanCode™ Deploy.
  - You must have been granted access to UrbanCode™ Deploy.
1. Log in to UrbanCode™ Deploy, if you are not already logged in.

#### Result

The UrbanCode™ Deploy dashboard is displayed.

2. Click **Resources**, and then click **Create Top-Level Group**.

#### Result

The **Create Resource** dialog is displayed.

3. Enter a name for the resource in the **Name** field.
4. Click **Save**.

### Results

You have created the resource in UrbanCode™ Deploy.

### What to do next

You must configure the resource. See [Configuring the resource on page 533](#).

---

Related information

[IBM UrbanCode Deploy Documentation](#)

## Configuring the resource

You must configure the resource to add an agent and associate the agent with the component.

### Before you begin

- You must be familiar with working with UrbanCode™ Deploy.
- You must have performed the following tasks:

- Been granted access to UrbanCode™ Deploy.
  - Created a component in UrbanCode™ Deploy. See [Creating a component in UrbanCode Deploy on page 526](#).
  - Created a resource in UrbanCode™ Deploy. See [Creating a resource in UrbanCode Deploy on page 533](#).
1. Log in to UrbanCode™ Deploy, if you are not already logged in.  
**Result**  
The UrbanCode™ Deploy dashboard is displayed.
  2. Click **Resources**.  
**Result**  
A list of resources that are available in UrbanCode™ Deploy is displayed.
  3. Perform the following steps to add an agent to the resource:
    - a. Click the resource from the list for which you want to add an agent.
    - b. Click the **Actions** icon from the last column, and then click **Add Agent**.
    - c. Select the agent from the drop-down list.



**Note:** The **Name** field is auto populated with the name of the agent.

- d. Click **Save**.  
**Result**  
The selected agent is added to the resource and you can view the status of the agent in the **Status** column.
4. Perform the following steps to add a component to the agent:
    - a. Click the agent from the list for which you want to add a component.
    - b. Click the **Actions** icon from the last column, and then click **Add Component**.
    - c. Select the component from the drop-down list.



**Note:** The **Name** field is auto populated with the name of the component.

- d. Click **Save**.  
**Result**  
The selected component is added to the agent.

## Results

You have configured the resource in UrbanCode™ Deploy.

**What to do next**

You must create an application. See [Creating an application in UrbanCode Deploy on page 535](#).

---

Related information

[IBM UrbanCode Deploy Documentation](#)

## Creating an application in UrbanCode™ Deploy

You must create an application to fetch all the components together that you want to deploy.

**Before you begin**

- You must be familiar with working with UrbanCode™ Deploy.
  - You must have been granted access to UrbanCode™ Deploy.
1. Log in to UrbanCode™ Deploy, if you are not already logged in.

**Result**

The UrbanCode™ Deploy dashboard is displayed.

2. Click **Applications**.
3. Click **Create Applications**, and then **New Applications**.
4. Enter a name for the application in the **Name** field.

**Result**

The **Environments** page for the application that you created is displayed.

**Results**

You have created the application in UrbanCode™ Deploy.

**What to do next**

You must configure the application. See [Configuring the application on page 535](#).

---

Related information

[IBM UrbanCode Deploy Documentation](#)

## Configuring the application

You must configure the application to associate resources with environments and define processes to run test assets.

**Before you begin**

- You must be familiar with working with UrbanCode™ Deploy.
- You must have performed the following tasks:

- Been granted access to UrbanCode™ Deploy.
  - Created a component in UrbanCode™ Deploy. See [Creating a component in UrbanCode Deploy on page 526](#).
  - Created a process for the component in UrbanCode™ Deploy. See [Creating a process in UrbanCode Deploy on page 527](#).
  - Configure the process for the component in UrbanCode™ Deploy. See [Configuring the process on page 528](#).
  - Created a resource in UrbanCode™ Deploy. See [Creating a resource in UrbanCode Deploy on page 533](#).
  - Configured the resource in UrbanCode™ Deploy. See [Configuring the resource on page 533](#).
  - Created an application in UrbanCode™ Deploy. See [Creating an application in UrbanCode Deploy on page 535](#).
1. Log in to UrbanCode™ Deploy, if you are not already logged in.  
**Result**  
The UrbanCode™ Deploy dashboard is displayed.
  2. Click **Applications**.  
**Result**  
A list of applications that are available in UrbanCode™ Deploy is displayed.
  3. Click the application that you want to configure from the **Name** column.  
**Result**  
The **Environments** page for the selected application is displayed.
  4. Perform the following steps to create an environment for the application that you selected:
    - a. Click **Create Environment**.
    - b. Enter a name for the environment in the **Name** field.
    - c. Click **Save**.
  5. Perform the following steps to configure resources to the environment:
    - a. Click the environment that you created.
    - b. Click **Add Base Resources**.  
**Result**  
A list of resources that are available in UrbanCode™ Deploy is displayed.
    - c. Select the checkbox to add resources to the environment.
    - d. Click **Save**.

**Result**

You can view the corresponding agent and the component that you added for the resource by using the **Expand** icon.

6. Perform the following steps to add the component to the application:
  - a. Click **Applications**, and then select your application from the list.
  - b. Click the **Components** tab, and then **Add Components**.
  - c. Select the checkbox from the drop-down list to add components to the application.
  - d. Click **Save**.
7. Perform the following steps to create a process for the application:
  - a. Click the **Processes** tab, and then **Create Process**.
  - b. Enter a name for the process in the **Name** field.
  - c. Click **Save**.

**Result**

The **Design** tab for the process that you created is displayed.

8. Drag the component process listed under the **Component Process Steps** option from the left navigation pane and drop it into the design area.
9. Select the component process from the drop-down list in the **Operational (No Version Needed) Process** field.
10. Click **Save**.
11. Click the **Edit** icon, and then change the name of the properties.
12. Click **OK**, and then click **Save**.

**Results**

You have configured the application to run test assets from UrbanCode™ Deploy.

**What to do next**

You can run test assets from UrbanCode™ Deploy. See [Running HCL OneTest Server tests on the UCD server on page 537](#).

---

 Related information

[IBM UrbanCode Deploy Documentation](#)

## Running HCL OneTest™ Server tests on the UCD server

After you install the HCL OneTest™ Server UCD plugin on the UCD server, you can create a `process request` that contains the test for your application, and then run the test on the UCD server.

**Before you begin**

You must have completed the following tasks:

- Added the tests that you created in the desktop clients for your application to the project on HCL OneTest™ Server.
- Installed the latest version of the HCL OneTest™ Server UCD plugin. See [Installing the plugin on page 524](#).
- Generated an offline user token from HCL OneTest™ Server. See [Generating an offline token on page 565](#).

### About this task

After you have installed the HCL OneTest™ Server UCD plugin on the UCD server, you can either use an existing component in your project or create a component. You can create a component process and select the **HCL OneTest™ Server test** step to edit the *step properties* for the test you want to run. After selecting the agent, you can create an application. You can then create an application process for the application, and then submit the application process for a run.

1. Log in to the UrbanCode Deploy (UCD) server, if you are not already logged in.
2. Click **Components** from the UCD dashboard, and then click **Create Component** to create a component.



**Note:** You can either use an existing component or create a component.

3. Create a component process in the component by performing the following steps:
  - a. Open the component that you created.
  - b. Click the **Processes** tab from the component dashboard, and then click **Create Process**.

#### Result

The **Create Process** dialog box is displayed.

- c. Enter the required values to create a component process and click **Save**.



**Note:** All mandatory fields are marked with an asterisk (\*) in the UI.

- i. Enter the process name in the **Name** field.
- ii. Select **Operational (No Version Needed)** from the **Process Type** list.



**Note:** The **Default Working Directory** field displays the folder path where the agent can download the artifacts and create temporary files.

The process that you created is listed in the **Processes** list and the **Design** tab for the process is displayed.



**Note:** The process opens in the process editor. The process editor lists the plugins and steps. The required **Start** and **Finish** steps represent the beginning and the end of the process and are automatically placed on the design area.

4. Select the process step you want to run by completing the following steps:
  - a. Search for the **HCL OneTest™ Server test** process step from the left design pane.
  - b. Select the **Run HCL OneTest™ Server test** process step and drag the test into the design area.

**Result**

The selected test is placed in between the **Start** and **Finish** steps.

5. Specify the properties for the selected test by performing the following steps:

- a. Click the **Edit** icon .

**Result**





The **Edit Properties for Run HCL OneTest™ Server test** dialog box of the selected test is displayed.

- b. Specify the properties for the selected test step by following the action in the table that follows.



**Note:** All mandatory fields are marked with an asterisk (\*) in the UI.

Field	Action required for a HCL OneTest™ Server test
Name	Enter the name of the step.
HCL OneTest™ Server URL	Enter the URL for HCL OneTest™ Server.
Offline Token	Enter the offline token that you generated from HCL OneTest™ Server.
Team Space Name	Enter the name of the team space that contains the project.

Field	Action required for a HCL OneTest™ Server test
	 <b>Note:</b> The license on the team space must be configured and you must be a member of that team space.
Project Name	Enter the name of the project from the available names in the team space.   <b>Note:</b> You must be an Owner or a Tester of the project that is available in the team spaces to run the tests from the UCD server.
Branch Name	Select the branch where the test assets are stored.
Repository Link	Enter the repository path for the test to run.
File Path	Enter the file path of the HCL OneTest™ Server test that you want to run.
Environment	Enter the HCL OneTest™ API environment details for the test.   <b>Note:</b> This field is applicable only for an API test.
Field	Optional action for a HCL OneTest™ Server test
Custom Trust Store Path	Enter the file path of your trust store followed by the file name in the <b>Custom Trust Store Path</b> field if the HCL OneTest™ Server uses an internal CA certificate and you have imported the certificate to a custom trust store. <sup>2</sup>
Custom Trust Store Password	Enter the trust store password if you have modified the password while creating the custom trust store. <sup>2</sup>   <b>Note:</b> If you have not modified the trust store password, you can keep the <b>Custom Trust Store Password</b> field blank.
Working Directory	Specify an alternative path to the working directory for this step. <sup>2</sup>
Precondition	Specify any conditions that are to be completed before the test runs. You can edit the script by clicking the script displayed. <sup>2</sup>
Post Processing Script	Specify if you want to run any scripts after the completion of the test run. You can click <b>New</b> to add new scripts. <sup>2</sup>

2. You need not set this property for a step when you are running an HCL OneTest™ Server test.



Field	Action required for a HCL OneTest™ Server test
Use Impersonation checkbox	Select this checkbox to run the test as a different user. <sup>2</sup>
Auth Token Restriction	Set the authentication token actions by applying token restrictions.  The <b>System Default</b> is selected by default. You can add a new token restriction or edit the one already added. <sup>2</sup>

c. Click **OK** to save the properties for the test.


6. Click **Save** in the design area.

7. Click the **Resources** tab from the UCD dashboard and create a resource by clicking **Create Top-Level Group**.


#### Result

The created resource is displayed on the **Resource Tree** tab page.

8. Select the agent that runs the test by completing the following steps:

 **Important:** You must have already installed the agent on the UCD server that you want to use.

a. Select the resource displayed on the **Resource Tree** tab page.


b. Click the **Horizontal ellipsis** icon  for the selected resource.

c. Click **Add Agent**.

d. Select the agent to add to the resource, and then click **Save**.

#### Result

The selected agent is added to the resource in the **Resource Tree** pane and the status of the agent can also be viewed.

 **Important:** The agent must be **Online** for the test to run.

9. Add the component to the agent by performing the following steps:

a. Click the **Horizontal ellipsis** icon  for the agent.

b. Click **Add Component** on the list.

c. Select the component to add to the resource, and then click **Save**.

#### Result

The selected component is added to the agent for the resource in the **Resource Tree** pane.

10. Create an application by completing the following steps:

- a. Click the **Applications** tab from the UCD dashboard.
- b. Click **Create Application**.
- c. Complete the details in the **Create Application** dialog box, and then click **Save**.

**Result**

The **Environments** tab page is displayed for the created application.

- d. Click **Create Environment** to create an environment for the application that you created.
- e. Complete the details in the **Create Environment** dialog box, and then click **Save**.

**Result**

The environment that you created is displayed.

- f. Click the environment to open, and then click **Add Base Resources**.
- g. Select the resource from the list in the **Add Resource to Environment** dialog box, and then click **Save** to add the resource to the environment.

**Result**

The resource added to the environment is displayed.



**Note:** When you add a resource to an environment, the corresponding agent and the component are displayed for the resource.

- h. Click the application from the `breadcrumbs`.

**Result**

The **Environments** tab page is displayed for your application.

- i. Add the component to the application by performing the following steps:
  - i. Click the **Components** tab.
  - ii. Click **Add Component**.
  - iii. Select the component from the list in the **Add a Component** dialog box, and then click **Save**.

**Result**

The selected component is displayed on the **Components** tab page.

- j. Create a process for the application by performing the following steps:
  - i. Click the **Processes** tab.
  - ii. Click **Create Process**.
  - iii. Complete the details in the **Create an Application Process** dialog box, and then click **Save**.

**Result**

The **Design** tab page for the application process that you created is displayed.


- k. Select the component process from the left pane and drag it into the design area.



**Note:** You can click the **Edit** icon  to add the properties, if required.

- l. Click **Save** in the design area.

11. Select the application process to run the test by completing the following steps:

- a. Click **Applications** from the UCD dashboard.
- b. Click the application that you configured for a test run.
- c. Click the **Request Process** icon .

**Result**

The **Run Process on <environment name>** window is displayed.

- d. Select the application process that contains the test from the **Process** list.
- e. Click **Submit**.

**Result**

The UCD dashboard shows the progress of the application process request.


## Results

You have used the HCL OneTest™ Server UCD plugin to integrate HCL OneTest™ Server with UCD and run the test from your project on the UCD server.

After the UCD process request runs successfully, you can view the status of the completed process request displayed as follows:

- **Success:** When the test run is successful
- **Failed:** When the test run is failed

## What to do next

- You can view the details of the test run as a process from the UCD dashboard by performing the following action:
  - Expand the *step*. You can then expand the application process. You can then hover over the process, and then click the **Output Log** icon . The output log is displayed. You can verify the log details.
- You can also view the test reports and logs of the test that was run on the UCD server from the **Results** page on HCL OneTest™ Server. See [Test results and reports overview on page 436](#).

## Integration with other applications

You can integrate certain applications with HCL® OneTest™ Data to generate the test data.

You can find instructions to integrate other applications with HCL® OneTest™ Data.

## Generating the test data by using Jenkins

When you perform a test during the continuous integration and continuous deployment process on Jenkins, you might want to generate the test data. You can generate random test data to test your application by integrating Jenkins with HCL® OneTest™ Data by using the HCL® OneTest™ Data Jenkins Plugin.

### Before you begin

You must have completed the following tasks:

- Installed the latest version of Jenkins.
- You must have an account in the Jenkins application.
- Downloaded the latest version of **HCL OneTest Data Jenkins Plugin** from the [HCL License & Delivery portal](#).
- Installed **HCL OneTest Data Jenkins Plugin** in Jenkins.



**Note:** You must restart the Jenkins application after installing the **HCL OneTest Data Jenkins Plugin**.

- Logged in to HCL OneTest™ Server.
- Created a project and a schema in HCL® OneTest™ Data.
- Established a connection between HCL® OneTest™ Data and a supported database. See
  - [Establishing a JDBC connection with HCL OneTest Data on page 194](#)
  - [Establishing a MongoDB connection with HCL OneTest Data on page 206](#)

### About this task



You can write the test data generated by integrating Jenkins with HCL® OneTest™ Data into a file or any supported database. After integration with Jenkins, **HCL OneTest Data Jenkins Plugin** supports the insertion of the generated test data in both JDBC supported database and MongoDB. To write the generated test data in the database, you must establish a connection between HCL® OneTest™ Data and the supported database.


1. Log in to the Jenkins application.
2. Create a Jenkins free-style software project.


For more details about how to create a project in Jenkins, refer to the related link.

The project dashboard is displayed.

3. From the project dashboard, perform the following steps:
  - a. Click the **Add build step** list under **Build** and select **Run an HCL OneTest Data Generation**.
  - b. Set the properties for the **HCL OneTest Data Jenkins Plugin** for HCL OneTest Data by referring to the following table:

Field	Action	Required/Optional
Name	Enter the name of the build.	Required
Use Custom Trust Store	<p>Select the checkbox if the SSL certificate is available in the custom trust store.</p> <p> <b>Note:</b> If you do not find details of the custom trust store in the application plugin, then the application navigates to the default trust store to access the certificate.</p> <p>The default trust store is at the following location: <code>\$JAVA_HOME/jre/lib/security/cacerts</code></p>	Optional
Custom Trust Store Path	Enter the path of the custom trust store where the certificate is stored.	Required, if <b>Use Custom Trust Store</b> is selected.
Use Custom Password for the Trust Store	<p>Select the checkbox if the default password for the custom trust store is changed when you place the certificate in the custom trust store.</p> <p> <b>Note:</b> The default password to access the custom trust store is <code>changeit</code>.</p>	Optional
Custom Trust Store Password	Enter the password to access the custom trust store.	Required, if <b>Use Custom Password for the Trust Store</b> is selected.
Server URL	<p>Enter the URL of HCL OneTest™ Server.</p> <p>The format for the URL is as follows: <code>https://&lt;fully-qualified-dns-name&gt;/</code></p>	Required
Offline Token	Enter the offline token that is generated in HCL OneTest™ Server.	Required
Project	Select the name of the project from the list of your projects.	Required

Field	Action	Required/Optional
Schema	Select the name of the schema from the list of schemas associated with the project you selected.	Required
Root Element	Specify the root path of the element for which you want to generate the test data.  For example, Root:NewType1	Required
Number of Records	Enter the number of records you want to generate.  This field accepts only numbers.	Required
Numeric Seed Value	Enter the seed value that acts as an instance of random data when you generate the test data.  This field accepts both positive and negative numbers.	Optional
Output Data Storage	Select the data storage type. The data storage is a location where you want the generated test data to be written.  You can select <code>FILE</code> , <code>JDBC</code> , or <code>MONGODB</code> as a data storage type.  <code>FILE</code> : The generated test data is written into a file and you can download it in your local file system.  <code>JDBC</code> : The generated test data is written in the selected JDBC supported database.  <code>MONGODB</code> : The generated test data is written into MongoDB.	Required
Connection Names	 <b>Note:</b> This field is enabled only when you select <code>JDBC</code> as the data storage type.  Select the connection name from the populated list of connection names.	Required
Output Format	Select the output file format of the generated test data from the populated list of the output formats. The output file format is based on the schema you selected.	Required

Field	Action	Required/Optional
Data File Location	<p>Specify the location of the output file. If the specified location is invalid, by default, the output file is saved in the HCL® OneTest™ Data server.</p> <p> <b>Notes:</b></p> <ul style="list-style-type: none"> <li>You can find the output file in the HCL® OneTest™ Data pod at the following location:           <pre style="margin-left: 20px;">/opt/hcl/hip-rest/output/&lt;accountId&gt;/&lt;userId&gt;/&lt;projectId&gt;/&lt;schemaId&gt;/&lt;genMapPath&gt;</pre> </li> <li>This field is not applicable if you select the data storage type as <code>JDBC</code>.</li> </ul>	Optional

c. Click **Save**.

4. From the Jenkins dashboard, select the project and click **Build Now**.

## Results

You have successfully generated the test data by using the **HCL OneTest Data Jenkins Plugin**.



**Note:** If the test data generation request fails, you can view the test data generation logs. See [Viewing the test data generation logs on page 548](#).

## What to do next

After the build completes, you can perform the following tasks:

- If you selected `FILE` as a data storage type, the generated test data is downloaded in the local file system at the specified location.
- If you selected `JDBC` as the data storage type, then you can use the generated test data from the database.
- If you selected `MONGODB` as the data storage type, then you can use the generated test data from the database.

---

### Related information

[Building a software project](#)

## Viewing the test data generation logs

After the Jenkins build generates the test data, you can view the details of the test data generation from the logs.

1. Click the build number from **Build History**.
2. Click **Console Output** to open the console for the project.

### Exemple

The following sample shows the logs of the generated test data when you select `FILE` as the data storage type:

```

Started by user
Running as SYSTEM
Building in workspace C:\Program Files (x86)\Jenkins\workspace\HelloWorld

----- START Build Step -----
Master/Slave details
System Information : LP1-AP-51837142/10.115.94.185
Windows OS

Data Generation information:
Server URL: https://otd-build.nonprod.hclnp.com/
Project: testing
Schema: schema
Data Location:
Seed Value: 10
Root Element: Root:NewType1

Include Header: true
No Of Records: 10
Data Storage: FILE
Output Format: Excel

Status: Starting the data generation..

Data Generation Response is
{"code":200,"project_id":"4400","schema_id":"5ebd2915612cae00fe8e48dd","connectionURL":null,"
dbSchemaName":null,"message":"Data generation
succeeded.","data_location":"5ebd2906612cae00fe8e48db_e584d146-db2e-4b5b-bb15-495b1a53a18a_GL
zXjhMqlX_1","timestamp":"2020-05-14T13:14:52.623Z","tableName":null}
Status: Test data generation completed successfully

Status: Downloading the Generated Test Data

Not able to save the file as the location was invalid/blank
File is available
in /
opt/hcl/hip-rest/output/5ebbc065612cae002d82b141/5ebd2906612cae00fe8e48db/4400/5ebd2915612cae0
0fe8e48dd/e584d146-db2e-4b5b-bb15-495b1a53a18a/GLzXjhMqlX.xlsx location of OTD Server
----- END Build Step -----
Finished: SUCCESS

```



The following sample shows the logs of the generated test data when you select `JDBC` as the data storage type:

```

Started by user1
Running as SYSTEM
Building in workspace C:\Program Files (x86)\Jenkins\workspace\HelloWorld6
----- START Build Step -----
Master/Slave details
System Information : LP1-AP-51837142/10.115.94.185
Windows OS

Test data generation information:
Server URL: https://otd-build.nonprod.hclnp.com/
Project: otdproject
Schema: anonymous_schema
Data Location:
Seed Value: 10
Root Element: DB:Table:Row

Include Header: false
No Of Records: 10
Data Storage: JDBC
Connection Name: SampleJDBC
Output Format: Native

Status: Starting the test data generation..

Test data generation response:
["code":200,"project_id":"1050","schema_id":"5e830314e4332f0192e3487e","connectionURL":"jdbc:
mysql://10.134.198.10:8081","dbSchemaName":"sampleSchema","message":"Data generation
succeeded.","data_location":null,"timestamp":"2020-03-31T09:06:48.005Z","table name":"tasks"]

Status: Test data generation completed successfully
Data is written into the database
Connection URL: jdbc:mysql://10.134.198.10:8081
Database schema name: sampleSchema
Table name: tasks

```

The following sample shows the logs of the generated test data when you select `MONGODB` as the data storage type:

```

Started by user
Running as SYSTEM
Building in workspace C:\Program Files (x86)\Jenkins\workspace\HCLOneTestData Integration
----- START Build Step -----
Master/Slave details
System Information : LP1-AP-51837142/10.115.94.185
Windows OS

Data Generation information:
Server URL: https://otd-fvt1.nonprod.hclnp.com/
Project: testproject
Schema: EmployeeDetails
Data Location:

```

```

Seed Value: 1
Root Element: Root:Employee

Include Header: false
No Of Records: 1
Data Storage: MONGODB
Connection Name: TestMongo
Output Format: Json

Status: Starting the data generation..

Data Generation Response is
{"code":200,"databaseName":"oneTestDataDB","project_id":"2850","schema_id":"5efedcb33d358500a1662993","connectionURL":"{my-ots}-mongodb:27017","dbSchemaName":null,"message":"Data generation succeeded.","data_location":null,"timestamp":"2020-07-03T07:29:06.055Z","tableName":null,"collectionName":"employee"}

Data is written into Mongo Database
Connection URL: {my-ots}-mongodb:27017
Database Name: oneTestDataDB
DB Collection Name: employee

Status: Test data generation completed successfully
----- END Build Step -----
Finished: SUCCESS

```

## Generating the test data by using UrbanCode™ Deploy

When you perform a test while you deploy an application on UrbanCode™ Deploy, you might want to generate test data. You can generate a random test data to test your application by integrating HCL® OneTest™ Data UCD Plugin with HCL® OneTest™ Data.

### Before you begin

You must have completed the following tasks:

- Installed the latest version of the UrbanCode™ Deploy server and agent on the target system.
- Configured the agent and verified that the agent is running on the target system. For information about how to configure the agent, refer to [Configure the agent and target system](#).
- Downloaded the latest version of HCL® OneTest™ Data UCD Plugin from the [HCL License & Delivery portal](#).
- Installed HCL® OneTest™ Data UCD Plugin on the UrbanCode™ Deploy server. See [Installing the HCL OneTest Data UCD Plugin on page 553](#).
- Created an account on the UrbanCode™ Deploy server.
- Logged in to HCL OneTest™ Server.
- Created a project and a schema in HCL® OneTest™ Data.
- Established a connection between HCL® OneTest™ Data and a supported database. See
  - [Establishing a JDBC connection with HCL® OneTest™ Data on page 194](#)
  - [Establishing a MongoDB connection with HCL OneTest Data on page 206](#)



## About this task

You can write the test data generated by integrating HCL® OneTest™ Data UCD Plugin with HCL® OneTest™ Data into a file or any supported database. After integration with HCL® OneTest™ Data UCD Plugin, HCL® OneTest™ Data supports the insertion of the generated test data in both JDBC supported database and MongoDB. To write the generated test data in the database, you must establish a connection between HCL® OneTest™ Data and the supported database.

## Procedure

1. Log in to the HCL UrbanCode Deploy server.
2. Create a component.
3. Create a component process, and then set the properties for the component by referring to the following table:

Field	Action	Required/Optional
Name	Enter a name for the HCL UrbanCode Deploy application process.	Required
Server URL	Enter the URL of HCL OneTest™ Server.  The format for the URL is as follows: https://<fully-qualified-dns-name>/	Required
Offline Token	Enter the offline token that is generated in HCL OneTest™ Server.	Required
Project	Enter the name of your project.	Required
Schema	Enter the name of the schema associated with the project you selected.	Required
Root Element	Specify the root path of the element for which you want to generate the test data.  For example, Root:NewType1	Required
Number of Records	Enter the number of records you want to generate.	Required
Numeric Seed Value	Enter the seed value that acts as an instance of random data when you generate the test data.	Optional
Data Storage	Select the data storage type. The data storage is a location where you want the generated test data to be written.  You can select <code>FILE</code> , <code>JDBC</code> , or <code>MONGODB</code> as a data storage type.	Required

Field	Action	Required/Optional
	<p><b>FILE:</b> The generated test data is written into a file and you can download it in your local file system.</p> <p><b>JDBC:</b> The generated test data is written in the selected JDBC supported database.</p> <p><b>MONGODB:</b> The generated test data is written into MongoDB.</p>	
Connection Name	<p> <b>Note:</b> This field is applicable only when you select <code>JDBC</code> or <code>MONGODB</code> as the data storage type.</p> <p>Enter the name of the <code>JDBC</code> or <code>MONGODB</code> connection.</p>	Required
Output Format	Specify the file format of the generated test data.	Required
Data File Location	<p>Specify the location for the output file. If the specified location is invalid, by default, the output file is saved in the HCL® OneTest™ Data server.</p> <p> <b>Notes:</b></p> <ul style="list-style-type: none"> <li>You can find the output file in the HCL® OneTest™ Data pod at the following location: <pre> /opt/hcl/hip-rest/output/&lt;accountId&gt;/&lt;userId&gt;/&lt;projectId&gt;/&lt;schemaId&gt;/&lt;genMapPath&gt; </pre> </li> <li>This field is not applicable if you select the data storage type as <code>JDBC</code> or <code>MONGODB</code>.</li> </ul>	Optional



**Note:** You can ignore the following property fields while you set up the integration of HCL UrbanCode Deploy with HCL® OneTest™ Data:

- Working Directory
- Post Processing Script
- Precondition
- Use Impersonation

4. Create a resource and select the agent.
5. Add the component that you created to the agent.

6. Create an application.
7. Create an environment from the **Applications** dashboard.
8. Add the resource and the component that you created to the environment.
9. Create a process for the application by clicking the **Processes** tab from the **Applications** dashboard.

The page of the application process is displayed.

10. Click the component process that you created in step 3 from the **Component Process Steps** on the left navigation pane and drag it into the design area.
11. Click **Save**.
12. Go to the **Applications** dashboard, and then click **Request Process** for the environment of the application process that you want to execute.

The **Run Process on environment name** dialog box is displayed.

13. Select the application process that you want to execute and click **Submit**.

The HCL UrbanCode Deploy dashboard shows the progress of the application process request to generate the test data.

## Result

You have successfully generated the test data by using the HCL® OneTest™ Data UCD Plugin for HCL® OneTest™ Data.

You can view the completed request process with the status displayed as **Success** or **Failed**.



### Note:

If the test data generation request fails, you can view the logs of the process. See [Viewing the UrbanCode Deploy logs on page 554](#)

## What to do next

After the successful completion of process, you can perform the following tasks:

- If you selected `FILE` as a data storage type, the generated test data is downloaded in the local file system at the specified location.
- If you selected `JDBC` as the data storage type, then you can use the generated test data from the database.
- If you selected `MONGODB` as the data storage type, then you can use the generated test data from the database.

## Installing the HCL® OneTest™ Data UCD Plugin

When you want to generate the test data by using the HCL® OneTest™ Data UCD Plugin for HCL® OneTest™ Data, you must install the HCL® OneTest™ Data UCD Plugin on the UrbanCode™ Deploy server.

1. From **Settings**, click **Automation Plugins**.
2. Click **Load Plugin**.
3. Enter the path of the compressed plug-in file, and then click **Submit**.

## Results

The plug-in is listed on the **Automation Plugins** pane.

## What to do next

After you successfully installed the HCL® OneTest™ Data UCD Plugin on the server, you must configure the HCL® OneTest™ Data UCD Plugin and generate the test data. See [Generating the test data by using UrbanCode Deploy on page 550](#).

## Viewing the UrbanCode™ Deploy logs

After completion of the generation of test data, you can view the details of the process in the UrbanCode™ Deploy console log.

You can click the **Output Log** icon from the console log of the UrbanCode Deploy dashboard to view the output log.

The following sample shows the logs of the generated test data when you select `FILE` as the data storage type:

```

=====
plugin: HCL OneTest Data UCD Plugin, id: com.urbancode.air.plugin.otd, version: 1
plugin command: 'cmd' '/C' '"D:\UCD_Agent\opt\groovy-1.8.8\bin\groovy.bat -cp
"D:
\UCD_Agent\var\plugins\com.urbancode.air.plugin.otd_1_200b8a9ae7959125d32be1cde7aa20472d2b47c4
de3754dbd7627bd6778d33d0\classes;D:\UCD_Agent\var\plugins\com.urbancode.air.plugin.otd_1_200b8
a9ae7959125d32be1cde7aa20472d2b47c4de3754dbd7627bd6778d33d0\lib\commons-codec-1.10.jar;D:\UCD_
Agent\var\plugins\com.urbancode.air.plugin.otd_1_200b8a9ae7959125d32be1cde7aa20472d2b47c4de375
4dbd7627bd6778d33d0\lib\commons-logging-1.2.jar;D:\UCD_Agent\var\plugins\com.urbancode.air.plu
gin.otd_1_200b8a9ae7959125d32be1cde7aa20472d2b47c4de3754dbd7627bd6778d33d0\lib\groovy-all-1.8.
4.jar;D:\UCD_Agent\var\plugins\com.urbancode.air.plugin.otd_1_200b8a9ae7959125d32be1cde7aa2047
2d2b47c4de3754dbd7627bd6778d33d0\lib\groovy-plugin-utils-1.2.jar;D:\UCD_Agent\var\plugins\com.
urbancode.air.plugin.otd_1_200b8a9ae7959125d32be1cde7aa20472d2b47c4de3754dbd7627bd6778d33d0\
lib\gson-2.8.6.jar;D:\UCD_Agent\var\plugins\com.urbancode.air.plugin.otd_1_200b8a9ae7959125d32
be1cde7aa20472d2b47c4de3754dbd7627bd6778d33d0\lib\hamcrest-core-1.1.jar;D:\UCD_Agent\var\plug
ins\com.urbancode.air.plugin.otd_1_200b8a9ae7959125d32be1cde7aa20472d2b47c4de3754dbd7627bd6778
d33d0\lib\httpClient-4.5.6.jar;D:\UCD_Agent\var\plugins\com.urbancode.air.plugin.otd_1_200b8a9
ae7959125d32be1cde7aa20472d2b47c4de3754dbd7627bd6778d33d0\lib\httpcore-4.4.10.jar;D:\UCD_Ag
ent\var\plugins\com.urbancode.air.plugin.otd_1_200b8a9ae7959125d32be1cde7aa20472d2b47c4de3754d
bd7627bd6778d33d0\lib\json-simple-1.1.1.jar;D:\UCD_Agent\var\plugins\com.urbancode.air.plugin.
otd_1_200b8a9ae7959125d32be1cde7aa20472d2b47c4de3754dbd7627bd6778d33d0\lib\jsoup-1.11.3.jar
;D:\UCD_Agent\var\plugins\com.urbancode.air.plugin.otd_1_200b8a9ae7959125d32be1cde7aa20472d2b4
7c4de3754dbd7627bd6778d33d0\lib\junit-4.10.jar"
D:
\UCD_Agent\var\plugins\com.urbancode.air.plugin.otd_1_200b8a9ae7959125d32be1cde7aa20472d2b47c4
de3754dbd7627bd6778d33d0\OneTestDataGenerationUCD.groovy
D:\UCD_Agent\var\temp\logs-63ec6300-a796-49c2-b3ad-794d0e28b1be\input.props
D:\UCD_Agent\var\temp\logs-63ec6300-a796-49c2-b3ad-794d0e28b1be\output.props"
working directory: D:\UCD_Agent\var\work\UCDComponent

```

```

properties:

  PLUGIN_INPUT_PROPS=D:\UCD_Agent\var\temp\logs-63ec6300-a796-49c2-b3ad-794d0e28b1be\input.pr
ops

  PLUGIN_OUTPUT_PROPS=D:\UCD_Agent\var\temp\logs-63ec6300-a796-49c2-b3ad-794d0e28b1be\output.pr
ops
  connectionName=
  dataFileLocation=C:\Users\user\Downloads
  dataStorage=FILE
  offlineToken=****
  outputFormat=CSV
  project=testing
  records=10
  rootElement=Root:NewType1
  schema=schema
  seedValue=10
  serverUrl=https://otd-build.nonprod.hclnp.com/
environment:
  AGENT_HOME=D:\UCD_Agent
  AH_AUTH_TOKEN=****
  AH_WEB_URL=https://LP1-AP-51837142.PROD.HCLPNP.COM:8443
  AUTH_TOKEN=****
  DS_AUTH_TOKEN=****
  DS_SYSTEM_ENCODING=Cp1252
  JAVA_OPTS=-Dfile.encoding=Cp1252 -Dconsole.encoding=Cp1252

  PLUGIN_HOME=D:\UCD_Agent\var\plugins\com.urbancode.air.plugin.otd_1_200b8a9ae7959125d32be1cde
7aa20472d2b47c4de3754dbd7627bd6778d33d0
  UD_DIALOGUE_ID=63ec6300-a796-49c2-b3ad-794d0e28b1be
  WE_ACTIVITY_ID=172134c9-a5ec-22cd-3c8f-4d3d9789c96d
=====
Data Generation information:
Server URL: https://otd-build.nonprod.hclnp.com/
Project: testing
schema: schema
Root Element: Root:NewType1
Output Format: CSV
Data Location: C:\Users\user\Downloads
Seed Value: 10
Data Storage: FILE
Connection Name:
Include Header: true
No Of Records 10
Validating: server URL , offlineToken
Successfully validated server URL and offlineToken
Status: Test Data Generation Started...
Data Generation Response is [code:200, project_id:4400, schema_id:5ebd2915612cae00fe8e48dd,
  connectionURL:null, dbSchemaName:null, message:Data generation succeeded.,
  data_location:5ebd2906612cae00fe8e48db_4876c2b6-1424-442f-a1f1-5c3cad77576e_biWSum9yW8_1,
  timestamp:2020-05-14T13:08:15.840Z, tableName:null]
Status: Test data generation completed successfully

```

```
Status: Downloading the Generated Test Data
File saved to
C:
\Users\user\Downloads\5ebd2906612cae00fe8e48db_4876c2b6-1424-442f-a1f1-5c3cad77576e_biWSum9yW8
_1.csv location
Status: Completed the download of Generated Test Data
```

The following sample shows the logs of the generated test data when you select `MONGODB` as the data storage type:

```
=====
plugin: HCL OneTest Data UCD Plugin, id: com.urbancode.air.plugin.otd, version: 1
plugin command: 'cmd' '/C' '"D:\UCD_Agent\opt\groovy-1.8.8\bin\groovy.bat -cp
"D:
\UCD_Agent\var\plugins\com.urbancode.air.plugin.otd_1_73a2c474971db8d52b80c0335e047f8f6cc65
6e1784133370b8d253310f6a\classes;D:\UCD_Agent\var\plugins\com.urbancode.air.plugin.otd_1_73a2c
474971db8d52b80c0335e047f8f6cc656e1784133370b8d253310f6a\lib\commons-codec-1.10.jar;D:\UCD_
Agent\var\plugins\com.urbancode.air.plugin.otd_1_73a2c474971db8d52b80c0335e047f8f6cc656e178
4133370b8d253310f6a\lib\commons-logging-1.2.jar;D:\UCD_Agent\var\plugins\com.urbancode.air.plu
gin.otd_1_73a2c474971db8d52b80c0335e047f8f6cc656e1784133370b8d253310f6a\lib\groovy-all-1.8.
4.jar;D:\UCD_Agent\var\plugins\com.urbancode.air.plugin.otd_1_73a2c474971db8d52b80c0335e047
f8f6cc656e1784133370b8d253310f6a\lib\groovy-plugin-utils-1.2.jar;D:\UCD_Agent\var\plugins\com.
urbancode.air.plugin.otd_1_73a2c474971db8d52b80c0335e047f8f6cc656e1784133370b8d253310f6a\
lib\gson-2.8.6.jar;D:\UCD_Agent\var\plugins\com.urbancode.air.plugin.otd_1_73a2c474971db8d52b
80c0335e047f8f6cc656e1784133370b8d253310f6a\lib\hamcrest-core-1.1.jar;D:\UCD_Agent\var\plug
ins\com.urbancode.air.plugin.otd_1_73a2c474971db8d52b80c0335e047f8f6cc656e1784133370b8d2533
10f6a\lib\httpclient-4.5.6.jar;D:\UCD_Agent\var\plugins\com.urbancode.air.plugin.otd_1_73a2c47
4971db8d52b80c0335e047f8f6cc656e1784133370b8d253310f6a\lib\httpcore-4.4.10.jar;D:\UCD_Ag
ent\var\plugins\com.urbancode.air.plugin.otd_1_73a2c474971db8d52b80c0335e047f8f6cc656e17841
33370b8d253310f6a\lib\json-simple-1.1.1.jar;D:\UCD_Agent\var\plugins\com.urbancode.air.plugin.
otd_1_73a2c474971db8d52b80c0335e047f8f6cc656e1784133370b8d253310f6a\lib\jsoup-1.11.3.jar
;D:\UCD_Agent\var\plugins\com.urbancode.air.plugin.otd_1_73a2c474971db8d52b80c0335e047f8f6c
c656e1784133370b8d253310f6a\lib\junit-4.10.jar"
D:
\UCD_Agent\var\plugins\com.urbancode.air.plugin.otd_1_73a2c474971db8d52b80c0335e047f8f6cc65
6e1784133370b8d253310f6a\OneTestDataGenerationUCD.groovy
D:\UCD_Agent\var\temp\logs-48a47aee-3889-4a3b-ba91-298cbc33a99e\input.props
D:\UCD_Agent\var\temp\logs-48a47aee-3889-4a3b-ba91-298cbc33a99e\output.props"
working directory: D:\UCD_Agent\var\work\OTDComponent
properties:

PLUGIN_INPUT_PROPS=D:\UCD_Agent\var\temp\logs-48a47aee-3889-4a3b-ba91-298cbc33a99e\input.pr
ops

PLUGIN_OUTPUT_PROPS=D:\UCD_Agent\var\temp\logs-48a47aee-3889-4a3b-ba91-298cbc33a99e\output.pr
ops
connectionName=TestMongo
dataFileLocation=
dataStorage=MONGODB
offlineToken=****
outputFormat=JSON
project=testproject
records=1
```



```

rootElement=Root:Employee
schema=EmployeeDetails
seedValue=1
serverUrl=https://otd-fvt1.nonprod.hclnp.com/
environment:
AGENT_HOME=D:\UCD_Agent
AH_AUTH_TOKEN=****
AH_WEB_URL=https://LP1-AP-51837142.PROD.HCLPNP.COM:8443
AUTH_TOKEN=****
DS_AUTH_TOKEN=****
DS_SYSTEM_ENCODING=Cp1252
JAVA_OPTS=-Dfile.encoding=Cp1252 -Dconsole.encoding=Cp1252

PLUGIN_HOME=D:\UCD_Agent\var\plugins\com.urbancode.air.plugin.otd_1_73a2c474971db8d52bfdb80c0
335e047f8f6cc656e1784133370b8d253310f6a
UD_DIALOGUE_ID=48a47aee-3889-4a3b-ba91-298cbc33a99e
WE_ACTIVITY_ID=1731397b-b82e-1d8a-dc50-153fd3010ce0
=====
Data Generation information:
Server URL: https://otd-fvt1.nonprod.hclnp.com/
Project: testproject
schema: EmployeeDetails
Root Element: Root:Employee
Output Format: JSON
Data Location:
Seed Value: 1
Data Storage: MONGODB
Connection Name: TestMongo
Include Header: false
No Of Records 1
Validating: server URL , offlineToken
Successfully validated server URL and offlineToken
Status: Test Data Generation Started...
Data Generation Response is [code:200, databaseName:oneTestDataDB, project_id:2850,
schema_id:5efedcb33d358500a1662993, connectionURL:{my-ots}-mongodb:27017, dbSchemaName:null,
message:Data generation succeeded., data_location:null, timestamp:2020-07-03T07:33:11.497Z,
tableName:null, collectionName:employee]
Status: Test data generation completed successfully
Data is written into Mongo database
Connection URL: {my-ots}-mongodb:27017
Database Name: oneTestDataDB
Collection name: employee

```

## Generating the test data by using HCL Launch

When you perform a test while you deploy an application on HCL Launch, you might want to generate test data. You can generate a random test data to test your application by integrating HCL OneTest Data Launch Plugin with HCL® OneTest™ Data.

### Before you begin

You must have completed the following tasks:


- Installed the latest version of the HCL Launch server and agent on the target system.
- Configured the agent and verified that the agent is running on the target system. For information about how to configure the agent, refer to [Configure the agent and target system](#).
- Downloaded the latest version of HCL OneTest Data Launch Plugin from the [HCL License & Delivery portal](#).
- Installed HCL OneTest Data Launch Plugin on the HCL Launch server. See [Installing the HCL OneTest Data Launch Plugin on page 561](#).
- Created an account on the HCL Launch server.
- Logged in to HCL OneTest™ Server.
- Created a project and a schema in HCL® OneTest™ Data.
- Established a connection between HCL® OneTest™ Data and a supported database. See
  - [Establishing a JDBC connection with HCL OneTest Data on page 194](#)
  - [Establishing a MongoDB connection with HCL OneTest Data on page 206](#)



### About this task


You can write the test data generated by integrating HCL OneTest Data Launch Plugin with HCL® OneTest™ Data into a file or any supported database. After integration with HCL OneTest Data Launch Plugin, HCL® OneTest™ Data supports the insertion of the generated test data in both JDBC supported database and MongoDB. To write the generated test data in the database, you must establish a connection between HCL® OneTest™ Data and the supported database.

### Procedure

1. Log in to the HCL Launch server.
2. Create a component.
3. Create a component process, and then set the properties for the component by referring to the following table:

Field	Action	Required/Optional
Name	Enter a name for the HCL Launch application process.	Required
Server URL	Enter the URL of HCL OneTest™ Server.  The format for the URL is as follows: https://<fully-qualified-dns-name>/	Required
Custom Trust Store Path	Enter the path of the custom trust store where the certificate is stored.   <b>Note:</b> If you do not find details of the custom trust store in the application plugin, then the application navigates to the default trust store to access the certificate.	Optional

Field	Action	Required/Optional
	 The default trust store is at the following location: <code>\$JAVA_HOME/jre/lib/security/cacerts</code>	
Custom Trust Store Password	Enter the password to access the custom trust store.	Optional
Offline Token	Enter the offline token that is generated in HCL OneTest™ Server.	Required
Project	Enter the name of your project.	Required
Schema	Enter the name of the schema associated with the project you selected.	Required
Root Element	Specify the root path of the element for which you want to generate the test data.  For example, Root:NewType1	Required
Number of Records	Enter the number of records you want to generate.	Required
Numeric Seed Value	Enter the seed value that acts as an instance of random data when you generate the test data.	Optional
Data Storage	<p>Select the data storage type. The data storage is a location where you want the generated test data to be written.</p> <p>You can select <code>FILE</code>, <code>JDBC</code>, or <code>MONGODB</code> as a data storage type.</p> <p><code>FILE</code>: The generated test data is written into a file and you can download it in your local file system.</p> <p><code>JDBC</code>: The generated test data is written in the selected JDBC supported database.</p> <p><code>MONGODB</code>: The generated test data is written into MongoDB.</p>	Required
Connection Name	 <b>Note:</b> This field is applicable only when you select <code>JDBC</code> or <code>MONGODB</code> as the data storage type.  Enter the name of the <code>JDBC</code> or <code>MONGODB</code> connection.	Required
Output Format	Specify the file format of the generated test data.	Required

Field	Action	Required/Optional
Data File Location	<p>Specify the location for the output file. If the specified location is invalid, by default, the output file is saved in the HCL® OneTest™ Data server.</p> <p> <b>Notes:</b></p> <ul style="list-style-type: none"> <li>You can find the output file in the HCL® OneTest™ Data pod at the following location:</li> </ul> <pre style="margin-left: 40px;">/opt/hcl/hip-rest/output/&lt;accountId&gt;/&lt;userId&gt;/&lt;projectId&gt;/&lt;schemaId&gt;/&lt;genMapPath&gt;</pre> <ul style="list-style-type: none"> <li>This field is not applicable if you select the data storage type as <code>JDBC Of MongoDB</code>.</li> </ul>	Optional



**Note:** You can ignore the following property fields while you set up the integration of HCL Launch with HCL® OneTest™ Data:

- Working Directory
- Post Processing Script
- Precondition
- Use Impersonation

4. Create a resource and select the agent.
5. Add the component that you created to the agent.
6. Create an application.
7. Create an environment from the **Applications** dashboard.
8. Add the resource and the component that you created to the environment.
9. Create a process for the application by clicking the **Processes** tab from the **Applications** dashboard.

The page of the application process is displayed.

10. Click the component process that you created in step 3 from the **Component Process Steps** on the left navigation pane and drag it into the design area.
11. Click **Save**.
12. Go to the **Applications** dashboard, and then click **Request Process** for the environment of the application process that you want to execute.
13. Select the name of the application, environment, and process from the drop-down list, and then click **Next**.
14. Verify that the default settings of the component versions are retained, and then click **Next**.
15. Toggle the **Run Now** switch to the on position to set the **Schedule Deployment**, and then click **Next**.
16. Check the deployment details, and then click **Submit Deployment**.

## Result

You have successfully generated the test data by using the HCL OneTest Data Launch Plugin for HCL® OneTest™ Data.

You can view the completed request process with the status displayed as **Success** or **Failed**.



### Note:

If the test data generation request fails, you can view the logs of the process. See [Viewing the HCL Launch logs on page 561](#).

## What to do next

After the successful completion of the process, you can perform the following tasks:

- If you selected `FILE` as a data storage type, then you can download the generated test data in the local file system at the specified location.
- If you selected `JDBC` as the data storage type, then you can use the generated test data from the database.
- If you selected `MONGODB` as the data storage type, then you can use the generated test data from the database.

## Installing the HCL OneTest Data Launch Plugin

When you want to generate the test data by using the HCL® OneTest™ Data for HCL® OneTest™ Data, you must install the HCL® OneTest™ Data on the HCL Launch server.

1. From **Settings**, click **Automation Plugins**.
2. Click **Load Plugin**.
3. Enter the path of the compressed plug-in file, and then click **Submit**.

## Results

The plug-in is listed on the **Automation Plugins** pane.

## What to do next

After you successfully installed the HCL® OneTest™ Data on the server, you must configure the HCL® OneTest™ Data and generate the test data. See [Generating the test data by using HCL Launch on page 557](#).

## Viewing the HCL Launch logs

After completion of the generation of test data, you can view the details of the process in the HCL Launch console log.

You can click the **Output Log** icon from the console log of the HCL Launch dashboard to view the output log.

The following sample shows the logs of the generated test data when you select `FILE` as the data storage type:

```

=====
plugin: HCL OneTest Data Launch Plugin, id: com.hcllaunch.air.plugin.otd, version: 2
plugin command: 'cmd' '/C' '""D:\HCL Launch\LaunchAgent\opt\groovy-2.4.15\bin\groovy.bat" -cp
"D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_ebae55c6a5da97650f3890760d07778
5e4fa877523edeaf3cfda916b293b4a1\classes;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_ebae55c6a5da97650f3890760d07778
5e4fa877523edeaf3cfda916b293b4a1\lib\commons-codec-1.10.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_ebae55c6a5da97650f3890760d07778
5e4fa877523edeaf3cfda916b293b4a1\lib\commons-logging-1.2.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_ebae55c6a5da97650f3890760d07778
5e4fa877523edeaf3cfda916b293b4a1\lib\groovy-all-1.8.4.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_ebae55c6a5da97650f3890760d07778
5e4fa877523edeaf3cfda916b293b4a1\lib\groovy-plugin-utils-1.2.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_ebae55c6a5da97650f3890760d07778
5e4fa877523edeaf3cfda916b293b4a1\lib\gson-2.8.6.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_ebae55c6a5da97650f3890760d07778
5e4fa877523edeaf3cfda916b293b4a1\lib\hamcrest-core-1.1.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_ebae55c6a5da97650f3890760d07778
5e4fa877523edeaf3cfda916b293b4a1\lib\httpClient-4.5.6.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_ebae55c6a5da97650f3890760d07778
5e4fa877523edeaf3cfda916b293b4a1\lib\httpcore-4.4.10.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_ebae55c6a5da97650f3890760d07778
5e4fa877523edeaf3cfda916b293b4a1\lib\json-simple-1.1.1.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_ebae55c6a5da97650f3890760d07778
5e4fa877523edeaf3cfda916b293b4a1\lib\jsoup-1.11.3.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_ebae55c6a5da97650f3890760d07778
5e4fa877523edeaf3cfda916b293b4a1\lib\junit-4.10.jar" "D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_ebae55c6a5da97650f3890760d07778
5e4fa877523edeaf3cfda916b293b4a1\OneTestDataGenerationHCLLaunch.groovy" "D:\HCL
Launch\LaunchAgent\var\temp\logs-89c29385-395a-4c98-b695-de7eb9dbdee5\input.props" "D:\HCL
Launch\LaunchAgent\var\temp\logs-89c29385-395a-4c98-b695-de7eb9dbdee5\output.props""
working directory: D:\HCL Launch\LaunchAgent\var\work\OTDComponent1
properties:
  PLUGIN_INPUT_PROPS=D:\HCL
Launch\LaunchAgent\var\temp\logs-89c29385-395a-4c98-b695-de7eb9dbdee5\input.props
  PLUGIN_OUTPUT_PROPS=D:\HCL
Launch\LaunchAgent\var\temp\logs-89c29385-395a-4c98-b695-de7eb9dbdee5\output.props
  connectionName=
  dataFileLocation=C:\Users\user1\Downloads
  dataStorage=FILE
  offlineToken=****
  outputFormat=CSV
  project=otdproject
  records=10
  rootElement=Root:NewType1
  schema=otdschema
  seedValue=10
  serverUrl=https://otd-fvt2.nonprod.hclpnp.com/
environment:
  AGENT_HOME=D:\HCL Launch\LaunchAgent
  AH_AUTH_TOKEN=****

```

```

AH_WEB_URL=https://LP1-AP-51837142.PROD.HCLPNP.COM:8443
AUTH_TOKEN=****
DS_AUTH_TOKEN=****
DS_SYSTEM_ENCODING=Cp1252
JAVA_OPTS=-Dfile.encoding=Cp1252 -Dconsole.encoding=Cp1252
PLUGIN_HOME=D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_ebae55c6a5da97650f3890760d07778
5e4fa877523edeadf3cfd916b293b4a1
UD_DIALOGUE_ID=89c29385-395a-4c98-b695-de7eb9dbdee5
WE_ACTIVITY_ID=17443a96-551a-d05a-6041-0d9b77073106
=====
Data Generation information:
Server URL: https://otd-fvt2.nonprod.hclpnp.com/
Project: otdproject
schema: otdschema
Root Element: Root:NewType1
Output Format: CSV
Data Location: C:\Users\user1\Downloads
Seed Value: 10
Data Storage: FILE
Connection Name:
Include Header: true
No Of Records 10
Validating: server URL, offlineToken
Successfully validated server URL and offlineToken
Status: Test Data Generation Started...
Data Generation Response is [code:200, databaseName:null,
project_id:3200, schema_id:5f48ad2b1491d2009db126fd, connectionURL:null,
dbSchemaName:null, message:Data generation succeeded.,
data_location:5f48ad1c1491d2009db126fb_077ae85b-5527-4aba-a64f-5b58d15df899_6dH9eZZfoq_1,
timestamp:2020-08-31T08:36:56.526Z, tableName:null, collectionName:null]
Status: Test data generation completed successfully
Status: Downloading the Generated Test Data
File saved to
C:
\Users\user1\Downloads\5f48ad1c1491d2009db126fb_077ae85b-5527-4aba-a64f-5b58d15df899_6dH9eZZfo
q_1.csv location
Status: Completed the download of Generated Test Data

```

The following sample shows the logs of the generated test data when you select `MONGODB` as the data storage type:

```

=====
plugin: HCL OneTest Data Launch Plugin, id: com.hcllaunch.air.plugin.otd, version: 2
plugin command: 'cmd' '/C' '"D:\HCL Launch\LaunchAgent\opt\groovy-2.4.15\bin\groovy.bat" -cp
"D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_fcffff6fc25d3aad57596e97b05f621
f6a850ad06d6b0495e2954cbf5e1be0e0\classes;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_fcffff6fc25d3aad57596e97b05f621
f6a850ad06d6b0495e2954cbf5e1be0e0\lib\commons-codec-1.10.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_fcffff6fc25d3aad57596e97b05f621
f6a850ad06d6b0495e2954cbf5e1be0e0\lib\commons-logging-1.2.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_fcffff6fc25d3aad57596e97b05f621
f6a850ad06d6b0495e2954cbf5e1be0e0\lib\groovy-all-1.8.4.jar;D:\HCL

```

```

Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_fcffff6fc25d3aad57596e97b05f621
f6a850ad06d6b0495e2954cbf5e1be0e0\lib\groovy-plugin-utils-1.2.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_fcffff6fc25d3aad57596e97b05f621
f6a850ad06d6b0495e2954cbf5e1be0e0\lib\gson-2.8.6.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_fcffff6fc25d3aad57596e97b05f621
f6a850ad06d6b0495e2954cbf5e1be0e0\lib\hamcrest-core-1.1.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_fcffff6fc25d3aad57596e97b05f621
f6a850ad06d6b0495e2954cbf5e1be0e0\lib\httpClient-4.5.6.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_fcffff6fc25d3aad57596e97b05f621
f6a850ad06d6b0495e2954cbf5e1be0e0\lib\httpcore-4.4.10.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_fcffff6fc25d3aad57596e97b05f621
f6a850ad06d6b0495e2954cbf5e1be0e0\lib\json-simple-1.1.1.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_fcffff6fc25d3aad57596e97b05f621
f6a850ad06d6b0495e2954cbf5e1be0e0\lib\jsoup-1.11.3.jar;D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_fcffff6fc25d3aad57596e97b05f621
f6a850ad06d6b0495e2954cbf5e1be0e0\lib\junit-4.10.jar" "D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_fcffff6fc25d3aad57596e97b05f621
f6a850ad06d6b0495e2954cbf5e1be0e0\OneTestDataGenerationHCLLaunch.groovy" "D:\HCL
Launch\LaunchAgent\var\temp\logs-49ec0c81-7f25-4494-accd-4bf9bd384d93\input.props" "D:\HCL
Launch\LaunchAgent\var\temp\logs-49ec0c81-7f25-4494-accd-4bf9bd384d93\output.props""
working directory: D:\HCL Launch\LaunchAgent\var\work\OTDComponent1
properties:
  PLUGIN_INPUT_PROPS=D:\HCL
Launch\LaunchAgent\var\temp\logs-49ec0c81-7f25-4494-accd-4bf9bd384d93\input.props
  PLUGIN_OUTPUT_PROPS=D:\HCL
Launch\LaunchAgent\var\temp\logs-49ec0c81-7f25-4494-accd-4bf9bd384d93\output.props
connectionName=MongoConnection
dataFileLocation=
dataStorage=MONGODB
offlineToken=****
outputFormat=JSON
project=otdproject
records=1
rootElement=Root:NewType1
schema=otdschema
seedValue=
serverUrl=https://otd-fvt1.nonprod.hclnp.com/
environment:
AGENT_HOME=D:\HCL Launch\LaunchAgent
AH_AUTH_TOKEN=****
AH_WEB_URL=https://LP1-AP-51837142.PROD.HCLPNP.COM:8443
AUTH_TOKEN=****
DS_AUTH_TOKEN=****
DS_SYSTEM_ENCODING=Cp1252
JAVA_OPTS=-Dfile.encoding=Cp1252 -Dconsole.encoding=Cp1252
PLUGIN_HOME=D:\HCL
Launch\LaunchAgent\var\plugins\com.hcllaunch.air.plugin.otd_2_fcffff6fc25d3aad57596e97b05f621
f6a850ad06d6b0495e2954cbf5e1be0e0
  UD_DIALOGUE_ID=49ec0c81-7f25-4494-accd-4bf9bd384d93
  WE_ACTIVITY_ID=174d8e29-f157-0a93-07a4-5a90df84fb41
=====
Data Generation information:

```



```

Server URL: https://otd-fvt1.nonprod.hclpnp.com/
Project: otdproject
schema: otdschema
Root Element: Root:NewType1
Output Format: JSON
Data Location:
Seed Value:
Data Storage: MONGODB
Connection Name: MongoConnection
Include Header: false
No Of Records 1
Validating: server URL , offlineToken
Successfully validated server URL and offlineToken
Status: Test Data Generation Started...
Data Generation Response is [code:200, databaseName:mdb, project_id:1150,
  schema_id:5f72c22171ff1a00a009ae50, connectionURL:{my-ots}-mongodb:27017, message:Data
  generation succeeded., timestamp:2020-09-29T08:02:59.141Z, collectionName:users]
Status: Test data generation completed successfully
Data is written into Mongo database
Connection URL: {my-ots}-mongodb:27017
Database Name: mdb
Collection name: users

```

## Managing access to HCL OneTest™ Server

Desktop clients and third-party integrations use offline user tokens to connect to HCL OneTest™ Server.

You can configure the desktop clients and third-party integrations to access HCL OneTest™ Server by using the offline user tokens created from the server.

Offline user tokens are used in the following cases:

- To retrieve secrets configured on the server to be used in certain tests in HCL OneTest™ API.
- To enable publishing of test results and reports from desktop clients to HCL OneTest™ Server.
- To enable the Resource Monitoring Service in HCL OneTest™ Performance to monitor a data source during a run on HCL OneTest™ Server.
- To enable agents to connect with HCL OneTest™ Server.
- To enable HTTP proxies to connect with HCL OneTest™ Server.
- To enable third-party integrations.


### About this task

From HCL OneTest™ Server, you can create an offline user token, copy it, and then use that token to enable connections to be authenticated by HCL OneTest™ Server.

You can also delete all the tokens, when you no longer need access or you want to prevent access to the server.

Creating a token

- Follow these steps:

1. Click the **User** icon  from the menu bar, and then select **Create Token**.

The Create Offline User Token dialog box is displayed.



**Note:** This token is not accessible after you copy it as the dialog box closes.

2. Copy the token and paste it into a private location for ongoing reference.

The dialog box closes.

3. Paste the token into the desktop client UI or in the application where this token is required.


See the following table for a list of where you can use the tokens and where you can find more information.

**Table 9. Using the offline user tokens to access HCL OneTest™ Server**

Use the offline user token in the following places	More information
HCL OneTest™ API	See <a href="#">HCL OneTest™ API and HCL OneTest™ Server</a> .
HCL OneTest™ Performance	See <a href="#">Publishing test results to the server</a> . See <a href="#">Enabling Enablement of Resource Monitoring services for a schedule</a> .
HCL OneTest™ UI	See <a href="#">Publishing test results to the server</a> .
Third-party integrations	See <a href="#">Integrating with other applications on page 486</a> .

### Deleting tokens

- Follow these steps:

1. Click the **User** icon  from the menu bar and select **Delete Tokens**.

The Delete All Offline User Tokens dialog box is displayed.

2. Inform the desktop client users that the tokens were deleted. Fix any test automation scripts that used tokens.

# Chapter 7. Test Manager Guide

This guide describes how to manage and track your overall test effort. This guide is intended for test managers.

---

Related information

[Managing access to server projects on page 588](#)

## Team space management

You might want to become a team space member, add members to your team space, change a user role in a team space, modify the team space, or delete a team space.

Each team space is associated with at least one *Team Space Owner*. You can assign multiple owners to your team space. Any user who becomes a member of a team space and is assigned the role of a *Member*, *Project Creator*, *Architect*, or *Team Space Owner* of a team space, can access the team space.

As an owner of a team space, after you create a team space, you must configure the team space license. You can then add members to the team space and can assign different roles to each member of the team space. You can update or delete a team space. You can also add a repository to a team space to store a system model and edit the repository settings. Apart from adding a repository, you can add, update, or delete the Resource Monitoring agent if no longer required.

As an *Architect* of a team space, you can manage a team space repository and design a system model.

As a *Project Creator* of a team space, you can create a project in the team space.

As a licensed user who is assigned with a *Member*, *Project Creator*, *Architect*, or *Team Space Owner* role of a team space, you can add a Docker host to your team space. You can view the configured Docker, agents, or intercepts of your team space. You can edit only those Docker hosts that you created.

## Viewing team spaces

After you log in to HCL OneTest™ Server for the first time, you can view all the existing team spaces on the **Team Space Dashboard** page.

### About this task

As a member of a team space, you can view your team spaces under **My Team Spaces**. You can also view the team spaces that are created by other users as the public team space and the Permissive team spaces under **Other Team Spaces**.

If the initial team space does not exist, then you can view the **Team Space Dashboard** without any navigation pane.

You as a member of a team space, when you want to use that team space immediately after you log in to HCL OneTest™ Server, you can suffix the alias in the application URL.

For example, if the alias of the team space is `first`, then you must use the following HCL OneTest™ Server URL in the address bar:

```
<server url>/#/first
```

Log in to HCL OneTest™ Server.

### Result

The **Projects** page of the initial team space is displayed.

You can view the team spaces in the following panels:

- **My Team Spaces:** You can view all the team spaces that you created. You can also view the team spaces in which you are a member.
- **Other Team Spaces:** You can view all the team spaces that are created by others.



**Note:** You cannot view the Restrictive private type team spaces.

### Results

You have viewed the team spaces that are created by you and other users on the **Team Space Dashboard**.

### What to do next

You can select a team space and can send a request to become a member of that team space. See [Becoming a team space member on page 575](#).

## Adding members to a team space

When you create a team space to manage projects, you must add users to your team space so that they can create and access the projects.

### Before you begin

- As an administrator, you are assigned the default role of a *Team Space Owner*. If you log in to HCL OneTest™ Server as another user, then you must have been assigned the role of a *Team Space Owner*.
- As an owner of a team space, you must have completed the following tasks:
  - Created a team space. See [Creating a team space on page 131](#).
  - Configured a license for the team space. See [Configuring licenses for team spaces on page 140](#).

You must have been assigned a role as a *Team Space Owner*. If you are a server administrator, then you are assigned with the default role of a *Team Space Owner*.

### About this task

As an owner of a team space, you can add multiple users to your **Permissive** or **Restrictive** team space.

1. Log in to HCL OneTest™ Server.

**Result**

The **Projects** page of the initial team space is displayed.

2. Click **Dashboard**.

**Result**

The **Team Space Dashboard** page of the initial team space is displayed.



**Note:** If the initial team space does not exist, then you can view the **Team Space Dashboard** page without any navigation pane.

3. Identify the team space for which you want to add the members.

4. Click the **Settings** icon .

Alternatively, click **Manage > Configuration** in the navigation pane.

**Result**

The **Team Space Configuration** page is displayed.

5. Click the **Details** tab.

6. Enter a user name in the **Add People** field on the to add users.

Alternatively, you can find the user name by using type search.

**Result**

The list of users is displayed.

7. Select the user.

8. Assign a role to the selected user.




**Note:** All members are assigned the default roles of a *Project Creator* and *Architect*.

You can choose the role for the user based on its associated actions:

Roles	Responsibilities
<i>Team Space Owner</i>	<ul style="list-style-type: none"> <li>◦ Create a team space.</li> <li>◦ Update and delete a team space.</li> <li>◦ Add members to a team space.</li> <li>◦ Remove members from the team space.</li> <li>◦ Assign role to the members.</li> <li>◦ Modify roles of the members.</li> <li>◦ Create and modify a project.</li> <li>◦ Add or update the system model.</li> <li>◦ Add a repository to store a system model.</li> <li>◦ Update and delete a repository in a team space.</li> <li>◦ Add, update, or delete Docker.</li> </ul>

Roles	Responsibilities
	<ul style="list-style-type: none"> <li>◦ Configure or update agent and intercepts.</li> <li>◦ Register, update, delete, and list Resource Monitoring sources in a team space.</li> </ul>
<i>Project Creator</i>	<ul style="list-style-type: none"> <li>◦ Create, update, and view a project</li> <li>◦ Add, update, or delete Docker.</li> <li>◦ Configure or update agent and intercepts.</li> </ul>
<i>Architect</i>	<ul style="list-style-type: none"> <li>◦ Create, update, delete, and view a repository in a team space.</li> <li>◦ Create, update, and delete components in the system model.</li> <li>◦ Add, update, or delete Docker.</li> <li>◦ Configure or update agent and intercepts.</li> </ul>
<i>Member</i>	<ul style="list-style-type: none"> <li>◦ View the list of team spaces in the <b>Team Space Dashboard</b>.</li> <li>◦ View the configuration of the team space.</li> <li>◦ Add, update, or delete Docker.</li> <li>◦ Configure or update agent and intercepts.</li> <li>◦ View the registered Dockers, agents, or intercepts.</li> <li>◦ View the Resource Monitoring agents in a team space.</li> <li>◦ View the registered Resource Monitoring agents.</li> <li>◦ View components in the system model.</li> </ul>

**Notes:**

- The members with the assigned role of a *Team Space Owner*, *Project Creator*, *Architect*, or *Member* of a team space can be assigned with any of the following project roles:
  - *Project Owner*
  - *Tester*
  - *Viewer*
- You can add or remove the role of a member by using the **Menu** icon  next to each user in the **Members** list.

9. Repeat the applicable steps to add another user.

**Results**

You have added members with assigned roles to your team space. You can view the list of all added users on the **Team Space Configuration** page.

**What to do next**

You can reassign or remove the roles of users to one or more team spaces that you own. See [Managing members and their roles in a team space on page 576](#).

All users can request to be a member of another team space. See [Becoming a team space member on page 575](#).

Users added as members to the team space can view the team space in **My Team Spaces**.

## Modifying the configuration of a team space

After you create a team space, if you want to modify the details of the team space, add a repository to your team space, or delete a team space, then you can perform these tasks on the **Team Space Configuration** page.

### Before you begin

You must have been assigned the role of a *Team Space Owner*.

### About this task

The **Team Space Configuration** page displays details of the team space.

As an owner of a team space, you can modify details of the team space, add members to your team space, and assign roles to the members of the team space on the **Team Space Configuration** page.

The *Team Space Owner* and *Architect* can add or remove a repository from the team space.

If the team space is no longer required, then the *Team Space Owner* can also delete that team space.

As a *Member*, *Project Creator*, or *Architect*, you can only view the configuration details of a team space on the **Details** tab of the **Team Space Configuration** page.

As a *Member*, and *Project Creator* you can only view the configuration details of a team space on the **Repository** tab of the **Team Space Configuration** page.

1. Log in to HCL OneTest™ Server.

#### Result

The **Projects** page of the initial team space is displayed.

2. Click **Dashboard**.

#### Result

The **Team Space Dashboard** page is displayed.

3. Select the team space that you want to modify from **My Team Spaces**.


4. Click the **Settings** icon  of the selected team space.

#### Result

The **Team Space Configuration** page is displayed.

5. Perform the modifications on the following details of the **Details** tabs of the team space:

Fields	More information
Name	Name of the team space.

Fields	More information
URL alias	An alias of the team space. Suffix the alias in the application URL to access a specific team space after you log into the application.
Available at	Displays the application URL by using an alias. You can copy this URL and use it when you want to access the team space immediately after you log in to the application.
Member Access	<p>Access to a team space based on the type of the team space. The team space can be of the following types:</p> <ul style="list-style-type: none"> <li>◦ <b>Permissive:</b> All licensed users can access this team space without the approvals of a <i>Team Space Owner</i>.</li> <li>◦ <b>Restrictive:</b> All licensed users must get approvals from the <i>Team Space Owner</i> to access this team space.</li> </ul>
Visibility	<p>The visibility mode of a team space can be of the following types:</p> <ul style="list-style-type: none"> <li>◦ <b>Public:</b> The team spaces are visible to all licensed users in the <b>Other Team Spaces</b> panel.</li> <li>◦ <b>Private:</b> The team spaces are not visible to all the licensed users in the <b>Other Team Spaces</b> panel.</li> </ul> <p> <b>Note:</b> The visibility mode is available only when the <b>Restrictive</b> type of team space is selected.</p>
Description	Brief description of the team space. This is an optional field.
Add People	List of members in a team space.

6. Click the **Repository** tab to modify the details of the repository of the team space.

### Results

You have modified the configuration details of the selected team space.

### What to do next



You can add members and assign roles to each member of your team space.

Related information

[Adding members to a team space on page 568](#)

[Managing members and their roles in a team space on page 576](#)

## Viewing the configuration of a team space

You can view the configuration of a team space as a member of any team space. You can view the configuration details on the **Team Space Configuration** page.

### Before you begin

You must be a member of a team space.

### About this task


The **Team Space Configuration** page displays the configuration details of the team space.

As a member of a team space, if you want to access your team space immediately after logging into HCL OneTest™ Server, then you must suffix the alias in the application URL. You can find details of the alias on the **Team Space Configuration** page. You can also view the list of members and their roles in that team space on the **Team Space Configuration** page.

1. Log in to HCL OneTest™ Server.
2. Click **Dashboard**.

#### Result


The **Team Space Dashboard** page is displayed.


3. Identify the team space for which you want to view the configuration from **My Team Spaces**.
4. Click the **Settings** icon  of the team space.

The **Team Space Configuration** page is displayed.

You can view the following details about the team space:

Fields	Tabs	More information
Name	<b>Details</b>	Displays the name of the team space.
URL alias	<b>Details</b>	Displays an alias of the team space. You must suffix this alias in the application URL to access a specific team space after you log in to the application.

Fields	Tabs	More information
Available at	<b>Details</b>	Displays the application URL by using an alias. You can copy this URL and use it when you want to access the team space immediately after you log in to the application.
Member Access	<b>Details</b>	<p>Displays the type of team space that defines the permissions to access the team space. A team space can be of the following types:</p> <ul style="list-style-type: none"> <li>◦ <b>Permissive:</b> All licensed users can access this team space without the approvals of a <i>Team Space Owner</i>.</li> <li>◦ <b>Restrictive:</b> All licensed users must get approvals from the <i>Team Space Owner</i> to access this team space.</li> </ul>
Visibility	<b>Details</b>	<p>The visibility mode of a team space can be of the following types:</p> <ul style="list-style-type: none"> <li>◦ <b>Public:</b> The team spaces are visible to all licensed users in the <b>Other Team Spaces</b> panel.</li> <li>◦ <b>Private:</b> The team spaces are not visible to all the licensed users in the <b>Other Team Spaces</b> panel.</li> </ul> <p> <b>Note:</b> The visibility mode is available only when the</p>

Fields	Tabs	More information
		 <b>Restrictive</b> type of team space is selected.
Members	<b>Details</b>	Displays a list of members of the team space.
Repository	<b>Repository</b>	Displays the repository details that host the system model.

### Results

You have viewed the configuration details of the selected team space.

## Becoming a team space member

After logging into HCL OneTest™ Server, the licensed users must be a member of any team space to perform test activities in a project. As a licensed user, if you are not assigned the role of a member in any team space, then you can request the *Team Space Owner* to become a member of any existing team space.

### Before you begin

You must be a licensed user.

### About this task

If you want to become a member of a **Restrictive** team space, then only the owner of that team space can grant you the role of a member of a team space. If you request to become a member of a **Permissive** team space, then you automatically become a member of that team space.



**Note:** The server administrator can join any team space automatically without any approval and is assigned with a default role of *Team Space Owner*.

1. Log in to HCL OneTest™ Server.

#### Result

The **Projects** page of the initial team space is displayed.

2. Click **Dashboard**.

#### Result

The **Team Space Dashboard** page is displayed.


3. Select the team space that you want to join from the list of other team spaces in the **Team Space Dashboard**.



**Note:**



You can view only the public type of **Restrictive** team spaces and the **Permissive** team spaces.

4. Click the **Key** icon  of the selected team space.
5. Confirm your request when prompted.

If you agree, then the *Team Space Owner* sees a notification about the request.

After opening the notification, the *Team Space Owner* can view the user who is requesting the access and can accept or decline the request. If the *Team Space Owner* accepts your request, you are added and assigned a *Member* role.



**Notes:**

- You can become a member of any team space only when the license of the selected team space is configured.
- If you are a server administrator, then the *Team Space Owner* role is the default role assigned.
- If you are a licensed user, then the *Member* role is the default role assigned and is restricted to specific actions.

You can then see the team space under **My Team Spaces** on the **Team Space Dashboard** page.

You might want to follow up with the *Team Space Owner* if your request is pending for some time.

## Results

You have become a member of another team space.

## What to do next

You can view the actions that you can perform based on your assigned role. See [Adding members to a team space on page 568](#).

## Managing members and their roles in a team space

After you assign roles to the members of a team space, as a *Team Space Owner*, you might want to remove a member from the team space or reassign the role to the existing members of your team space.

## Before you begin

You must have been assigned the role of a *Team Space Owner*.

## About this task

As a licensed user, you are assigned a role of a *Member* in any team space after the approval of the *Team Space Owner*.

As an owner of a team space, you can remove or reassign any member of your team space with the roles of a *Team Space Owner*, *Project Creator*, or *Architect*. You can also remove any member from the list of members of the team space.

1. Log in to HCL OneTest™ Server.

**Result**

The **Projects** page of the initial team space is displayed.

2. Click **Dashboard**.

**Result**

The **Team Space Dashboard** page is displayed.

3. Identify the team space for which you want to reassign the roles of the members or remove a member from the list of members.

4. Click the **Settings** icon  of the identified team space.

**Result**

The **Details** tab of the **Team Space Configuration** page is displayed.

5. Identify the member from the list of **Members**.

6. Click the **Menu** icon  next to the selected member.

7. Perform any one of the following operations to the identified member of the team space:

**Choose from:**

- Click **Remove Member** to remove the identified member from the team space.
- Select the role that you want to assign or remove the assigned role for the identified member.

You can choose multiple roles for the members based on the associated actions:

Roles	Actions
<i>Team Space Owner</i>	<ul style="list-style-type: none"> <li>▪ Create a team space.</li> <li>▪ Update and delete a team space.</li> <li>▪ Add members to a team space.</li> <li>▪ Remove members from the team space.</li> <li>▪ Assign role to the members.</li> <li>▪ Modify roles of the members.</li> <li>▪ Create and modify a project.</li> <li>▪ Add or update the system model.</li> <li>▪ Add a repository to store a system model.</li> <li>▪ Update and delete a repository in a team space.</li> <li>▪ Add, update, or delete Docker.</li> <li>▪ Configure or update agent and intercepts.</li> <li>▪ Register, update, delete, and list Resource Monitoring sources in a team space.</li> </ul>

Roles	Actions
<i>Project Creator</i>	<ul style="list-style-type: none"> <li>▪ Create, update, and view a project.</li> <li>▪ Add, update, or delete Docker.</li> <li>▪ Configure or update agent and intercepts.</li> </ul>
<i>Architect</i>	<ul style="list-style-type: none"> <li>▪ Create, update, delete, and view a repository in a team space.</li> <li>▪ Create, update, and delete components in the system model.</li> <li>▪ Add, update, or delete Docker.</li> <li>▪ Configure or update agent and intercepts.</li> </ul>
<i>Member</i>	<ul style="list-style-type: none"> <li>▪ View the list of team spaces in the <b>Team Space Dashboard</b>.</li> <li>▪ View the configuration of the team space.</li> <li>▪ Add, update, or delete Docker.</li> <li>▪ Configure or update agent and intercepts.</li> <li>▪ View the registered Dockers, agents, or intercepts.</li> <li>▪ View the Resource Monitoring agents in a team space.</li> <li>▪ View the registered Resource Monitoring agents.</li> <li>▪ View components in the system model.</li> </ul>



**Note:** The *Team Space Owner*, *Project Creator*, *Architect*, or *Member* of a team space can be assigned with any of the following project roles:

- *Project Owner*
- *Tester*
- *Viewer*

Similarly, you can remove the assigned role of any member of your team space.

## Results

You have completed any of the following tasks in your team space:

- Reassigned the role of the members.
- Removed the assigned role of the members.
- Removed the members.

---

Related information

[Adding members to a team space on page 568](#)

## Deleting a team space

You might want to delete a team space when it is no longer needed.

### Before you begin

You must have been assigned the role of a *Team Space Owner* of HCL OneTest™ Server.

### About this task

As a *Team Space Owner*, you can delete a team space that you do not need. When you delete a team space, it is a permanent deletion. You cannot undo the delete action on a team space.

When you delete the initial team space, you can only view the **Team Space Dashboard** page without any navigation pane.

After you delete the team space, you can use the name of the team space again.

1. Log in to HCL OneTest™ Server.

#### Result

The **Projects** page of the initial team space is displayed.

2. Click **Dashboard**.

#### Result

The **Team Space Dashboard** page is displayed.

3. Click **Manage > Configuration**.

#### Result

The **Team Space Configuration** page is displayed.

4. Click the **Details** tab.
5. Click **Delete Team Space**.
6. Confirm that you want to delete the team space when prompted.

If you proceed, the team space is deleted including all of its projects and members.

### Results

You have deleted the selected team space.

### What to do next

You can view the available team spaces on the **Team Space Dashboard** page. See [Viewing team spaces on page 567](#).

After you delete a team space, the license that is used for that team space is released and is available to configure the other team spaces, if required. See [Configuring licenses for team spaces on page 140](#).

## Test assets and a server project

HCL OneTest™ Server projects manage access to your test assets, which are stored in a Git repository. Projects are maintained in a team space. Projects are either public by default or private. Private projects are not discoverable by other users. You can either add your own project or you can request to be a member of another public project.

All the licensed users are assigned with a default role of a *Project Creator*. As a *Project Creator*, when you create a project immediately after logging in the application, the project is created in the initial team space. If you want to create a project in any specific team space, then you must become a member of that team space. When you add your own project, you can configure it now or later. Configuring the project includes adding details about the project, adding one or more Git repositories, optionally adding secrets, classifying encrypted datasets, and adding a change management system. To run test assets that are associated with an encrypted dataset, you must categorize the encrypted datasets by creating a classification. You can also add users to your project to access your test assets only if the users are the members of that team space. To use Jira to create defects for the tests available in your project, you must configure the Jira server for your project on HCL OneTest™ Server.

---

### Related information

[Team space overview on page 128](#)

[Becoming a team space member on page 575](#)

[Becoming a project member on page 587](#)

[Managing access to server projects on page 588](#)

[Managing repositories on page 592](#)

[Managing an encrypted dataset on page 605](#)

[Default user administration on page 105](#)

[Configuration of Jira as a change management system on page 480](#)

## Working with projects in a team space

After you log in to HCL OneTest™ Server, you must have a project to perform the test activities. To work with projects, you must create and configure projects.

### Before you begin

You must have logged into HCL OneTest™ Server.

### About this task

You must be assigned with the role of an *Owner*, *Tester*, or *Viewer* of a project.

1. Go to the **Projects** page from the **Initial Team Space** Home page. See [Viewing projects on page 586](#).
2. Add a project. See [Adding a project on page 582](#).
3. Add a repository. See [Adding repositories to a server project on page 583](#).



4. Add secrets in secrets collections. See [Secrets configuration on page 585](#).
5. Encrypt the data set resources. See [Data security](#).
6. Configure a change management system. See [Change management system](#).
7. Add members to a project and configure their roles. See [Adding users to a project on page 585](#).
8. View the test assets or resources maintained in the repositories. See [Selecting the global branch in a project on page 597](#)
9. Perform the following tasks based on the type of test assets or resources that you want to run:
  - a. Add agents to your project, if you are using remote agents as a location to run certain tests. See [Adding an agent to a project](#).
  - b. Add remote Docker, if you are using a remote Docker host as a location to run tests. See [Adding a remote Docker host to the project for running tests on page 299](#).
  - c. Read the considerations that you must take into account before you configure a run or start a virtual service. See [Prerequisites for running virtual services on page 392](#).
  - d. Read the considerations that you must take into account before you configure a test run. See [Prerequisites to running tests on page 271](#).
10. Configure a run of the test asset or resource. See [Test run configurations on page 305](#).
11. View the progress of a test run. See [Viewing the progress of running test assets on page 381](#).
12. Manage a running test. See [Management of running tests on page 378](#).
13. View the results of a test run after the run is completed. See [Test results on page 435](#).
14. Review the test results and create defects for test runs from the Results page. See [Creating Jira defects on page 484](#).

## Results

You have performed all the tasks with the projects.

## Repository considerations for a server project

To collaborate with other project stakeholders, you can open test assets from a local clone of the Git repository, pull project test assets from a Git repository, and push changes made to your local test assets to a Git repository. Before you add a project and add a repository to that project, you must consider some information about repositories.

Consider the following sections about using Git repositories with HCL OneTest™ Server. For more information about installing, setting up, and using Git, see the [Git documentation](#).

## Git

You must install Git or upgrade the version if you already have installed Git.

## Repositories and user identities

After you install Git, you must set up your Git repository and set up access for members. You must ensure that the repository contains your test assets.

Optionally, you can use a command line utility or Git tool to access the repository, upload your test assets, fetch or pull from the repository, push to the repository, clone the repository, and other operations you want to perform in Git.

## Local and shared repositories

After you create a remote or shared repository in Git, you can create a local version of the repository by cloning the remote repository. You must ensure that your test assets are available in the remote repository and are also cloned to the local repository.

Alternatively, if your test assets are on your local system, you can set up a Git repository in the bare mode, add the project files to the local repository, and then commit and push from the local repository to the remote repository in Git by using your preferred method.



**Note:** While copying the test assets from your local system to the repository, you must ensure that you copy the entire project that contains the test assets.

## User authentication for the Git repository

The administrator can set up different types of authentication for accessing the Git repository. HCL OneTest™ Server supports the following authentication types:

- HTTP with user name and password
- HTTP without user name and password
- HTTPS with user name and password
- HTTPS without user name and password
- SSH with SSH key and passphrase
- SSH with SSH key and without a passphrase

Based on the authentication type that is set for a repository, you must provide the same authentication values in HCL OneTest™ Server when you add a repository.

## Test assets

You must complete the following tasks in the desktop client where you are authoring your test before you check in and commit the test assets to the Git repository.

Test type	Task	More information
API suite in HCL OneTest™ API	Change the local stub to a remote stub.	See <a href="#">Test run considerations for API Suites on page 273</a> .
	Add the library files.	

## Adding a project

The first step is to add a project and provide some details about it.

### Before you begin

To add one or more projects to manage access to your test assets, you must log in to HCL OneTest™ Server by providing the application URL in a browser.

If you are a new user, and LDAP and Active Directory are not configured, then you must first sign up by completing a form that specifies user information such as an email, user name, and password. You can then log in by using that information.

You must be a member of any licensed team space.

### About this task

As a licensed user, you are by default assigned as both *Project Creator* and *Architect* role for the initial team space. You must select a team space to create a project.

You can view all the team spaces in the **Team Space Dashboard**. To create a project in any team space, you must be a member of that team space and assigned a *Team Space Owner* or *Project Creator* role in that team space.

After you select a team space from the **Team Space Dashboard**, you can give a name and description to add a project.

1. Log in to HCL OneTest™ Server.

#### Result

The **Projects** page of the initial team space is displayed.

2. Click **New Project** if you want to create a project in the initial team space. Otherwise, click **Dashboard** to select the team space, and then create a project.

The **Details** page is displayed.

3. Decide if you want a public project, which is the default, or a private project.

### Results

You have successfully created a project.

### What to do next

You can add a repository to your project.

---

Related information

[Default user administration on page 105](#)

## Adding repositories to a server project

You can add repositories to a server project to access assets and resources available in the respective repository.

### Before you begin

You must have completed the following tasks:

- Added a project on HCL OneTest™ Server.
- Been granted permission to access the repository.

### About this task

As a project owner or a tester, you can add one or more repositories to your project. When you add a repository, the Git repository is cloned to your project. While adding a repository, you must provide the necessary authentication credentials that are set for the Git repository that you want to add to your project. For example, if the authentication type is SSH, then you must provide the Git URL, a deploy key, and a passphrase.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project, and then click **Manage**.
3. Click the **Repositories** tab.
4. Click **Add Repository**.

The **Add Repository** dialog box is displayed.

5. Enter the URL of the Git repository that you want to add to your project in the **Git Repository** field.
6. Enter the required credentials based on any of the following authentication methods configured in the repository.

To gain access to the repository, you must use any one of the authentication methods:

Authentication method	Credentials required
SSH	<ul style="list-style-type: none"> <li>◦ Deploy key</li> <li>◦ Passphrase</li> </ul>
HTTPS	<ul style="list-style-type: none"> <li>◦ User name</li> <li>◦ Password</li> </ul>
HTTP	<ul style="list-style-type: none"> <li>◦ User name</li> <li>◦ Password</li> </ul>



#### Notes:

- You must have defined the authentication type and set the authentication credentials in the Git repository.
- If you use SSH to connect to your remote repository and HCL OneTest™ Server displays an `Auth Fail` exception while using the deliver changes option, you can resolve this exception error by regenerating your SSH keys by using the `-m PEM` option.

7. Click **Add**.

The Git repository is added to your project in the team space.



**Note:** Depending on the size of the repository that you are cloning, it can take a few to several minutes to clone the repository.

8. **Optional:** Repeat the steps to add another repository.

### What to do next

You can perform the following actions on the repository that you added:

- Update the authentication credentials if they are changed in the Git repository configuration.
- Delete a repository if it is no longer required.
- Refresh a repository to fetch and synchronize changes from the remote repository.
- Configure a webhook to notify the server when there is a push event in the remote repository.

---

Related information

[Managing repositories on page 592](#)

## Secrets configuration

If you are working with an API suite and the project test asset contains environment variables that are required for the test runs, you must configure the environment variables as secrets in a secrets collection by using the **Secrets** tab. Configuring secrets enable the API suite to run correctly on HCL OneTest™ Server.

---

Related information

[Protecting API test assets by using secrets on page 598](#)

## Adding users to a project

You can add users to your server project so they can access your test assets.

### Before you begin

- You own a project.
- The users you want to add to your project must be the members of the team space of that project.

### About this task


As a project owner, you can add other users to your public or private project. For example, if you have a large number of test assets that must be run, you might want your test team to help run them, and see the results.

1. Log in to HCL OneTest™ Server.

#### Result

The **Projects** page of the Initial Team Space is displayed.

2. Select the project for which you want to add the users.

3. Click the **Settings** icon  on the Project card.
4. Add users by entering the user name of the user you want to add. You can add a partial name and then press the **Enter** key to see the user that you want to add.
5. Select that user and then assign a role.
6. Click **Viewer**, **Tester**, or **Owner**. When you add a user, the default role is *Viewer*.

The added user after logging in can see the owner's project in their list of **My Projects**.

7. Repeat the applicable steps to add another user.

## Results

You have successfully added users to your project.

## What to do next

You can add or remove roles to one or more projects that you own. All users can request to be a member of another public project.

---

Related information

[Becoming a project member on page 587](#)

[Managing access to server projects on page 588](#)

## Viewing projects

After you log in to HCL OneTest™ Server, you can view the existing projects in the **Initial Team Space**.

### About this task

As a project owner, you can view your projects under the **My Projects** panel, and the projects that are created by other users as public projects are displayed under the **Other Projects** panel.

Log in to HCL OneTest™ Server.

The **Projects** page of the **Initial Team Space** is displayed.

You can view the following projects:

- The projects that you created or were added to as a member with either the *Tester* or *Viewer* role are displayed under the **My Projects** panel.
- The projects that are created by others as public projects are displayed under the **Other Projects** panel.

## Results

You have viewed the projects that are created by you and other users in the team space.

## What to do next

You can perform any of the following tasks:

- Create a project. See [Adding details to a server project on page 582](#).
- Request to become a member of a project that is displayed under the **Other Projects** panel. See [Becoming a project member on page 587](#).
- View the other pages in the team space and perform tasks on those pages. See [Tasks in a team space on page 132](#).

## Becoming a project member


You might want to request to be a member of another project. New users without any projects might also want to be a member of an existing project.

### About this task



As a project owner, you can add users to your public or private projects. All users can request to become a member of a public project.




**Note:** You can add a licensed user as a member of your project only if that user is the member of your team space.

1. Request to be a member of another project. Search for the project that you want to join in the list of other projects from the project Home page. Click that Project card or the **Key** icon . Only public projects are visible in the list of projects.
2. Confirm your request when prompted. If you agree, the project owner is notified. The project owner sees a pending request **Notification** icon on their Project card.

### My Projects

 myproj


No description.

Owner: [user55](#) 

Created: [a month ago](#)

Branch: **master**

Test Suites: **59**    Individual Tests: **40**

Last Run Time: Nov 4, 2020 8:59 AM

from beginning

1	0/59	1/59
In Progress	Fail	Success

After opening the notification, the owner can see the user that is requesting access and can accept or decline it.

If the project owner accepts your request, you are added as a member of the project with a Viewer role. A Viewer role is the default role assigned and is restricted to specific actions.

You then see the project under My Projects on the Home page.

If the project owner declines your request, follow up with the project owner.

**Results**

You have successfully sent a request to become a member of another project.

**Managing access to server projects**

You might want to add or remove a project member role from your project.

**About this task**


As a project owner, you can assign access by specifying a role when you add a user to your project. You can also assign more access or remove access for a user. Roles enable users to perform tasks on project resources such as running tests and viewing test results.

1. Choose the role in a project based on its associated actions.

Option	Description
<p><b>Owner of a project</b></p>	<ul style="list-style-type: none"> <li>◦ All Tester's actions</li> <li>◦ Update, archive, list, and view a project</li> <li>◦ Assign, update, delete, and list user roles for a project</li> <li>◦ Accept new member requests</li> <li>◦ Create, update, and delete a dataset classification in a project</li> <li>◦ Add, update, and remove an encrypted dataset from a classification</li> <li>◦ Update current row for a listed dataset in a project</li> <li>◦ Delete, list and view a report</li> <li>◦ Update, delete, and list a repository in a project</li> <li>◦ Get the test assets from a repository in a project</li> <li>◦ Create, update, delete and list secrets in a project</li> <li>◦ Create, update, delete, and list resource monitoring sources in a project</li> </ul>



Option	Description
	<ul style="list-style-type: none"> <li>◦ Get the content of a resource monitoring source in a project</li> <li>◦ Register, update, delete, and list resource monitoring agents</li> <li>◦ Create, configure, and delete the Jira change management system</li> <li>◦ Create and update defects in the Jira change management system</li> <li>◦ Create and update components in the system model</li> </ul>
<b>Tester of a project</b>	<ul style="list-style-type: none"> <li>◦ All Viewer's actions</li> <li>◦ List repositories in a project</li> <li>◦ Run a test and create a report</li> <li>◦ Stop a test while running</li> <li>◦ Get the test assets from a repository in a project</li> <li>◦ Get a list of datasets in a project</li> <li>◦ Get the content of a dataset</li> <li>◦ Update the current row for a listed dataset in a project</li> <li>◦ Add, update, and remove an encrypted dataset from a classification</li> <li>◦ Delete a report if you created the report</li> <li>◦ Create and update defects in the Jira change management system</li> <li>◦ Create and update components in the system model</li> </ul>
<b>Viewer of a project</b>	<ul style="list-style-type: none"> <li>◦ List your own roles on a project</li> <li>◦ List running tests</li> <li>◦ List and view a report</li> <li>◦ List resource monitoring sources in a project</li> <li>◦ Get the content of a resource monitoring source in a project</li> <li>◦ List the resource monitoring agents</li> <li>◦ View components in the system model</li> <li>◦ View defects in the Jira change management system</li> </ul>
<b>Non-member of a project</b>	<ul style="list-style-type: none"> <li>◦ Request to be a member of a project</li> </ul>

2. Open the project that you own.  
From the project details, you can see a list of project members for your project and then add or remove a members role.
3. Add or remove the member access by using the **Menu** icon  next to each user in the member list.

## Archiving or unarchiving server projects

You can archive a project when that project is no longer needed but you want to retain it for future reference. If you do want to use an archived project, you can unarchive it.


### About this task

As a project owner, you can archive projects that are inactive. Archived projects are not visible to users or project members, but owners can show or hide their archived projects. Archived projects can be unarchived.

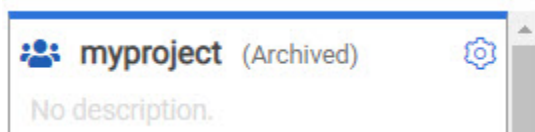


#### Note:

After you migrate the server data from a previous version to a newer version, the **Project Owner** must log in to HCL OneTest™ Server so that other members of the project can access the test assets in that project.

- Archive a project by following these steps:
  1. Open the project that you want to archive.
  2. From the project details, archive the project.
  3. Confirm that you want to archive the project when prompted. If you proceed, the project is archived and is hidden in your project list. To show or hide the archive projects, select the appropriate option by using the **Menu** icon  from the menu bar.

When visible, the archived project shows as archived in the list of **My Projects**.



**Note:** The option to configure a project continues to display after you archive a project.

- Unarchive a project by following these steps:
  1. Open an archived project and unarchive it.
  2. Confirm that you want to unarchive the project when prompted. If you proceed, the project is unarchived and is shown in your project list.

## Deleting server projects

You might want to delete a project when the project is no longer needed, and you want to free up disk space.

### About this task

As a project owner, you can delete a project that you no longer want. When you delete a project it is permanent. You cannot undo the delete project action.

After you delete the project, you can use the project name again.

1. Open the project that you want to delete.
2. From the project details, delete the project.
3. Confirm that you want to delete the project when prompted. If you proceed, the project is deleted including all of its test assets, results, and members.



**Note:** The repository that you used for the project is not deleted.

## An overview of test assets, modifications, and scheduled runs

After you configure a project with one or more Git repositories and run some test assets, you can view statistics about those test assets.

From the **Overview** page, several cards display charts, sliders, and tables of the statistics of the test assets in your server project. You can click some of the charts, sliders, or tables to see more information, such as the progress of a scheduled run or the results of a run. You can also review some statistics by date and see more details such as the proportion of test assets by type.

You can view a server project, which might contain multiple repositories and multiple branches, by selecting the name of the branch from the **Branch** drop-down list. For more information about selecting branches, see the related links.

### Test suites

This card displays a slider of the percentage of test suites by type. A test suite is a collection of tests, including schedules, compound tests, API Suites, and AFT Suites. You can hover over the slider and see the proportion of test suites by test suite type. The number of suites shown are relative to the selected branch.

### Individual tests

This card displays a slider of the percentage of individual tests by type that are used in the test suites. You might have performances tests, API tests, or UI tests according to the desktop clients that you used to create the tests. You can hover over the slider and see the proportion of individual tests by type. The number of tests shown are relative to the selected branch.

### Execution results

This card displays a chart of the proportion of test suites that were run by status and by date. If no date is selected, the card displays all the run results associated with the project since it was created. You

can see the run results by clicking a test suite status. You can also hover over the chart and see the test suite status and the proportion of test assets. You can view statistics by test suites run.

#### **Notes:**

- The results shown are generated from test suites run.
- The statistics exclude canceled test suites.
- The number of test suites and individual tests that are associated with your project are shown on the **Home** page along with a verdict summary. You can see the run results by clicking a verdict. You can also see the last date and time when you ran one or more test assets.
- The number of suites shown are relative to the selected branch.

#### **Last run**

This card displays a table of the last three runs by date and time. You can see the run results by clicking each last run.

#### **New or modified**

This card displays a chart of new and modified test assets that were made in your Git repository for a week or for the last seven days based on a selected date range. You can hover your mouse over each bar in the chart and see the proportion of test assets by type. The data shown is relative to the selected branch.

#### **Scheduled runs**

This card displays a table of the next three scheduled runs by date and time. You can see the progress of a scheduled run by clicking each scheduled run.

---

#### Related information

[Test assets and a server project on page 580](#)

[Selecting the global branch in a project on page 597](#)

## Managing repositories

After you create a project in HCL OneTest™ Server, you can add repositories that contain test assets and resources to the project. You can also change the repository details, update the user credentials of a repository, delete a repository, create a webhook for the configured repository.

### Tasks in a repository

To add a repository in your project and work on the repository, you can perform the tasks listed in the following table:

Task	More information
Adding a repository	<a href="#">Adding repositories to a server project on page 583</a>
Updating the repository credentials	<a href="#">Updating the authentication credentials of the repository on page 594</a>
Refreshing a repository	<a href="#">Refreshing repositories manually on page 595</a>
Deleting a repository	<a href="#">Deleting a repository on page 595</a>
Creating a webhook for the repository	<a href="#">Creating webhooks on page 598</a>

## Adding repositories to a server project

You can add repositories to a server project to access assets and resources available in the respective repository.

### Before you begin

You must have completed the following tasks:

- Added a project on HCL OneTest™ Server.
- Been granted permission to access the repository.

### About this task

As a project owner or a tester, you can add one or more repositories to your project. When you add a repository, the Git repository is cloned to your project. While adding a repository, you must provide the necessary authentication credentials that are set for the Git repository that you want to add to your project. For example, if the authentication type is SSH, then you must provide the Git URL, a deploy key, and a passphrase.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project, and then click **Manage**.
3. Click the **Repositories** tab.
4. Click **Add Repository**.

The **Add Repository** dialog box is displayed.

5. Enter the URL of the Git repository that you want to add to your project in the **Git Repository** field.
6. Enter the required credentials based on any of the following authentication methods configured in the repository.

To gain access to the repository, you must use any one of the authentication methods:

Authentication method	Credentials required
SSH	<ul style="list-style-type: none"> <li>◦ Deploy key</li> <li>◦ Passphrase</li> </ul>
HTTPS	<ul style="list-style-type: none"> <li>◦ User name</li> <li>◦ Password</li> </ul>

Authentication method	Credentials required
HTTP	<ul style="list-style-type: none"> <li>◦ User name</li> <li>◦ Password</li> </ul>

**Notes:**

- You must have defined the authentication type and set the authentication credentials in the Git repository.
- If you use SSH to connect to your remote repository and HCL OneTest™ Server displays an `Auth Fail` exception while using the `deliver changes` option, you can resolve this exception error by regenerating your SSH keys by using the `-m PEM` option.

7. Click **Add**.

The Git repository is added to your project in the team space.



**Note:** Depending on the size of the repository that you are cloning, it can take a few to several minutes to clone the repository.

8. **Optional:** Repeat the steps to add another repository.**What to do next**

You can perform the following actions on the repository that you added:

- Update the authentication credentials if they are changed in the Git repository configuration.
- Delete a repository if it is no longer required.
- Refresh a repository to fetch and synchronize changes from the remote repository.
- Configure a webhook to notify the server when there is a push event in the remote repository.

## Related information

[Managing repositories on page 592](#)

## Updating the authentication credentials of the repository

If the authentication credentials of the Git repository which was added to your project are changed, then you must update the credentials of the repository in HCL OneTest™ Server.


**Before you begin**

You must have completed the following tasks:

- Been assigned a *Member* or *Project Creator* role in a team space.
- Been assigned the *Tester* or *Owner* role in the project with access permissions to edit repositories.

**About this task**

In HCL OneTest™ Server, the repository is refreshed at regular intervals to pull the latest changes committed to the Git repository. After you update your authentication credentials of the repository, the repository is refreshed to ensure that you have access to the Git repository and to the latest assets in the repository.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.
3. Go to the **Repositories** page.
4. Click the **Menu** icon  and then click the **Change** icon.

The **Edit repository** page is displayed.

5. Enter the new credentials and click **Update**.

The new credentials are updated and your access to the repository is retained.

## Deleting a repository

You can delete repositories that you no longer require in your test environment.

**Before you begin**

You must have completed the following tasks:

- Configured the repository that contains the test assets in your project.
- Been assigned a project owner or a tester role with access to delete repositories.

**About this task**

You can delete a repository that you have configured for your project.

1. Open your project and go to the Repositories page.

You can see a list of repositories that you added to the project.

2. Delete a repository that you want to remove from the project.

The repository is deleted and removed from the list.


## Refreshing repositories manually

You can manually trigger the refresh activity for a repository to instantly fetch and synchronize the changes from the remote repository.

**Before you begin**

You must have completed the following tasks:

- Configured the repository that contains the test assets in your project.
- Been assigned a *Member* or a *Project Creator* role.
- Been assigned a *Owner* or a *Tester* role.





1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open your project.
3. Go to the **Repositories** page.
4. Click the **Menu** icon , and then select **Refresh**.

The refresh activity is triggered.



**Note:** The **Refresh** option is disabled when the refresh is in progress. After the refresh is completed, the **Refresh** option is enabled.

The following table lists the icons and their description based on the status of the refresh activity:

Refresh icon	Description
	<p>The last refresh activity was successful.</p>
	<p>The last refresh activity was successful but errors were found while processing some of the test assets. To view the error details of the test assets for a specific repository, you can click the arrow inline with the repository name.</p> <p>You can resolve the errors of these test assets in the Git server, and then refresh the server repository, if required.</p> <p> <b>Note:</b> The repository, which contains the branch that is selected in the <b>Branch</b> field, displays the unprocessed test assets for that branch. Other repositories display the unprocessed test assets for their default branch.</p>
	<p>The refresh activity was not successful due to any one of the following reasons:</p> <ul style="list-style-type: none"> <li>◦ Internal error</li> <li>◦ Connection issue</li> <li>◦ Authentication failure</li> <li>◦ Disk quota exceeded</li> <li>◦ Repository unreachable</li> </ul>

## Results



You have manually refreshed the repository and the **Last Refreshed** field is updated with the following details:

- The time when it was last refreshed in the remote repository.
- Latest commit ID and message of the remote repository.

## Selecting the global branch in a project

After you add one or more repositories to your project, you can start accessing all branches in all repositories of that project. You can select only one branch as a global branch from the list of branches in the **Execution**, **Datasets**, and **Overview** pages.

### Before you begin

You must have completed the following tasks:

- Been assigned a member or Project Creator role in a team space
- Added at least one repository in your project.

### About this task

You can access all the branches contained in all the repositories, which are added to the project, by using the **Branch** field on the **Execution**, **Datasets**, and **Overview** pages.

After you select a branch from any of the repositories as the global branch, this selection is retained for you as the global branch unless you change the branch on any of the following pages: **Overview**, **Execution**, or **Datasets** pages.



#### Note:

- When you add the first repository to a project, the global branch is set to the default branch of the repository.
- Branches are arranged in an increasing order of their names in the list.
- When you change the branch in one of the pages, then the same branch is displayed as selected in the other pages and the contents of that page displays the assets from the selected branch.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Open the project.

The project **Overview** page is displayed.

3. Go to the **Overview** page.
4. Select a branch from the **Branch** field.

The branch that you selected in the **Overview** page is reflected in the **Execution** and **Datasets** pages and the test assets are filtered based on the global branch selection on each of these pages.

### Results


You have selected a global branch in a project.

## Creating webhooks

You can configure a webhook for a repository in the Git server to notify HCL OneTest™ Server whenever there is a push event in the remote repository. Then, the refresh activity is triggered to fetch the latest changes to the server repository.

### Before you begin

You must have completed the following tasks:

- Configured the repository that contains the test assets in your project.
  - Been assigned a *Member* or *Project Creator* role in a team space.
  - Been assigned the *Tester* or *Owner* role in the project.
1. Log in to HCL OneTest™ Server and open the team space that contains your project.
  2. Open your project.
  3. Go to the **Repositories** page.
  4. Click the **Menu** icon  and then click **Webhook**.

The **Webhook Configuration** dialog box is displayed.

5. Copy the URL and the Secret of the repository that you want to configure.



**Note:** The URL and the Secret details are specific to a repository.

6. Click the link of the repository for which you want to configure the webhook.

The remote repository page is displayed.

7. Paste the details, which you copied from HCL OneTest™ Server, in the appropriate fields of the remote repository.

### Results

You have configured the webhook for a repository in the Git server. The changes of the remote repository are synchronized in the repository of your project.

## Protecting API test assets by using secrets

Secrets are key-value pairs that are created for your project in HCL OneTest™ Server under a secrets collection. You can create secrets collections for your project that enable you or members in your project to use secrets at test runtime either in HCL OneTest™ Server or in desktop clients.

The secrets collections in a project in HCL OneTest™ Server has a separate access control list managed by the members with access to the secrets collections. Controlling access to secrets means controlling access to applications and systems under test. The introduction of secrets (under secrets collections) for a project has

simplified managing access to separate environments. If a member of a project does not have access to a secret, for example, a *server credential* then the member cannot accidentally or maliciously run tests against that server. For example, tests that must access the database server by using the server credentials to retrieve stored data can only be run by a member if the access to the secrets is granted.



**Note:** Secrets and secrets collections are applicable to test assets authored in HCL OneTest™ API that enable running tests in defined environments. Secrets are not applicable to tests authored in HCL OneTest™ UI or HCL OneTest™ Performance.

As a project member with the *Owner* or *Tester* role, you can create secrets collections in the project. You can grant or restrict access to the secrets collection that you create in the project.

Members with access to a secrets collection can access, edit, or delete the secrets collection in HCL OneTest™ Server and can view secrets, edit secrets, or delete secrets.

Members with access to secrets collections can grant access to or remove the following:

- Other members added specifically
- All members with a specific role

Members in the project with the *Owner* or *Tester* role and with access the secrets collection can use the secrets in the secrets collection, in tests at runtime.

If you are configuring a project to run an API Suite with tests that refer to secret values, you must configure the secrets under a secrets collection by using the **SECRETS** tab. You must complete the following tasks:

1. Create a secrets collection. See [Step 1 on page 600 in Managing secrets collections on page 599](#).
2. Add secrets in the secrets collection created. See [Step 1 on page 602 in Creating a secret in a secrets collection on page 601](#).
3. Grant access to project members or member roles, who can access the secrets collection. See [Step 1 on page 602 in Granting access to members or member roles on page 602](#).

## Managing secrets collections

You can create secrets in a secrets collection for your project. Secrets are credentials required in certain tests during test runs. Secrets stored in the collection can be used by members to run tests on different environments and eliminates the need to store secrets in multiple locations. You can opt to edit or delete a secrets collection that you configured for your project any time after you create a secrets collection.

### Before you begin

- You must have created a project on HCL OneTest™ Server. See [Test assets and a server project on page 580](#).
- You must have completed the following tasks before you edit or delete a secrets collection:

- Configured a secrets collection in your project.
- Created secrets in the selected secrets collection. See [Creating a secret in a secrets collection on page 601](#).
- You must be a member with the *Owner* or *Tester* role to create a secrets collection.
- You must be a member with access to the secrets collection to edit or delete the secrets collection. See [Granting access to members or member roles on page 602](#).

### About this task


You must configure secrets collections in your project so that the members of the project can use secrets contained in a collection during test runs. You can configure secrets so that you can use them in different test environments.

As a member with access to the secrets collection, you can opt to edit or delete a secrets collection configured in a project. For example, you might want to edit the secrets collection name or delete the secrets collection if the testing environment has changed and if secrets that are configured earlier are not required.

- To create a secrets collection, go to [Step 1 on page 600](#).
- To edit or delete a secrets collection, go to [Step 4 on page 600](#).

To create a secrets collection:

1. To create a secrets collection while configuring a new project in the HCL OneTest™ Server UI, open the **SECRETS** tab in the **Project Configuration** and create a secrets collection. Use **Add Collection**.
2. Alternatively, to create a secrets collection in an existing project, complete the following steps:
  - a. Log in to HCL OneTest™ Server and from the User Interface (UI) open the project listed under **My Projects** for which you want to create a secrets collection.
  - b. Open the Project Configuration page, and then open the **SECRETS** tab to create a secrets collection.
3. Enter a name for the secrets collection as its *Identifier*.

 **Tip:** You can create a secrets collection that contains secrets for a particular test environment in your project. For example, the secrets collection *test\_env* can contain secrets that application testers can use in tests that they run while the secrets collection *dev\_env* can contain secrets that application developers can use in tests they run.

A message is displayed for the successful creation of the secrets collection.

The secrets collection created is displayed.


You can add secrets to the secrets collection you created.

To edit or delete a secrets collection:

4. Log in to HCL OneTest™ Server and from the UI open the project listed under **My Projects**.
5. Open the secrets collection from the **SECRETS** tab in the Project Configuration page.

If there are multiple secrets collections in the project, select the secrets collection that you want from the list.

- To edit a secrets collection, go to [Step 6 on page 601](#).
  - To delete a secrets collection, go to [Step 7 on page 601](#).
6. To edit a secrets collection, complete the following steps:


- a. Click the **Edit** icon  to edit the selected secrets collection.



**Note:** The **Edit** icon  is displayed only for the project owner.

- b. Edit the name of the secrets collection, and update the secrets collection.

The secrets collection is updated with the updated name.

7. To delete a secrets collection, click the **Delete** icon  to delete the selected secrets collection.

The selected secrets collection is removed from the list of secrets collections configured for the project.

## Results

You have completed the following tasks:

- Created a secrets collection for your project.
- Edited the name of a secrets collection in your project.
- Removed a secrets collection from your project.

## What to do next

- If you have created a new secrets collection, you must add secrets to your secrets collection.
- You must provide access to project members or member roles to the secrets collection by selecting members or member roles.

## Creating a secret in a secrets collection

You must create secrets in the secrets collections configured in your project so that the secrets contained in a collection can be used in certain tests by members of the project with access to the secrets collections during an API suite run.

### Before you begin

You must have created a project on HCL OneTest™ Server and configured a secrets collection in your project.

You must be a member with access to the secrets collection.

### About this task

You can also configure secrets such that the secrets can be used across different test environments by members with access to the secrets collection. Secrets correspond to the environment variables or tags that you create in a HCL OneTest™ API project specific to an environment.

1. To create a secret under a secrets collection while configuring a new project in the HCL OneTest™ Server UI, select the secrets collection listed in the **SECRETS** tab in the Project Configuration page and create a secret under the secrets collection.
2. Alternatively, to create a secret under a secrets collection in an existing project, complete the following tasks:
  - a. Log in to HCL OneTest™ Server and from the UI open the project listed under **My Projects**.
  - b. Open the secrets collection from the **SECRETS** tab in the Project Configuration page.
3. Enter a name for the secret as its *Identifier* and enter the `password` as the *Value* for the secret. For example, under the secrets collection (named as `test_env`), enter the name of the secret to access a database as `dbcred` and enter the `password` required to access the database as its value.

A message is displayed for successful creation of the secret.

## Results

You have created secrets in the selected secrets collection for your project.

## What to do next

- You can view, edit, or delete the secrets created under a secrets collection any time you want.
- You can use the secrets in the tests that require these secrets during test runs.

## Granting access to members or member roles

You can grant or revoke access to the secrets collection in your project to individual members with different roles or the all members with a specific role. Without access to the secrets collection, members cannot view, create, edit, delete, or use the secrets in the secrets collection.

## Before you begin

You must have created a project on HCL OneTest™ Server and configured a secrets collection in your project.

You must be a member with access to the secrets collection.

1. To grant access to a secrets collection while configuring a new project in HCL OneTest™ Server UI, select the secrets collection listed in the **SECRETS** tab in the Project Configuration page.
2. Alternatively, to grant access to a secrets collection in an existing project, complete the following tasks:
  - a. Log in to HCL OneTest™ Server and from the UI open the project listed under **My Projects**.
  - b. Open the secrets collection from the **SECRETS** tab in the Project Configuration page.  
If there are multiple secrets collections in the project, select the secrets collection that you want from the list.

3. To grant access to a secrets collection in a new project or an existing project, select from the following methods:
  - To add all members with a specific role, click the role listed under **Grant access to role**. For example, if you select **Testers**, then all members in the project with a tester role are granted access to the secrets collection. You can select any role or all the roles listed.
  - To select specific members to grant access to the selected secrets collection, enter the name or the email ID of the member in the field box and add them from the list that is displayed.




**Note:** Members added specifically are listed under **Members with access to this collection** but all the members granted access solely due to their roles are not listed.



**Important:** Irrespective of the role that the member (*Owner, Tester or Viewer*) was assigned in the project, the access to the secrets collections has to be specifically granted to the members from the **SECRETS** tab.

### Removing access to a secrets collection

4. To remove access granted to all members with a specific role or a specific member, select from the following methods:
  - To remove all members with a specific role, click the role listed under **Grant access to role** to clear the selection. For example, if **Testers** is selected and you clear it, then all members in the project with a tester role are removed from the access list to the secrets collection.
  - To remove specific members with access to the secrets collection, select the member and click the **Delete** icon .



#### Notes:

- Any member with access to the secrets collection can remove access of other members specifically added or of all members with a specific role.
- Members with access to the secrets collection can remove themselves from the access list. Members can do this when there is at least one member remaining in the list. After removing themselves, members cannot add themselves back to the access list and must be added by any of the other remaining members in the list.

### Results

You have added members from your project or members with specific role to the access list of people who can access secrets in the selected secrets collection, or you have removed specific members or members with specific role from the access list.

### What to do next

You can create secrets under secrets collections for your project.

## Managing secrets

You can view, edit, or delete the secrets configured under a secrets collection any time after you have created secrets or after you were granted access to the secrets collection. You can change the value of the secret by editing the secret. You can delete secrets that you no longer require in your test environment.

### Before you begin

You must have created a project on HCL OneTest™ Server and configured a secrets collection in your project.





You must have created secrets in the selected secrets collection or the secrets collection must contain secrets.

You must be a member with access to the secrets collection.

1. Log in to HCL OneTest™ Server and from the UI open the project listed under **My Projects**.
2. Complete the following steps:
  - a. Open the secrets collection from the **SECRETS** tab in the Project Configuration page.
  - b. Optionally, select the secrets collection that you want from the list if there are multiple secrets collections in the project.

The secrets configured in the selected secrets collection are displayed.

3. Complete the steps for the task you want to perform as listed in the following table:

Task	Steps
Viewing a secret value	<p>Click the <b>Show</b> icon  for the secret you want to view its value, which most likely is a password for the secret.</p> <p>The value configured for the secret is displayed.</p>
Editing a secret value	<p>Click the <b>Edit</b> icon  for the secret you want to edit, and enter a <i>new value</i> for the secret as its <i>Value</i>. The value can be a password for the secret.</p> <p> <b>Note:</b> You can only change the value of the secret.</p> <p>The value of the selected secret is changed.</p>
Deleting a secret	<p>Click the <b>Delete</b> icon  in the row of the secret you want to delete.</p> <p>After deleting it, the secrets list in the collection is removed from the list.</p>

### Results



- You viewed the password configured of the secret under a secrets collection that you created or were granted access.
- You changed the secret value of the secret under a secrets collection in your project.
- You deleted and removed the secret from the selected secrets collection in your project.

### What to do next

You can use secrets in the tests that require these secrets during test runs.

---

Related information

[Test run configurations on page 305](#)

[Becoming a project member on page 587](#)

## Managing an encrypted dataset

You can use encrypted datasets to limit access to confidential information such as account number or passwords. You can arrange data by an appropriate category so that project members can use datasets more effectively in certain tests and protect them.

### About this task

A dataset can contain classified information that other members can access with permission. As a project owner, you can group encrypted datasets into different classifications and enable project members to view and edit datasets and run tests associated with the encrypted datasets. After you have created a classification, you can change the classification for a dataset. You can also delete a classification if you do not require it in your test environment.



### Notes:

- You must grant access and provide an encryption key of an encrypted dataset to other members of the project to work with the encrypted dataset.
- A project member who has been added as a **Tester** role can work with the encrypted dataset.

## Creating a classification

As a project **Owner**, you can organize encrypted datasets by creating a classification so that project members can use and protect datasets more efficiently.


### Before you begin

You must have completed the following tasks:






- Created a project in . See [Test assets and a server project on page 580](#).
- Configured the repository that contains the test assets in your project. See [Adding repositories to a server project on page 583](#).
- Created at least one dataset and encrypted the dataset with an encryption key. See [Dataset encryption on page 154](#).

### About this task

After creating a classification, you can grant access and provide the encryption key to other members of the project to work with the encrypted dataset.

1. Go to the URL.
2. Enter your user name and password, and then click **Login**.
3. Open your project from the UI.
4. Click **Manage**, and then the **DATA SECURITY** tab.
5. Click **New classification** and enter a name for the classification.
6. Click **Create**.
7. Click the **Add** icon  to select the encrypted dataset to become part of the new classification.
8. Select a dataset from the list and enter the encryption key in the **Password** field for the dataset, and then click the **Add** icon.

The encrypted dataset that is added to the classification is displayed.

DETAILS	REPOSITORIES	SECRETS	DATA SECURITY
Classification			 
Credentials			
Datasets			
Name	Encrypted Columns	Actions	
JqueryAccordion	Tag programming languages_ _2	 	

### What to do next

You can grant access to other project members to use the encrypted dataset.

## Editing or deleting a classification

After you have created a classification, you can edit the name of the classification. You can also delete a classification when it is not required in your test environment.

### Before you begin

You must have completed the following tasks:

- Created at least two classifications. See [Creating a classification on page 605](#).

### About this task

You can opt to edit or delete a classification for your project any time after you create a classification. For example, you might want to edit the name of the classification or delete the classification if the classification that is created earlier are not required.




**Note:** You must be a project **Owner** to create, edit, or delete a classification.

- To edit a classification, go to [Step 1 on page 607](#).
- To delete a classification, go to [Step 2 on page 607](#).


1. Perform the following steps to edit a classification:

- a.
- b.
- c. Select a classification that you want to edit from the list.



- d. Click the **Edit** icon  to edit the selected classification.
- e. Edit the name of the classification, and then click **Save**.

2. Perform the following steps to delete a classification:

- a.
- b. Select a classification that you want to delete from the list.
- c. Click the **Delete** icon .

d. Click **Delete** in the **Delete classification** window to confirm the deletion of the classification.



**Note:** Before deleting a classification, you must move the added encrypted datasets to another classification else the delete icon is disabled.


## Results

- You have edited the name for a classification in your project.
- You have deleted a classification.







## Moving an encrypted dataset to another classification

When you add many encrypted datasets to the same classification, you can move some of them to another classification.

### Before you begin

- 1.
- 2.
3. Select a classification from the list that has the encrypted dataset.
4. Click the **Edit** icon  from the **Actions** column of a dataset.
5. Select a classification from the list and enter the encryption key of the dataset in the **Password** field.
6. Click **Save**.

A classification for a dataset is updated successfully.

DETAILS	REPOSITORIES	SECRETS	DATA SECURITY
Classification			 
Imp Credentials			
Datasets			
Name	Encrypted Columns	Actions	
JqueryAccordion	Tag programming languages_ _2	 	

## Results


You have moved the encrypted dataset from one classification to another.

## Removing a dataset from the classification

You can remove a datasets added to a classification when they are no longer required.

**Before you begin**

You must have completed the following tasks:

- - Created a classification and added the encrypted dataset to it. See [Creating a classification on page 605](#).
- 1.
  - 2.
  3. Select a classification from the list that has an encrypted dataset.
  4. Click the **Delete** icon  from the **Actions** column of a dataset.
  5. Click **Remove** in the **Change the classification for the Dataset** window.



**Note:** Removing dataset from the classification also removes the password stored in for encrypted data. You must enter the password again to gain access to the encrypted columns.

**Results**

You have removed the datasets from a classification.

## Granting classification access to members or members roles

You can grant or revoke access to the classification in your project to individual members with different roles or the all members with a specific role. Without access to the classification, members cannot view, create, edit, delete, or use the classification.

**Before you begin**

You must be a project **Owner** and have completed the following tasks:

- - Added one or more users to your project. See [Adding users to a project on page 585](#).
  - 
  - Created at least one classification. See [Creating a classification on page 605](#).
- 1.
  - 2.
  3. Select the classification from the drop-down list.
  4. Choose any of the following methods to grant access to a member:  
**Choose from:**

- To add all members with a specific role, click the role listed under **Grant access to role**. For example, if you select **Testers** then all members in the project with a tester role are granted access to the selected classification. You can select any role or all the roles listed.
- To select specific members to grant access to the selected classification, enter the name or the email ID of the member in the **Grant access to member** field and add them from the list that displays.




**Note:** Members added specifically are listed under **Members with access to this classification** but the members added for a role are not displayed.



**Important:** Irrespective of the role that the member (**Owner, Tester or Viewer**) was assigned in the project, the access to the classification has to be specifically granted to the members from the **DATA SECURITY** tab.

5. Choose any of the following methods to revoke access from a member:

**Choose from:**

- To remove all members with a specific role, click the role listed under **Grant access to role** to clear the selection. For example, if **Testers** is selected and you clear it, then all members in the project with a tester role are removed from the access list to the classification.
- To remove specific member with access to the classification, select the member and click the **Delete** icon .



**Notes:**

- Any member with access to the classification can remove access of other members specifically added or of all members with a specific role.
- Members with access to the classification can remove themselves from the access list provided that there is at least one member in the list. After removing themselves, members cannot add themselves back to the access list and must be added by any of the other members in the list.

## Results

You have granted or revoked the classification access to project members.

---

Related information

[Adding a project on page 582](#)

## Managing notifications

HCL OneTest™ Server provides a feature to display notifications for different events within the user interface (UI) of HCL OneTest™ Server. You can configure an SMTP server on HCL OneTest™ Server when you want HCL OneTest™ Server to send out notifications about the different events as emails to the subscribed users.

HCL OneTest™ Server provides you the following features to receive notifications about server events:

Channels	Description
In-app	Notifications are displayed within the UI of HCL OneTest™ Server.
SMTP	Notifications are sent by HCL OneTest™ Server in emails to subscribed users.

When you register as a user in HCL OneTest™ Server, the in-app and SMTP channels are automatically subscribed for you.

### Managing notifications in the server UI

Viewing the in-app notifications or notifications in the server UI is a feature that is enabled automatically when you register as a user in HCL OneTest™ Server.

You are subscribed to the following channels for all events as the default option when you sign up on the server.

- In-app
- SMTP

If you want to receive notifications about specific events and not all events, you can change the default subscriptions and modify your subscription for the server events. You can also view the in-app notifications that are displayed for you. See [Managing in-app notifications on page 612](#).

### Managing SMTP notifications

You, as a *Server Administrator* or *Team Space Owner* can configure an existing Simple Mail Transfer Protocol (SMTP) server so that HCL OneTest™ Server can send out notifications about the server events in emails to the registered users. See [Configuring an SMTP server to manage email notifications on page 622](#).

You are subscribed to the following channels for all events as the default option when you sign up on the server.

- In-app
- SMTP

If you, as a registered user of HCL OneTest™ Server want to receive the email notifications about specific events and not all events, you can change the default subscriptions and modify the subscription for the server events. See [Managing email notifications on page 617](#).

## Managing in-app notifications

You can manage the in-app notifications by changing the subscriptions that are automatically enabled for you for all server events. You can view the server events that are automatically subscribed and then change the subscription for any or all events. For example, as a team space owner, you can choose to be notified when members of the team space create new projects or delete their projects but not be notified when tests are run.


### Before you begin

- 
- 

### About this task

The in-app notifications that are automatically subscribed for you, are displayed in the server UI under the following categories for the server events that occur:

Category	Notifications displayed
User	Include information about actions performed by team space members in the team space.
System	Include information about actions performed by registered users at the server level.

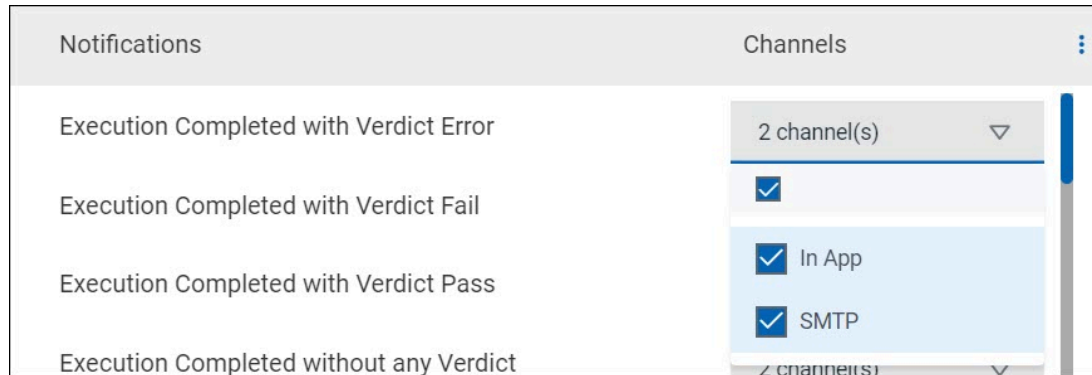
- 1.
2. Select from the following actions that you want to perform:
  - Modify notification settings. Go to [Step 3 on page 612](#).
  - View notifications. Go to [Step 4 on page 614](#).
  - Unsubscribe from the in-app notifications. Go to [Step 5 on page 615](#).
3. Perform the following steps to modify the notification settings:
  - a. Click the **User** icon  on the menu bar.
  - b. Click **Notification Settings**.
 

The **Notification Settings** dialog is displayed.
  - c. Change the language in which you want to receive the notifications, if you do not want to use the default language.
 

The default language is the locale language set on the computer that hosts .

You can notice that for each of the server events, both the channels are displayed as selected, which is the default selection unless you have modified the subscriptions.






- d. Identify the server events for which you want to change the subscription.
- e. Perform the actions indicated in the following table to clear the subscriptions for server events that you do not want to be notified and to select the in-app subscription for the server events that you want to be notified.

When	Action
<p>You do not want to receive the in-app notifications for a specific server event.</p>	<p>Perform the following steps:</p> <ol style="list-style-type: none"> <li>i. Expand the channel list in the row of the server event.</li> <li>ii. Clear the checkbox in the header row to clear the subscriptions to both channels.</li> <li>iii. Click <b>Apply</b>.</li> </ol> <p>You have unsubscribed from both channels and the in-app notifications for the server event is not displayed for you when the event occurs.</p>
<p>You want to receive the in-app notifications for a specific server event.</p>	<p>Perform the following steps:</p> <ol style="list-style-type: none"> <li>i. Expand the channel list in the row of the server event.</li> <li>ii. Select any of the following actions: <ul style="list-style-type: none"> <li>▪ Clear the <b>SMTP</b> checkbox.</li> </ul> <p>The <b>In App</b> selection is retained while the checkbox in the header row and for the <b>SMTP</b> options are cleared.</p> <ul style="list-style-type: none"> <li>▪ Clear the checkbox in the header row to clear the subscriptions to both channels, and then select the <b>In App</b> checkbox.</li> </ul> </li> <li>iii. Click <b>Apply</b>.</li> </ol>

When	Action
	You have unsubscribed from the SMPT channel and retained the subscription for the in-app notifications for the server event.

f. Go to [Results on page 617](#).

4. Perform the following steps to view the in-app notifications:

a. Click the **Notifications** icon  on the menu bar.

The **Notifications** dialog is displayed. The notifications are displayed in the following tabs:

- User
- System

The notifications in the **USER** tab is displayed as the default option.

b. View the notifications in the **USER** tab.

c. Click the **SYSTEM** tab to view the notifications at the server level.

You can find the information about the default notifications that you can view based on your role in the server as in-app notifications in the **USER** tab and **SYSTEM** tab.

Category	Notification details	or	or		
User notifications	Information about events at the server level or across team spaces performed by other team space owners.	✓	✓	✗	✗
	Information about events in the team space performed by other team space members.	✓	✗	✗	✗
	Information about events in the project performed by other project members.	✗	✓	✓	✗
	Information about events performed by you in a project.	✗	✗	✓	✓


Category	Notification details	or	or		
System notifications	Information about events at the server level or across team spaces.	✓	✓	✗	✗
	Information about events in the team space performed by other team space members.	✓	✓	✗	✗
	Information about events in the projects at the team space level performed by other project members.	✓	✓	✗	✗

d.

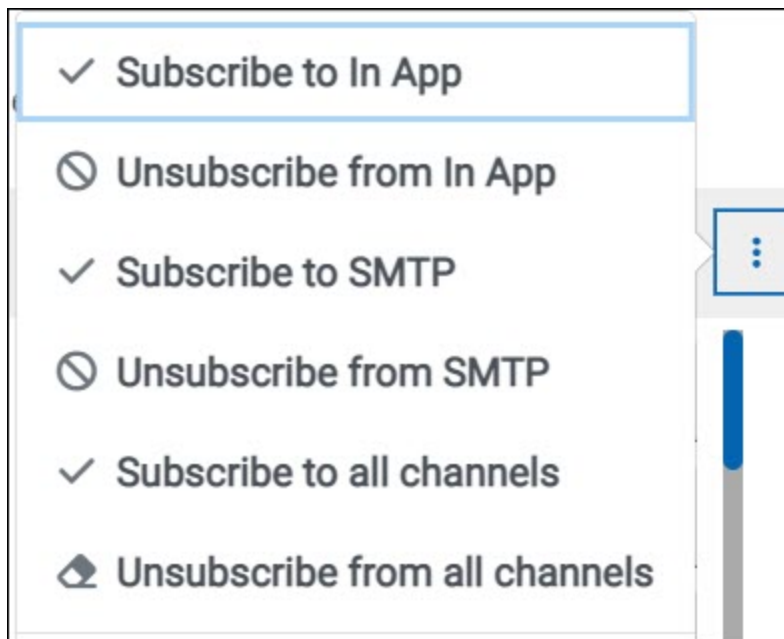
5. Perform the following steps to unsubscribe to the in-app notifications:

a.

b.

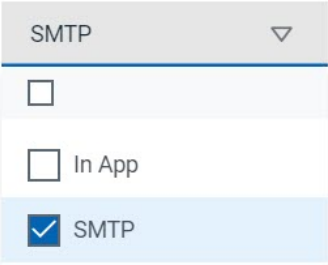
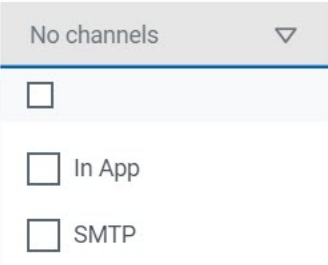
c. Click the **Open action menu** icon  in the row of **Notifications**.

The following panel is displayed:



You can notice that the subscriptions to in-app, SMTP, and all channels are displayed as selected.

d. Perform the actions indicated in the following table for the option you want:

When	Action	Results in
<p>You do not want to receive any in-app notifications.</p>	<p>Click the <b>Unsubscribe from In App</b> option, and then click <b>Apply</b>.</p>	<p>The in-app channel selection for all server events is cleared and you are unsubscribed from the in-app channel. The SMTP notifications is displayed as selected for subscription for all server events.</p> 
<p>You do not want to receive any in-app or the SMTP email notifications.</p>	<p>Click the <b>Unsubscribe from all Channels</b> option, and then click <b>Apply</b>.</p>	<p>The in-app channel and the SMTP channel selections for all server events are cleared and you are unsubscribed from both the channels. No channel is displayed as selected for subscription for all server events.</p> 
<p>You want to subscribe to the in-app notifications for all server events.</p>	<p>Click the <b>Subscribe to In App</b> option, and then click <b>Apply</b>.</p>	<p>The in-app channel for all server events is selected and you are subscribed to the in-app channel. The in-app notifications is</p>

When	Action	Results in
		displayed as selected for subscription for all server events.
You want to receive notifications as in-app and as SMTP email notifications.	Click the <b>Subscribe to all Channels</b> option, and then click <b>Apply</b> .	The in-app channel and the SMTP channel are displayed as selected for all server events and you are subscribed to both the channels.

e.

## Results

You completed the following tasks:

- Modified the notification subscriptions.
- Viewed the notifications displayed for you in the **USER** tab and **SYSTEM** tab.
- Unsubscribed from or subscribed to either the in-app channel or both the channels.

---

Related information

[Managing notifications on page 611](#)

## Managing email notifications

You can manage the email notifications by changing the subscriptions that are automatically enabled for you for all server events. You can view the server events that are automatically subscribed and then change the subscription for any or all events. For example, as a project owner, you can choose to be notified when members of the project add a repository or run a test in a team space but not be notified when projects are created in the team space.

### Before you begin

- Ensured that the or has configured a Simple Mail Transfer Protocol (SMTP) server. See [Configuring an SMTP server to manage email notifications on page 622](#).
- 
- 

### About this task

The email notifications in the SMTP channel are automatically subscribed for you for all server events. You can view the server events that are subscribed in the **Notifications Settings** dialog. You can either unsubscribe from the SMTP channel or change your subscriptions for the server events for which you do not want the email notifications.

- 1.
2. Select from the following actions that you want to perform:
  - Modify notification settings. Go to [Step 3 on page 618](#).
  - Unsubscribe from the email notifications. Go to [Step 4 on page 619](#).
3. Perform the following steps to modify the email notifications settings:

a. Click the **User** icon  on the menu bar.

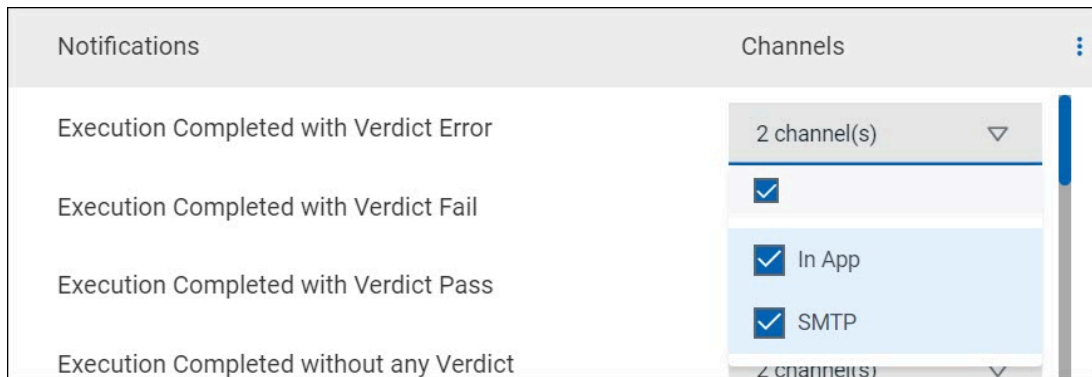
b. Click **Notification Settings**.

The **Notification Settings** dialog is displayed.

c. Change the language in which you want to receive the email notifications, if you do not want to use the default language.

The default language is the locale language set on the computer that hosts .

You can notice that for each of the server events, both the channels are displayed as selected, which is the default selection unless you have modified the subscriptions.



- d. Identify the server events for which you want to change the subscription.
- e. Perform the actions indicated in the following table to clear the subscriptions for server events that you do not want to be notified and to select the SMTP subscription for the server events that you want to be notified in emails.

When	Action
You do not want to receive the email notifications for a specific server event.	Perform the following steps: <ol style="list-style-type: none"> <li>i. Expand the channel list in the row of the server event.</li> <li>ii. Clear the checkbox in the header row to clear the subscriptions to both channels.</li> <li>iii. Click <b>Apply</b>.</li> </ol>

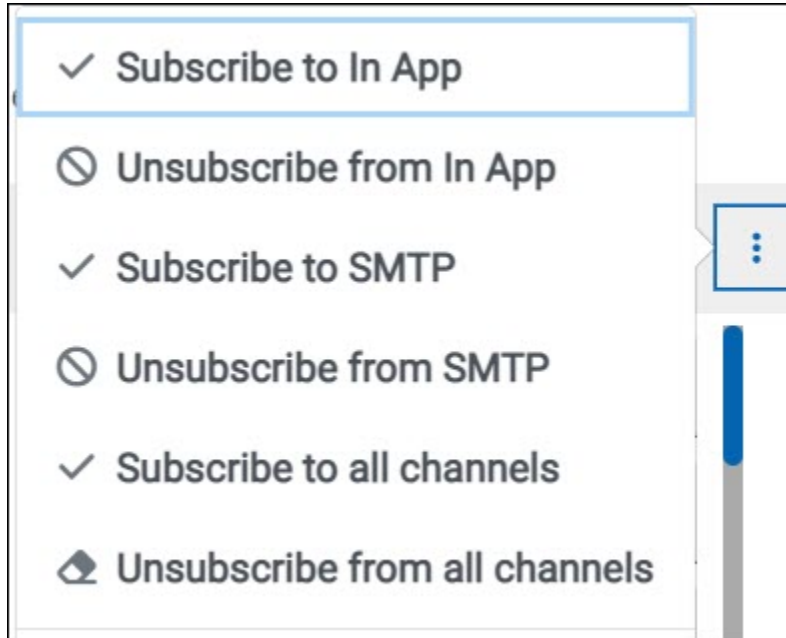
When	Action
	You have unsubscribed from both channels and the email notifications for the server event are not sent to you when the event occurs.
You want to receive the email notifications for a specific server event.	<p>Perform the following steps:</p> <ol style="list-style-type: none"> <li>i. Expand the channel list in the row of the server event.</li> <li>ii. Select any of the following actions: <ul style="list-style-type: none"> <li>▪ Clear the <b>In App</b> checkbox.</li> </ul> <p>The <b>SMTP</b> selection is retained while the checkbox in the header row and for the <b>In App</b> options are cleared.</p> <ul style="list-style-type: none"> <li>▪ Clear the checkbox in the header row to clear the subscriptions to both channels, and then select the <b>SMTP</b> checkbox.</li> </ul> </li> <li>iii. Click <b>Apply</b>.</li> </ol> <p>You have unsubscribed from the in-app channel and retained the subscription for the email notifications for the server event.</p>

f. Go to [Results on page 621](#).

4. Perform the following steps to unsubscribe to the email notifications:

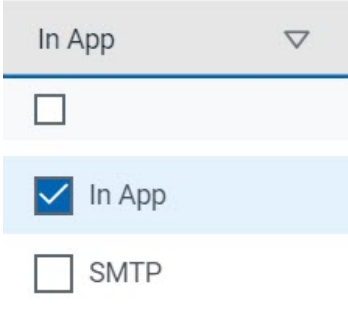
- a.
- b.
- c. Click the **Open action menu** icon  in the row of **Notifications**.

The following panel is displayed:

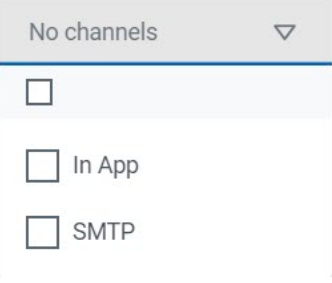


You can notice that the subscriptions to in-app, SMTP, and all channels are displayed as selected.

d. Perform the actions indicated in the following table for the option you want:

When	Action	Results in
<p>You do not want to receive any email notifications.</p>	<p>Click the <b>Unsubscribe from SMTP</b> option, and then click <b>Apply</b>.</p>	<p>The SMTP channel selection for all server events is cleared and you are unsubscribed from the SMTP channel. The <b>In App</b> option is displayed as selected for subscription for all server events.</p> 
<p>You do not want to receive any in-app or the SMTP email notifications.</p>	<p>Click the <b>Unsubscribe from all Channels</b> option, and then click <b>Apply</b>.</p>	<p>The in-app channel and the SMTP channel selections for all server events are cleared and</p>



When	Action	Results in
		<p>you are unsubscribed from both the channels. No channel is displayed as selected for subscription for all server events.</p> 
<p>You want to subscribe to the email notifications for all server events.</p>	<p>Click the <b>Subscribe to SMTP</b> option, and then click <b>Apply</b>.</p>	<p>The SMTP channel for all server events is selected and you are subscribed to the in-app channel. The <b>In App</b> option is displayed as selected for subscription for all server events.</p>
<p>You want to receive notifications as in-app and as SMTP email notifications.</p>	<p>Click the <b>Subscribe to all Channels</b> option, and then click <b>Apply</b>.</p>	<p>The in-app channel and the SMTP channel are displayed as selected for all server events and you are subscribed to both the channels.</p>

e.

## Results

You completed the following tasks:

- Modified the notification subscriptions.
- Unsubscribed from or subscribed to either the SMTP channel or both the channels.

---

Related information

[Managing notifications on page 611](#)

## Configuring an SMTP server to manage email notifications

Before you can receive email notifications about events that occur in HCL OneTest™ Server, you need to configure an existing Simple Mail Transfer Protocol (SMTP) server in HCL OneTest™ Server. When events occur in HCL OneTest™ Server, email notifications are sent to the registered users of HCL OneTest™ Server if they are subscribed to the SMTP notification channel.

### Before you begin


You must have completed the following tasks:

- Ensured that you are assigned a role as a *Team Space Owner* in the team space. See [Managing members and their roles in a team space on page 576](#).
- Ensured that an existing SMTP server is available to be connected to HCL OneTest™ Server.

### About this task

The configuration works for emails that are supported by the SMTP server. To use this capability, you must enable HCL OneTest™ Server to connect to the SMTP server to send email notifications.

As a server administrator, you can configure email notifications so that other users can receive information about actions on the server projects.

1. Log in to HCL OneTest™ Server and open the team space that contains your project.
2. Click the **Settings** icon  from the menu bar.

The **Notifications** page is displayed.

3. Click **Add Configuration**.
4. Provide the following details to configure the SMTP server:
  - A name to the configuration.
  - The host name and port number of the SMTP server.



**Note:** The default configuration uses a local SMTP server that listens to the default port 25. This SMTP server does not require any authentication.

- The email address of the sender. The email address is displayed in email notifications. The email notifications are sent to the users based on the notification settings.
- The encryption type for secure communications is when the SMTP server is on an external network. You can enable either **STARTTLS** or **SSL/TLS**.



**Note:** The SMTP server for the secure communication listens to the default port 465.

- Login credentials are required if the SMTP server requires authentication.

5. Click **Add**.



**Note:** The SMTP configuration is added after a valid connection is established with the SMTP server.

### Results

You configured an SMTP server to connect with HCL OneTest™ Server and send out email notifications about the server events to users who have subscribed to the SMTP channel.

### What to do next

You can modify the default subscriptions for the SMTP channel, see [Managing email notifications on page 617](#).

---

#### Related information

[Managing notifications on page 611](#)

# Chapter 8. Reference Guide

This guide describes conceptual information about data design environment components when you want to use the data fabrication feature in HCL OneTest™ Server.

## Files

You add a file to a project to use it as input in the project. You can upload a file from your local file system to a project from the **Files** tab to generate test data. Alternatively, you can upload it when you define an action.

You also can download a file from a project to your local file system. From your local file system, you can upload the file to a different project.

## Connections

A connection defines the properties that are required to connect to an external resource. An action uses a connection to provide source data to a project or write data from a source to a target. Multiple actions can use the same connection. You can define both connections and actions at the project level.

### Project-level connections

You can reuse project-level artifacts in multiple projects. You use the **Connections** tab to define a new project-level connection and its actions or to select an existing project-level connection and its actions.

For example, you can define a project-level connection to a database that is used as a data source or target of multiple projects. A project-level action might be an SQL query and schema that multiple maps use to look up the same database data.

### Connection testing

You can test project-level connections. When the Design Server has access to an external resource, testing validates the properties of the resource connection. If the Design Server can not access the external resource, the connection test fails, even if the connection properties are correct.

## Resource adapters

HCL® OneTest™ Data resource adapters are used to retrieve data. These resource adapters provide access to the databases.

The following resource adapters are supported by the HCL® OneTest™ Data

- Java Database Connectivity (JDBC) adapter
- Excel adapter
- Systems Applications and Products in Data Processing (SAP) adapter

---

## Related information

[HCL OneTest Data support to write the generated test data in JDBC-supported databases on page 191](#)

## SAP applications

This topic provides an information about integration of SAP applications with HCL® OneTest™ Data.

### Introduction

The Systems Applications and Products in Data Processing (SAP) applications acts as resource adapters. HCL® OneTest™ Data integrates with the SAP application to generate a schema, and then generate the test data to perform application testing.

### SAP applications overview

The SAP applications is a unique data transformation technology for interfacing SAP R/3 software applications with third-party applications and legacy systems. Connectivity to the SAP R/3 environment is provided for both inbound and outbound data, and is supported on the major SAP platforms.

Pack for SAP Applications makes use of SAP's synchronous and asynchronous (transactional) Remote Function Calls (RFCs). This approach ensures effective and secure communication between external systems and SAP R/3. Detailed knowledge of RFCs is not required to create interfaces for SAP R/3 that are reliable, efficient, and easy to maintain. Some knowledge of Remote Communication Basis is helpful but not required.

In addition, the pack can be used with several file-based interfaces, such as SAP DXOB/DMI and EDI subsystems.

The Pack for SAP Applications is used for:

- Initial data load
- Ongoing real-time or near-real time interfaces



**Note:** The Pack for SAP Applications is applicable to the following versions: SAP R/3 4.6, 4.7, mySAP 2004 (ECC 5.0) and mySAP 2005 (ECC 6.0).

### Overview of SAP R/3 interfaces

The Pack for SAP Applications supports the following interfaces and technologies that include the automatic generation of metadata-specific schemas:

- Intermediate Documents
  - Application Link Enabling (ALE)
  - ALE Message Handler (AMS)
  - Electronic Data Interchange (EDI)
- DXOB/Data Migration Interface (DMI)
- For the adapter-specific commands, see the [Adapter Commands List on page 627](#).

## Intermediate Documents (IDocs)

The IDocs interface consists of the definition of a data structure and the processing logic for this data structure. This interface is used to exchange business data between two different systems. SAP R/3 support for the IDoc interface includes both Electronic Data Interchange (EDI) and Application Link Enabling (ALE) technologies.

## Application Link Enabling (ALE)

ALE links multiple SAP R/3 systems in distributed environments, and non-SAP systems to SAP R/3 without file-to-file transfers. The SAP R/3 architecture supports the transfer of data between applications using messaging as opposed to files. SAP R/3 enables the creation of a distributed, fully integrated suite of loosely coupled applications.

The SAP component supports the integration of distributed SAP R/3 systems under ALE and the flow of information among multiple SAP R/3 systems to accomplish specific business objectives. It also embraces integration among non-SAP R/3 applications, such as third-party applications or legacy systems, by providing the conversion technology required to accomplish this level of integration among diverse systems.

The ALE adapter (JALE) adds Unicode character set data support inbound to and outbound from SAP.

## ALE Message Handler (AMS)

The AMS interface provides connectivity between SAP R/3 and one or more disparate applications sending and receiving IDocs for one or more SAP R/3 instances. AMS does not provide any conversion function and delivers IDocs without changing them. Transactional RFC must be the communication method used between the SAP R/3 system and the ALE Message Handler.

## Electronic Data Interchange (EDI)

EDI enables companies with the ability of communicating business transactions and documents electronically with their partners, vendors, and clients.

## DXOB/Data Migration Interface (DMI)

DXOBs are SAP R/3 business objects that can be transferred automatically into the SAP R/3 system. Examples of business objects are organizational units.

The Pack for SAP Applications support for the DXOB interface includes data transfer object structures in both beta and released formats. During data transfer, data is transferred from an external system into the SAP R/3 system.

SAP applications support the data transfer of numerous SAP business objects. The data transfer program specifies the data format definition that is necessary to import the data into the SAP R/3 system. The pack provides the data transformation server to convert the external system data format to the format of the DX object.

## Business Application Programming Interface (BAPI)

The BAPI adapter (JBAPI) enables full Unicode support when installed with the latest version of ITX. The JBAPI adapter provides unified access to both Unicode and non-Unicode systems.

BAPI provides a business-oriented interface for accessing SAP R/3 business processes and data from external systems. BAPIs are part of the SAP business framework, an open component-based architecture enabling software components from SAP and other providers to be integrated.

## The R/3 adapters

The SAP specific components for the SAP R/3 runtime environment are the R/3 adapters for Windows™ based and UNIX™ based platforms.

This section introduces the R/3 adapters and provides instructions for adapter use, adapter settings, and adapter commands, including the following:

## Overview of the R/3 adapters

Both client and server functionality is supported within an R/3 adapter. An R/3 adapter contains the Remote Function Calls (RFCs) necessary to pass inbound and outbound data to and from SAP R/3. The adapter includes transparent integration of the RFCs so that it is not necessary to possess detailed knowledge about the RFCs. The adapters also manage transactional information such as message transaction IDs (TIDs) as needed. This feature ensures seamless integration with SAP R/3 and also provides a robust auditing mechanism.

The R/3 adapters used to interface with the SAP R/3 application are:

- JALE
- JBAPI

## Adapter commands list

The following table lists each R/3 adapter command, the command syntax, and the interfaces for which it can be used for a source (outbound data), target (inbound data), or both.

**Table 10. Adapter command list**

Name	Command Syntax	JALE	JBAPI
Program ID	-A <i>pgm_id</i>	source	
Audit	-AR3[+][U] [%tid% <i>full_path</i> ]	source/target	
Backup	-B[I][X] [%tid% <i>full_path</i> ]	source/target	
Load Balancing	-BAL	target	target
Bytes Per Character	-BPC	source/target	
Client Number	-C <i>clnt_num</i>	source/target	target
Character set encoding	-enc	source	
Destination	-D <i>dest_key</i>	source/target	target
GatewayHost	-G <i>gtwy_name</i>	source	
IDoc Field Generation	-GEN [0 !][ <i>flds</i> ]	target	
Group	-GROUP <i>NAME</i>	target	target
Host ID	-H <i>host_name</i>	source/target	target
Logon Language	-L <i>lang_cd</i>	target	target
Listen	-LSN {0  <i>dur</i> [: <i>int</i> ]}	source	
Listener Threads	-N	source	
Password	-P <i>pwd</i>	target	target
Packet Size	-PKT <i>IDoc_qty</i>	target	
Reprocess backup files	-R	source	
System ID	-S <i>sys_num</i>	source/target	target
Timeout	-timeout <i>secs</i>		target
Trace	-T[V E N][+][ <i>full_path</i> ]	source/target	target



**Table 10. Adapter command list (continued)**

Name	Command Syntax	JALE	JBAPI
Transaction ID	<code>-TID trans_ID</code>	source/target	
IDoc Type	<code>-TY OTHER\$   doc_type*</code>	source	
User ID	<code>-U usr_id</code>	target	target
Gateway Service	<code>v</code>	source	

## Intermediate Documents (IDocs)

IDocs are SAP standard data containers or formats that provide the basis for both ALE and EDI interfaces to SAP R/3. The topics below discuss using IDocs with SAP R/3 and provides the specific information required for ALE and EDI interfaces:

### Overview of IDocs

Several hundred IDocs are shipped with each SAP R/3 system. ALE and EDI interfaces use functionally equivalent IDocs; however, they differ in the manner in which the IDocs are communicated to or from SAP R/3.

- ALE IDocs are communicated through memory buffers, without intermediate files, directly to or from an RFC port using transactional remote function calls (RFCs).
- EDI IDocs are passed using an intermediate file.

These interfaces (ALE and EDI) that use the IDoc structure represent SAP's strategic approach to interfacing SAP R/3 with legacy and third-party applications where loose coupling is appropriate. Therefore, the SAP R/3 IDoc approach should be considered the first choice for developing interfaces where asynchronous, near-real time, or batch links are required. This pack supports SAP-supplied IDocs as well as user-defined IDocs.

One or more IDocs are held in a container called a packet. A packet is the largest data container in an IDoc schema. The restart attribute is automatically added to reject invalid IDocs contained within a packet.

### IDoc structural format

The IDoc format consists of hierarchically structured segments with multiple levels of nesting. IDocs exist for data such as customers and materials, for transactional data such as sales orders, and for control data such as company codes. In ALE terms, these are called IDoc types, which are described as follows:

#### Control record

There is one control record, and its format is identical for all IDoc types. It consists of the IDoc ID, Sender ID, Receiver ID, IDoc type, and EDI attribute.

### Data records

There can be one or more data records. A data record consists of a fixed administration section (IDoc ID, Sequence/Hierarchy) and a data part (Segment). A Segment is the formal definition for header data and item data. The number and format of the segments can be different for each IDoc type.

### Status records

The status records describe the processing stages through which an IDoc can pass, and contain the IDoc ID and status information. There will be several status records.

The IDoc types are assigned to release-independent message types, which themselves are assigned to object types in the Business Object Repository (BOR). Customer enhancements are generally subject to naming conventions. IDoc data segments are a maximum of 1000 bytes in length and generally only contain CHAR fields. The data segments of an IDoc type are described by release-independent segment types and have release-specific segment definitions that are stored as internal structures in the Dictionary. This means IDocs can be sent with different data contents to different recipients because the recipient's release is defined in the sending system.

## Data Transfer Objects (DXOB)

Data transfer objects (DXOB) are SAP R/3 business objects that can be transferred into an SAP R/3 system. To use SAP R/3 to create DXOB interfaces including the creation of initial data transfer files (DXOB reports) and using Connections and Actions to generate the corresponding schemas read the sections below:

## Business Application Programming Interface (BAPI)

The BAPI adapter (JBAPI) enables full Unicode support when installed with the latest version of ITX. The JBAPI adapter provides unified access to both Unicode and non-Unicode systems.

BAPI provides a business-oriented interface for accessing SAP R/3 business processes and data from external systems. BAPIs are part of the SAP business framework, an open component-based architecture enabling software components from SAP and other providers to be integrated.

## Overview of the Interface (BAPI)

SAP Business Objects reside in the Business Framework and provide the interoperability of software components. Collectively, Business Objects are contained within the Business Object Repository (BOR). They cover a wide range of SAP R/3 business data and processes. BAPIs allow external non-SAP R/3 systems, such as HCL® OneTest™ Data, to access the methods on a SAP Business Object by using appropriate adapters. SAP Business Objects and their BAPIs provide an object-oriented view of SAP R/3 functionality.

BAPIs are accessed in SAP R/3 using the transaction code BAPI.

*Vendor* is a Business Object in the Financial Accounting area of Business Framework. Expanding the Vendor Business Object exposes the available methods.

ChangePassword is a method (BAPI) on the Vendor Business Object. This method provides the capability to change a vendor's password, which is selected by the key field **VendorNo**. The BAPI name of this method is found by clicking

the **Detail** tab. The BAPI name is BAPI\_VENDOR\_CHANGE\_PASSWORD, the name you specify in Connections and Actions.

The dialog box also provides documentation on the parameters and attributes of the BAPI. All BAPIs are composed of a combination of Importing, Exporting, and/or Table parameters. Importing parameters are input, Exporting are output, and Tables are in/out. A BAPI may have any combination of these parameter types. Documentation for the parameters, structures, and scalars of a BAPI can be found in the BAPI Browser, the SAP Assistant, and the Function Module.

## Schema design

A schema is a graphical data dictionary that contains metadata definitions of the structure of a document to generate test data. Schema provides visual cues for understanding the structure and complete definition of an object.

Schema designer helps to:

- Create and manage schema that define properties for data structure
- Create data validation rules

## Introduction to schema design

Use the **Schemas** tab to define, modify, and view schemas. A schema describes the syntax, structure, and semantics of your data.

The syntax of data refers to its format including tags, delimiters, terminators, and other characters that separate or identify sections of data. The structure of data refers to its composition including repeating substructures and nested groupings. The semantics of data refer to the meaning of the data including rules for data values, relationships among parts of a large data object, and error detection and recovery.

## About schemas

A schema defines the contents for generating the test data. A schema is a mechanism for defining each element of your data. Similar to a data dictionary, a schema contains a collection of type definitions.

Because data definitions are defined in a schema, you must be familiar with the specifications that define your data before attempting to create one.

A data file is a simple example. The data file is made up of records and each record is made up of fields. In this case, there are three kinds of data objects: a file, a record, and a field.

In a file of records, think of the data in terms of the three data objects. For example, one type defines the entire file; another type defines the entire record contained in that file. Other types in the same schema define the data fields of the record.

A schema has three data type classifications: group, category, and item. The *root* type is the base type from which all other types stem, representing the data objects of all types in the schema. *Root* is the default name, however, you can modify any type name.

Types within a schema are listed in alphabetical order by default. To change the order for new types, modify the root type properties.

## Subtypes

Subtypes are created for a number of reasons like identifying distinct data object. Another reason is that specific types of an object may have different properties such as different date formats.

Think of subtypes as different "flavors". Ice cream flavors can be chocolate, strawberry, or vanilla. To create a schema to represent ice cream as data, the different flavors of ice cream could be subtypes of the type **IceCream**.

The type **IceCream** is generic and describes any kind of ice cream. The type **Chocolate**, a subtype of **IceCream**, represents a certain kind of ice cream: chocolate.

A file of purchase order data may have two different kinds of records: header and detail. The schema representing this data might have a **Record** type with **Detail** and **Header** as subtypes of **Record**.

## Schema hierarchy

The types in a schema are arranged in a hierarchy. If a type is subordinate to another type, it is called a *subtype*. The type on the branch stemming above a specific type is called its *supertype*.

Subtypes have more specific properties. For example, two different kinds of fields, **Name** and **Date**, may be defined as subtypes of **Field**.

The item type **Field** describes any field. The item types **Date** and **Name** are more specific kinds of fields. In a classification hierarchy such as this, the deeper the subtype is in the type tree, the more specific the data characteristics.

### Example scenario

Imagine that a schema represents your house. There is a type for the entire house (**House**). A type for each room (**Bathroom**, **Bedroom**, and so on), and types for the different furnishings in these rooms (**Bed**, **Chair**, and so on). Each type represents a complete object. A house is made up of rooms. Inside these rooms are beds, chairs, couches, and so on. You know this, but you cannot see it by looking at the classification hierarchy view of the schema. You must open one of the types in the group view to see its components.

## Defining data

For optimum test data generation, the entire contents of your data must be defined. Using the Type Designer, you can define input data so that each data object of the source data is identified. You can define the output data according to your output specifications.

How specifically you define the data is up to you. For example, if there is a large section of data that you want to process quickly, define it loosely as a chunk of text.

The purpose of the Type Designer is to define your data.

## Objects, types, and classes

Schemas consist of objects, types, and classes.

### Objects

A data object is a complete unit that exists in your input or is built on output. A data object can be simple (such as a date) or complex (such as a purchase order).

A data object is some portion of data in a data stream that can be recognized as belonging to a specific type. When you create types, identify all of the objects that make up the data: input objects and output objects.

### Types

A type defines a set of data objects that have the same characteristics. For example, the type "Date" can be defined as representing data objects in the form MM-DD-YY. The type "CustomerRecord" can be defined as representing data objects, each of which consists of a **Company**, **Address**, and **Phone** data object.

### Classes

A type is classified according to whether or not it consists of other objects. Each type in a schema must be defined in one of three classes: item, group, or category.

- **Item type**

An item type represents a simple data object that does not consist of other objects. It is a default class of schema.

- **Group type**

A group type represents a complex data object that consists of other objects. For example, "FullName" is a group type that contains the components: FirstName, MiddleInitial, and LastName. Each group must have at least one item type.

- **Category type**

A category type is used for inheritance and for organizing other types in a schema. For example, you might have a category named "OrderField" to organize the different kinds of order fields in your schema.

## Type properties

The properties of a type define the characteristics of the data objects of that type.

For *item* types, properties define whether that item is text, a number, a date and time, or a syntax value. Properties include such characteristics as size, pad characters, and justification.

For *group* types, the properties are related to the format of that group. The format of a group may be explicit or implicit. In addition, type properties include syntax objects that appear at the beginning or end of the object, as well as release characters.

## Defining type properties

### About this task

You can define specific properties for each type. Type properties generally consist of the following:

- Definition of a type's **Class**, **Group** or **Item** properties
- Type **Initiator**, **Terminator**, and **Release** character
- A list of where a type is used in the definitions of other types

## Basic type properties

The following type properties are common to categories, groups, and items:

- [Name on page 634](#)
- [Class on page 635](#)
- [Description on page 636](#)
- [Intent on page 636](#)
- [Partitioned on page 636](#)
- [Order Subtypes on page 636](#)
- [Initiator on page 636](#)
- [Terminator on page 637](#)
- [Release Characters on page 637](#)
- [Empty on page 639](#)
- [National Language on page 658](#)
- [Document Type on page 640](#)
- [Where Used on page 640](#)

Item-specific properties are discussed in more detail in "[Item Properties](#)" on page 643. Group-specific properties are discussed in more detail in "[Group Properties](#)" on page 681.

## Name

The name of the type should be as descriptive as possible and reflect the information that the type represents.

A type name must confirm to the following guidelines:

- A type name cannot be more than 256 characters long.
- A type name cannot be a reserved word or contain a reserved symbol. For a list of reserved words, refer to Design Studio Introduction documentation.
- A type name cannot contain only digits and/or periods.
- A type name may contain numbers and special characters.
- A type name cannot contain spaces. Use an underscore instead.
- A type name can contain the following:
  - Letters
  - Digits
  - ASCII characters 128-255
  - Special characters: #, ~, %, \_ , ? or \
  - Double-byte characters (Japan edition only)
- A type name cannot have the same name as another type on the same level in the same schema.

A type name can be up to 256 characters in length, but use a short type name if possible. The full path name of each type is used in map rules in the map designer.

## Class

The class of a type describes whether the type represents a simple data object (item), a complex data object (group), or whether is used to organize types (category). Define the class of the selected type by selecting the check box.

### Category

Categories organize types that have common properties. Each category has a set of item properties and a set of group properties. Subtypes of the category inherit these properties.

A category does not define data objects in detail. A category type may be a component of a partitioned group, after a component with the identifier attribute. A category cannot be a component of a non-partitioned group.

### Item

The item type represents a simple data object that does not consist of any objects. An item does not have components.

### Group

The group type represents a complex data object that consists of components.

To change a type's class, select the type and choose the desired class in the **Properties** window.

If you change a type to an item, all the types in its subtree become items. If you change a type to a group, all the types in its subtree become groups. This is because the schema has a classification hierarchy and all the nested subtypes of an item must be items. Similarly, all the nested subtypes of a group must be groups.

## Description

Use this property to record a brief description of the type. The description entered is for informational purposes only. It is not used for identifying data objects at runtime.

It is good practice to add a meaningful description of the type when you are defining it.

## Intent

Indicates whether the type is a general type or an XML type.

The type can only be an XML type if the type tree was created using the XML DTD Importer or the XML Schema Importer.

Refer to [“XML Properties in the Type Tree” on page 690](#) for more information about type properties created from XML.

## Partitioned

If the data of this type can be divided into mutually exclusive subtypes, it can be partitioned. For information about partitioning, see [“Partitioning” on page 705](#).

### **Yes**

Partitioning is enabled for this type.

### **No**

Partitioning is not enabled for this type.

## Order subtypes

Choose the method in which the subtypes of this type will be added or viewed in the schema.

### **Ascending**

Add and view subtypes of this type in alphabetic or numeric order.

### **Descending**

Add and view subtypes of this type in reverse alphabetic or numeric order.

### **User-defined**

Add subtypes of this type to manually reorder the subtypes.

## Initiator

An initiator is a syntax object that appears at the beginning of a data object. Defining an initiator for a type specifies that when data of that type appears, the initiator appears at the beginning of the data object. The initiator becomes part of the data type definition.



**None**

There is no initiator.

**Literal**

The initiator is literal. Expand the **Initiator** property to enter the literal initiator value and data language of the initiator value.

**Variable**

Allow for possible values. Expand the **Initiator** property to define the variable terminator **Default**, **Item**, and **Find** properties.

If each record begins with an asterisk \*, define the \* as a literal initiator of the record type.

The following data represents the classes in a college English department. An asterisk \* is displayed at the beginning of each **ClassRecord**.

For information about other symbols used in the **Value** field of the **Initiator** property, see the Type Tree Importer documentation.

## Terminator

A terminator is a syntax object that appears at the end of a data object. The terminator becomes part of the data type definition.

**None**

There are no terminators.

**Literal**

A constant value. Expand the **Terminator** property to define the literal terminator **Value**.

**Variable**

Allow for possible values. Expand the **Terminator** property to define the variable terminator **Default**, **Item**, and **Find** properties.

For example, a carriage return/linefeed (`CR/LF`) at the end of a record is the record's terminator.

Generally, if the data ends with a given syntax object, you should define a literal terminator. For example, it is very common to define a `CR/LF` as a terminator of a record when you know that the record always ends with a `CR/LF`, regardless of where the record appears.

## Release characters

A release character is a one-byte character in your data indicating that the character(s) following it should be interpreted as data, not as a syntax object. The release character is not treated as data, but the data that follows it is treated as actual data.

## Building release characters for output data

If a release character is defined for a type, a release character is inserted for each occurrence of a syntax object in the data of any item contained in that type.

## Guidelines for using release characters

Guidelines for using release characters include the following:

- Release characters apply to character data only, *not* binary data.
- Characters defined as pad characters are not released.
- The maximum size of an item does *not* include the release characters.

## Release character example

The group type **Record** type has a literal delimiter of , and a release character of ?. The group type **Record** has three item components. Data for the record looks like the following:

Miller?, MD,Harkin Hospital,1996

The ? releases the comma after **Miller**.

In the first field, the actual data value is **Miller, MD**. Because the comma appears as part of the data, it is necessary to have the release character ?, which indicates that the , following it is data, *not* a delimiter. This data would be interpreted as the following:

Data for component #1: `Miller, MD`

Data for component #2: `Harkin Hospital`

Data for component #3: `1996`

A release character can apply to a delimiter, a terminator, or even the release character itself.

If a release character appears in the data and it is not followed by a syntax item, the release character is ignored.

## Building release characters for output data

If a release character is defined for a type, a release character is inserted for each occurrence of a syntax object in the data of any item contained in that type.

## Guidelines for using release characters

Guidelines for using release characters include the following:

- Release characters apply to character data only, *not* binary data.
- Characters defined as pad characters are not released.
- The maximum size of an item does *not* include the release characters.

## Release character example

The group type **Record** type has a literal delimiter of , and a release character of ?. The group type **Record** has three item components. Data for the record looks like the following:

```
Miller?, MD,Harkin Hospital,1996
```

The ? releases the comma after **Miller**.

In the first field, the actual data value is **Miller, MD**. Because the comma appears as part of the data, it is necessary to have the release character ?, which indicates that the , following it is data, *not* a delimiter. This data would be interpreted as the following:

```
Data for component #1: Miller, MD
```

```
Data for component #2: Harkin Hospital
```

```
Data for component #3: 1996
```

A release character can apply to a delimiter, a terminator, or even the release character itself.

If a release character appears in the data and it is not followed by a syntax item, the release character is ignored.

## Empty

The **Empty** property provides alternative type syntax for groups or items when they have no data content.

When the **Empty** property is specified for a type and there is no data content, the **Empty** syntax is displayed. For example, this can be used for XML data that contains either start and end tags or an empty tag.

You can use the **Empty** type property instead of syntax object items to potentially improve the schema runtime processing time (during data validation).

The following options are available for the **Empty** property.

### None

Default setting. Select **None** if you do not have an initiator, terminator, or release character.

### Literal

A constant value. When **Literal** is selected, the **Value**, **Ignore Case**, **Required**, and **National language** sub-fields become available. You can use only one literal to indicate a zero-length data item.

## Empty type property example

Consider an XML element called `Comment`. Presuming that this element has a simple content of an arbitrary length, it can be represented in a schema as a text item, with the initiator value `<Comment>` and the terminator value `</Comment>`.

This item can then be used to validate the following data:

```
<Comment>Some comment...</Comment>
```

The resulting value for the item will be:

```
Some comment...
```

As another example, the following data is also successfully validated:

```
<Comment></Comment>
```

The resulting value for the item will be a zero-length string.

The problem occurs with the following data: `<Comment />`

From the XML perspective, this data is equivalent to the `<Comment></Comment>` data shown above, as it represents the **Empty** element `Comment`. However, from the schema perspective, this data is invalid because it does not start with `<Comment>` and does not end with `</Comment>` as required by the initiator and terminator item property values.

This is where the **Empty** property is useful. By defining the **Empty** property for the item to have a value of `<Comment />`, it will be possible to validate the data `<Comment />`.

Even if the data does not match the syntax described by the initiator and terminator properties, it will be validated because it will match the **Empty** property value. It will be treated as an *empty item*, that is, the resulting value for the item will be a zero-length string, similar to the example shown previously.

## Document Type

The **Document Type** setting is a component of Document Verification. A requirement of using Document Verification is to set the **Document Type** properties of the associated schema object.

To use the Document Verification option for XML documents, you must do the following:

- Set the **Document Type** property to **XML**.
- Set the **Document Type Metadata** value to **Schema** or **DTD**.
- Specify the **Location** of the DTD or Schema file.

### Default

This is the default value, which indicates a non-XML document type.

### XML

Indicates an XML document type. When this value is set to **XML** and the appropriate **DocumentVerification** map settings are in place, the XML document will be validated by an external program in addition to the standard data validation process.

See the map designer documentation for information about using the Document Verification option.

## Where used

**Where Used** shows how and where the type is used within other types in the tree.

## Symmetric swapping

When the input and output objects have different symmetric swapping characteristics, the Symmetric Swapping property is considered "on" for the related mapping rule.

### Examples

Original text	Resulting text
<code>abcF(E)Dghi</code>	<code>abcF(E)Dghi</code>

#### Input Object

No swapping

#### Output Object

Swapping

Original text	Resulting text
<code>abcF(E)Dghi</code>	<code>abcF(E)Dghi</code>

#### Input Object

Swapping

#### Output Object

No swapping

Original text	Resulting text
<code>abcF(E)Dghi</code>	<code>abcF)E(Dghi</code>

#### Input Object

No swapping

#### Output Object

No swapping

Original text	Resulting text
<code>abcF(E)Dghi</code>	<code>abcF)E(Dghi</code>

#### Input Object

Swapping

#### Output Object

Swapping

## Orientation

*Orientation* is to change characters from one direction to a different direction.

### Example

A simple orientation example is to define English characters "abc" as a bidirectional object with left-to-right (LTR) orientation. If you moved the object to a right-to-left (RTL) object, the output should be "cba". The following example demonstrates a string of characters that change in orientation.

Original text	Resulting text
ABC DEF ã ã; ãç	ãçã;ã FED CBA

#### Input

left-to-right (LTR)

#### Output

right-to-left (RTL)

## Shaping

*Text shaping* is the concept of combining or separating characters where possible. The Arabic usage is with ligatures (such as the LAM, ALEF, and LAM-ALEF characters). For example, in a non-shaped word, both LAM and ALEF characters would appear separately. In a shaped word, the two characters would be replaced with a single LAM-ALEF character.

*Numeric shaping* is the idea that normal regular numerals (0–9) can be shaped to be Arabic-Indic

### Examples

```

ã, = LAM
ï° = ALEF
Û™ï»» = LAM ALEF
    
```

Original text	Resulting text
abcï»»def	abcÛ™ï°def

#### Input

Shaped

#### Output

Unshaped

Original text	Resulting text
abcÜ„i°def	abcï»»def
<b>Input</b>	
Unshaped	
<b>Output</b>	
Shaped	

## Item properties

Item types define data for a selected type. Item types are divided into the following subclasses: **Number**, **Text**, **Date & Time**, and **Syntax**. Each item subclass has a specific set of properties.

## Item subclass

The subclass of an item represents the characteristics of the data. Both category and item types have the **Item Subclass** property.

To define the subclass of an item type:

1. Access the **Properties** for the selected category or item type.
2. For the **Item Subclass** property, select a value that defines the data for the type:

### Number

Any number excluding active syntax objects. The interpretation can be either character or binary. See [“Number Item Subclass properties” on page 644](#) .

### Text

Any character excluding active syntax objects. The interpretation can be either character or binary. See [“Text Item Subclass properties” on page 666](#) .

### Date & Time

A valid date and time format based on the data. The interpretation can be either character or binary. See [Date & Time Item Subclass properties” on page 670](#) .

### Syntax

Syntax objects are used as separators between portions of data. A restricted set of values used to define a dynamic delimiter, initiator, terminator, or release character. The interpretation is character. A syntax object cannot be longer than 120 bytes. Syntax objects, used as syntax, cannot be mapped. See [“Syntax Item Subclass properties” on page 679](#) .

For text and syntax objects, character size constraints can be in bytes or characters. Numbers are considered in digits and date-time formats are sized by the format, such as CCYYMMDD.

## Number item subclass properties

Number item subclass properties can be of **Character** or **Binary** value. When you choose **Number** as the **Item Subclass** value, you must choose value as: **Binary** or **Character**.

The value defines how the data should be interpreted. Items with a subclass of **Number**, **Text**, and **Date & Time** can be interpreted as either character or binary. Items with a subclass of **Syntax** can be interpreted as character only.

## Interpret as binary

Binary text items have content size and pad properties. Binary data is required to be sized or of a fixed size.

Binary number items interpret the data as a binary number or as a byte stream.

Items with an item subclass of **Text** can be interpreted as character or binary.

The binary presentation options are Integer, Float, Packed, or BCD.

## Binary integer presentation

The **Integer** value is a whole number.

### Byte Length

Can have a length of **1, 2, 4, or 8** bytes

### Byte order

See ["Byte order" on page 645](#) .

### Sign

See ["Sign" on page 651](#) .

## Binary float presentation

The **Float** value is a number with decimals in a location as needed.

### Byte Length

Can have a length of **4, 8, or 10** bytes

## Binary packed presentation

The **Packed** value is a number that has size, decimal places, and sign properties.

The binary packed properties include Length, Implied places, and Sign.

### Length

Can have a length of **1 - 16** bytes

### Implied places

Can be from **0 - 31**



**Sign**

Can be **Trailing-** or **Trailing+**

For **Packed** numbers, the **Implied places** cannot exceed the **Length**.

**Binary BCD presentation**

The **BCD** value is a binary coded decimal number that has size.

**Byte Length**

Use the **Byte Length** property to select the number of bytes that equals the length of the item.

The **Byte Length** property is an option when the **Number** item subclass property is interpreted with a **Binary** value.

**Byte order**

Use the **Byte order** property to define the way that bytes in the data are ordered by selecting one of the following values from the Byte order drop-down list.

<b>Value</b>	<b>Description</b>
Big Endian	The most significant byte has the lowest address. (Usually systems such as IBM, HP, and Solaris.)
Little Endian	The least significant byte has the lowest address. (Usually systems such as Intel and VAX.)
Native	The order is dictated by the platform.

**Byte order** is an option when the **Number** item subclass is interpreted with a **Binary** value.

**Interpret as character**

Character number items interpret the data as symbolic data. Symbolic data has the same meaning on different computers. For example, a comma is symbolic because it has the same meaning, regardless of the computer type.

Character text items can have content size and pad properties.

When you select **Character**, the **Release** property is specific to the **Character** value and is *not* available as an option for a **Binary** value. For more information, see ["Release characters" on page 637](#).

When interpret is set to **Character**, integer, decimal, and zoned presentation options are available.

**Character integer presentation**

The **Integer** value is a whole number.

Further information on the **Integer** value is found in ["Integer Separators" on page 647](#).

## Character decimal presentation

The **Decimal** value is a number that contains decimals.

More information about the **Decimal** value is found in section ["Decimal Separators" on page 649](#) .

## Zoned character presentation

The **Zoned** value is a number that has a size, decimal place, pad, and sign properties.

Use the "Size (digits)" property to specify the size of the zoned number. The size of a zoned number can also be specified in terms of minimum and maximum. The minimum size must be less than or equal to the maximum size.

When the content size of a character-zoned number is used, the last digit and the sign are combined into the same digit but the content size does *not* include the initiator, terminator, release characters, or pad characters.

When the content size of a character-zoned number is important (for example, when the SIZE function is used on the character zoned number), the content size includes the sign but does not include the initiator, terminator, release characters, or pad characters.

Use the "Sign" [on page 651](#) property to define whether the zoned number will be signed. The default value is **No**.

The size of a zoned number is specified in terms of the minimum and maximum number of digits. If the size is specified with the **Min** and **Max** properties, the sign is *not* included.

### Example

Zoned Number Not Signed	Signed Zoned Number
1230	123{
1231	123A
(Length = 4 digits)	(Length = 4 digits)

## Places > implied

Numbers have implied decimal places. The number of decimal places cannot exceed the length specification. The list of available decimal places varies with the length selected. In the **Value** column, enter a total number of decimal places (not to exceed 31).

## Size (digits)

Use the Size (digits) property to specify the minimum and maximum size of a number item.

### Min

The minimum number of digits of the data object.

The default value is 0.

**Max**

The maximum number of digits of the data object.

If there is no maximum size, a **Max** value is not required.

**Excluded from min and max size**

For character number items, there are certain objects excluded from the **Min** or **Max** count. For each **Presentation** option, the following table lists what is excluded from the minimum or maximum **Size (content)** count.

<b>Presentation:</b>	<b>Integer</b>	<b>Decimal</b>	<b>Zoned</b>
	initiators	initiators	initiators
	pad characters	pad characters	pad characters
	release characters	release characters	release characters
	separators	separators	terminators
	signs	signs	
	terminators	terminators	

**Size example**

In the following ASCII number there are a total of 12 characters:

```
+1234567.999
```

However, because the initiator (+) and separator (.) are not counted, the number would validate for a maximum size of 10 (**Max=10**).

**Separators**

Integer and decimal number items can have a separator for thousands and fractional places. If you opt to have a separator, choose **Yes** and sub-options will appear that enable you to choose the format and syntax.

The **Zoned** value of the **Presentation** property does not have separators.

**Integer separators**

For integer numbers, the separator is the thousands separator.

Integer character numbers have separator properties of:

- Format
- 1000's Syntax
- Value

These properties specify the placement and value of the thousands separator.

## Separator > format

Use the **Separators Format** property to specify the placement and value of the thousands separator.

To define the separator format, select one of the separator formats from the drop-down list in the **Value** column.

**#[.]####**

The thousands separator is not required on input, but if present, it must be in the proper location. The separator is not built on output.

**#.####**

The thousands separator is required on input in the proper location. The separator is built on output.

## 1000's syntax > value

Define the thousands separator as either literal or variable. Select one of the following from the drop-down list in the **Value** column:

### **Literal**

The thousands syntax is a constant literal specified in the **1000's Syntax > Value** column.

### **Variable**

The thousands syntax allows for variable values. Use the **1000's Syntax > Default, Item, and Find** properties to define the variable fraction separator.

## 1000's syntax (literal) > value

For the literal thousands separator, define the literal separator by typing the literal separator character in the **Value** column or by clicking the browse button to display the **Symbols** dialog box from which you can insert any non-printable value.

A separator can be from one to 120 bytes in length. A separator cannot start or end with a digit and cannot be the ? character. The ? character is a reserved character that can be used as a wildcard to represent any single valid character.

## 1000's syntax (variable) > default

Use the **1000's Syntax > Default** property to define the default variable separator value for the thousands place. The default literal separator value for thousands syntax is a comma.

Enter the default variable separator character or click the browse button to display the **Symbols** dialog box in which you can insert any non-printable value.

## 1000's syntax (variable) > item

For integer numbers with a variable thousands separator, the **Separator** property is **Yes**, and the **1000's Syntax** is **Variable**, you can specify an item type for the variable thousands separator.

You can select the desired syntax item from the drop-down list. Or, with the focus on the **Item** property (in the **Value** column), press **Alt** and drag the default separator type item from the schema editor into the **Value** column.

An item type with a class of **Syntax** must exist in the schema to be a valid **1000's Syntax Item**.

## 1000's syntax (variable) > find

For integer numbers with a variable thousands separator, the **Separator** property is **Yes** and the **1000's Syntax** is **Variable**. The value of the separator can be the current value or the value of the separator can be determined each time an occurrence of that type is found.

### Yes

Determine the value of the separator each time an occurrence of that type is found. After the value of that separator is found, that particular value is used until it is reset either by another Find or by the occurrence of that separator as a component.

### No

The system uses the value to which the separator item is currently set or, if not set, it uses the default value.

## Decimal separators

For decimal numbers, the separator is the thousands separator and the fractional separator.

Decimal character numbers have separator properties of:

- Format
- 1000's Syntax
- Fraction syntax
- Value

These properties specify the placement and value of the fractional separator.

The item is assumed to have an implicit number of decimal places. To specify the number of implied decimal places, use the **Item Subclass Places** property to specify the number of implied decimal places.

## Separators > format

Use the **Separators Format** property to specify the placement and value of the fraction separator.

To define the separator format, select one of the separator formats from the drop-down list in the **Value** column.

#### ####[.##]

The fractional separator is present if there are fractional digits. There is no thousands separator.

#### ####.##

The fractional separator is always present, even if there are no fractional digits. There is no thousands separator.

#### #[,]###[.##]

The fractional separator is present if there are fractional digits. There is an optional thousands separator.

#### #[,]###.##

The fractional separator is always present. There is an optional thousands separator.

#### #,###.##

The fractional separator is always present. There is always a thousands separator if there are more than three whole number digits.

## Separators > 1000's syntax

Use the **Separators 1000's Syntax** to define the thousands syntax. Select one of the following from the drop-down list in the **Value** column:

### Literal

The thousands syntax is a constant literal specified in the **1000's Syntax > Value** column.

### Variable

The thousands syntax allows for variable values. Use the **1000's Syntax > Default, Item, and Find** properties to define the variable fraction separator.

See the following topics for additional information:

- ["1000's Syntax \(Literal\) > Value" on page 648](#)
- ["1000's Syntax \(Variable\) > Default" on page 648](#)
- ["1000's Syntax \(Variable\) > Item" on page 649](#)
- ["1000's Syntax \(Variable\) > Find" on page 649](#)

## Separators > fraction syntax

Use the **Fraction syntax** property to define the fraction syntax as either a constant literal value or a variable value.

You can define the **Fraction syntax** as literal or variable.

### Literal

A constant value.

Use the **Fraction syntax > Value** property to select a literal fraction separator from the **Symbols** dialog box.

### Variable

Allow for variable fraction separator values. The following options are available to define the variable fraction separator:

- **Default** - Use this property to define the default variable separator value for fractions. The default literal separator value for fractions is a period. Enter the default variable separator character or click the browse button to display the **Symbols** dialog box in which you can insert any non-printable value.
- **Item** - For decimal numbers with a variable fractional separator, the **Separator** property is **Yes** and the **Fraction syntax** is **Variable**, you can specify an item type for the variable fraction separator.



**Note:** An item type with a class of **Syntax** must exist in the schema to be a valid **Fraction syntax Item**.

- **Find** - For decimal numbers with a variable fraction syntax, the **Separator** property is **Yes** and the **Fraction Syntax** is **Variable**, the value of the separator can be either the current value or the value of the separator can be determined each time an occurrence of that type is found. Select an option:
  - **Yes** - Determines the value of the separator each time an occurrence of that type is found. After the value of that separator is found, that particular value is used until it is reset by another find or by the occurrence of that separator as a component.
  - **No** - The system uses either the value that the separator item is set to currently, or, if it is not set, uses the default value.

### Sign

Use the **Item Subclass Sign** property to define whether the number is signed for either the **Integer** or **Decimal** value. A sign is a symbol that identifies a number as being either positive or negative. A positive sign is plus (+); a negative sign is negative (-).

#### Yes

The number is signed. Expands the **Sign** property to define the sign values. If the **Sign** property is **Yes**, a minimum of at least one sign value (**If number is +**, **If number is -**, or **If number is 0**) must be specified and at least one value must be required as input.

#### No

The number is not signed. This is the default setting.

When you select **Yes**, you can further define the sign values. The following sub-options are available:

### **Leading-**

The sign precedes the number when it is negative only. For input, a sign is required for negative data, but is optional for positive data.

### **Trailing-**

The sign follows the number when it is negative only. For input, a sign is required for negative data, but is optional for positive data.

### **Leading+**

The sign always precedes the number. A sign is required for input and output data.

### **Trailing+**

The sign always follows the number. A sign is required for input and output data.

### **Custom**

Defaults to **Leading-** but can be changed.

**Sign** values may be specified for the following:

- **If Number is +** (positive numbers)
- **If Number is -** (negative numbers)
- **If Number is 0** (value of zero)

For each value (**If number is +**, **If number is -**, or **If number is 0**), specify the following:

#### **Leading sign**

Enter the symbol to be placed before a number.

#### **Trailing sign**

Enter the symbol to be placed after a number.

#### **Required on input**

If the number has a sign, at least one value (**If number is +**, **If number is -**, or **If number is 0**) must be required on input. Select an option from the drop-down list:

##### **Yes**

The sign is mandatory on input.

##### **No**

The sign is optional on input.

## **Pad**

Pad implementation determines how the pad value is sized during validation and how the pad value is sized when written as output.



When the data value of the target item is smaller than the minimum length of that item, pad characters can be used to pad the data to that minimum length. Output data is built according to the pad definitions of the types. Bytes are the default measurement for sizing, however, both byte and character sizing options are available.

#### Yes

Enables the **Pad** option. Allows the item to contain both content and pad characters.

#### No

All data is assumed to be content on input; no pad characters are built on output.

## Pad > value

Use the **Pad Value** property to define a 1-character pad character. The default pad value is 0.

Type the pad character symbol between angled brackets < > in the **Value** field or click the browse button to display the **Symbols** dialog box to insert any non-printable value.

For example, to enter a **FormFeed** value for the pad character, in the **Value** field enter <FF> or click the browse button to select the **FF (FormFeed)** symbol from the **Symbols** dialog box.

## Pad > Padded to

Use the **Pad > Padded to** property to define whether the data item is padded to a fixed size or to the minimum content size defined for the data object.

#### Fixed size

The data object is padded to a fixed size (in bytes or characters) that you specify in the **Length** field.

For any item padded to a fixed size, the item must have a value specified for the **Size (content) > Max** property and the **Padded to > Length** value must be greater than or equal to the **Size (content) > Max** value.

#### Min Content

The data is padded to the value specified as the minimum size (in the **Size (content) > Min** field).

You can further specify whether to count pad characters toward the length using the **Counts Toward Min Content** property.

## Padded to > length

To pad to a fixed size, you must specify the length.

Input data can contain both content and pad characters but when an item is built for output, the item is padded to the number of bytes or characters specified in this field.

The **Padded to > Length** value must be greater than or equal to the **Size > Max** value.

You can specify the method of measurement (bytes or characters) using the **Sized As** property.

## Padded to > Sized As

Use the **Padded to > Sized As** property to specify that padding is measured in either bytes or characters. The default setting is **Bytes**.

Trace file and audit log lengths are always reported in bytes.

**Table 11. Pad implementation example (bytes compared to characters)**

Scenario for empty field:	Result:
Data language is UTF-8	Two ª pad characters using 4 bytes.
<b>Pad &gt; Value</b> = ª (hexadecimal representation = 0xC3 0x80)	
<b>Pad &gt; Padded to</b> = Fixed Size	
<b>Padded to &gt; Length</b> = 5	
<b>Padded to &gt; Sized As</b> = Bytes	
Data language is UTF-8	Five ª pad characters totaling 10 UTF-8 bytes.
<b>Pad &gt; Value</b> = ª (hexadecimal representation = 0xC3 0x80)	
<b>Pad &gt; Padded to</b> = Fixed Size	
<b>Padded to &gt; Length</b> = 5	
<b>Padded to &gt; Sized As</b> = Characters	

## Padded to > Counts Toward Min Content

When you select **Padded to > Min Content**, the **Counts Toward Min Content** field is displayed. Use this setting to specify if pad characters should count toward the length of an object when determining if it meets its minimum content length.

### Yes

Pad characters are included in the count for the length of an object.

### No

Pad characters are not counted toward the length of an object.

You can use the **Propagate** function here to propagate the present **Counts Toward Min Content** value to the subtypes of the present type, if present.

If you are using the trace option, there might be cases where in the trace file the content length appears to be different than the input length. A trace file lists the input length (the length used to validate an object), which includes the input

length of each object plus the pad characters. In the case of numbers, signs and separators are also counted in the length.

### Counts Toward Min Content > **AcceptAllPads**

The **AcceptAllPads** property is applicable to data objects that contain only pad characters. Use this property when you need an object that contains only pad characters (no content) to either pass or fail validation depending on whether it is a mandatory or an optional type, regardless of the minimum size requirement.

#### Yes

When an object contains only pad characters, the minimum size requirement is ignored and the object passes size validation. The object then passes or fails type validation depending on whether it is a mandatory or an optional type.

#### No

(Default setting) Pad characters do not count toward the minimum size requirement, therefore an object that contains only pad characters and that does not meet the minimum size (content) requirement fails size validation.

### Example

The results of the following examples are based on these values:

#### Size (content)

5

#### Pad

Yes

#### Pad > Padded to

Min Content

#### Padded to > Counts Toward Min Content

No

Default Behavior: When **Accept All Pads** is set to **No**, the following results occur:

#### Input Data (X = pad character)

##### Valid?

ABCDE

Yes

ABC

No

ABCXX

No

**XXXXX**

No

When **Accept All Pads** is set to **Yes**, the following results occur:

**Input Data (X = pad character)**

**Valid?**

**ABCDE**

Yes

**ABC**

No

**ABCXX**

No

**XXXXX**

Yes (when type is optional)

**No (when type is mandatory)**

**X**

Yes (when type is optional)

**No (when type is mandatory)**

## Padded to > Allow Excess Trailing Pads

When you select **Padded to > Min Content**, the **Allow Excess Trailing Pads** property is displayed. Use this setting to flag an element as not valid when the following conditions apply:

- The element is padded to a minimum size.
- The number of pad characters exceeds the number that is required to achieve the minimum size.

## Pad > justify

Use the **Justify** property to specify whether the data is padded to the left or right.

### **Left**

The data will be on the left and will be padded (if necessary) on the right.

### **Right**

The data will be on the right and will be padded (if necessary) on the left.

Use the TRIMLEFT and TRIMRIGHT functions to exclude pad characters from the justified side.

In the following example, `xxxxxxxxxx` represent 10 spaces: For an input of `1234567891xxxxxxxxxx` with a right justified pad, the rule `=SIZE(input)` returns 20. For the same input, the rule `=SIZE(TRIMRIGHT(input))` would return 10 (removing the padding on the right).

## Pad > Apply Pad

Use the **Apply Pad** property to specify when to apply the pad character. Choose a value from the drop-down list.

### Fixed Group

Apply the pad characters only when the item appears in a fixed group.

### Any Context

Apply the pad characters when the item appears in any context.

Each item in a fixed group must be padded to a fixed size or have the same value for the minimum and maximum content size.

For example, suppose the item **Name** has a space pad character with this value:

Mary<sp><sp>

If the **Apply Pad** property is **Fixed Group**, the two spaces at the end are treated as pad characters only when the item appears in a fixed group. If the **Apply Pad** property is **Any Context**, then the spaces are always treated as pad characters.

Each item in a fixed group must be padded to a fixed size or have the same value for the minimum and maximum content size.

## Pad > fill

For padded signed numbers, this option determines placement of pad characters. Choose a value from the drop-down list.

### After Sign

Pad characters are placed after the sign.

### Before Sign

Pad characters are placed before the sign.

## Regular expressions

Regular expression is a sequence of characters to define a pattern.

Use the **Regular expression** property to generate text or numerical data that matches the given regular expression.

The maximum length of the regular expression is 2000 characters.



**Note:**



- Regular expressions do not support certain notations like \\d and \\w. You cannot use regular expressions in JSON and XSD schemas.
- When you use @, #, < symbol, you need to escape it with backslash (\).

## Restrictions

Restrictions of an item are the valid values of that item. When the **Interpret as** value is defined as **Character**, the **Restrictions** property is specific to the **Character** value and is not available as an option for a **Binary** value.

Depending on other **Item Subclass** settings, the **Restrictions** property can be set to **Value**, **Character**, or **Range**.

### Restrictions > ignore case

By default, restrictions are case-sensitive. To enable or disable the **Ignore case** property of the **Restriction**, choose from one of the following options:

#### Yes

Ignore case of restrictions in item type properties.

#### No

Do not ignore case-sensitive restrictions.

For example, if **Ignore Case** = **Yes** and the restriction **Value** is `ft`, the data values `ft`, `fT`, `Ft`, and `FT` are all valid.

### Restrictions > rule

This setting indicates whether you are going to include or exclude the criteria (as "valid" or "invalid") specified for the restrictions.

#### Include

Includes the restriction values as valid data.

#### Exclude

Excludes the restriction values as invalid data.

## National language

The **National language** default value is **Western**. For initiator, terminator, and release character **Literal** values, you can optionally specify a "Data language" on page 659.

When the **Interpret as** value is defined as **Character**, the **National language** property is specific to the **Character** value and is not available as an option for a **Binary** value.

## National language > data language

Use the **National language > Data language** property to define the data language or character set of this character text item.

## Supported code pages

HCL® OneTest™ Data supports the following code pages:

Arabic  
ASCII (deprecated)  
Big5  
Big5-HKSCS (IBM)  
BOCU-1  
CESU-8  
CII Kanji (deprecated)  
Cyrillic  
EBCDIC (deprecated)  
ebcdic-ar  
ebcdic-cp-ar1  
ebcdic-cp-ar2  
ebcdic-cp-be/ch  
EBCDIC-CP-DK/NO  
ebcdic-cp-es  
ebcdic-cp-fi/se/sv  
ebcdic-cp-fr  
ebcdic-cp-gb  
ebcdic-cp-he  
ebcdic-cp-it  
ebcdic-cp-roeece/yu  
ebcdic-de  
ebcdic-he  
ebcdic-is  
EBCDIC-JP-kana  
ebcdic-xml-us  
EUC (deprecated)  
EUC-CN  
EUC-JP  
euc-jp-2007  
EUC-TW  
euc-tw-2014  
GB\_2312-80  
gb18030

GBK  
Greek8  
Hebrew  
hp-roman8  
HZ-GB-2312  
IBM Kanji (deprecated)  
ibm-037 (ebcdic-cp-us/ca/wt/nl)  
ibm-1006  
ibm-1025  
ibm-1026  
ibm-1047  
ibm-1047-s390  
ibm-1097  
ibm-1098  
ibm-1112  
ibm-1122  
ibm-1123  
ibm-1124  
ibm-1125  
ibm-1129  
ibm-1130  
ibm-1131  
ibm-1132  
ibm-1133  
ibm-1137  
ibm-1140 (ebcdic-us-37+euro)  
ibm-1140-s390  
ibm-1141 (ebcdic-de-273+euro)  
ibm-1141-s390  
ibm-1142 (ebcdic-dk/no-277+euro)  
ibm-1142-s390  
ibm-1143 (ebcdic-fi/se-278+euro)  
ibm-1143-s390  
ibm-1144 (ebcdic-it-280+euro)  
ibm-1144-s390  
ibm-1145 (ebcdic-es-284+euro)  
ibm-1145-s390  
ibm-1146 (ebcdic-gb-285+euro)  
ibm-1146-s390  
ibm-1147 (ebcdic-fr-297+euro)  
ibm-1147-s390  
ibm-1148 (ebcdic-international+euro)



ibm-1148-s390  
ibm-1149 (ebcdic-is-871+euro)  
ibm-1149-s390  
ibm-1153  
ibm-1153-s390  
ibm-1154  
ibm-1155  
ibm-1156  
ibm-1157  
ibm-1158  
ibm-1160  
ibm-1162  
ibm-1164  
ibm-1250  
ibm-1251  
ibm-1252  
ibm-1253  
ibm-1254  
ibm-1255  
ibm-1256  
ibm-1257  
ibm-1258  
ibm-12712-s390  
ibm-1276 (Adobe Standard Encoding)  
ibm-1363 (korean)  
ibm-1363\_P110-1997  
ibm-1364  
ibm-1371  
ibm-1373\_P100-2002  
ibm-1386\_P100-2001  
ibm-1388  
ibm-1390  
ibm-1399  
ibm-16684  
ibm-16804-s390  
ibm-33722\_P120-1999  
ibm-37-s390  
ibm-437  
ibm-4517  
ibm-4899  
ibm-4909  
ibm-4971

ibm-5123  
ibm-5346  
ibm-5347  
ibm-5348  
ibm-5349  
ibm-5350  
ibm-5351  
ibm-5352  
ibm-5353  
ibm-5354  
ibm-5471\_P100-2006  
ibm-720  
ibm-737  
ibm-775  
ibm-803  
ibm-813  
ibm-8482  
ibm-850  
ibm-851  
ibm-852  
ibm-855  
ibm-856  
ibm-857  
ibm-858  
ibm-860  
ibm-861  
ibm-862  
ibm-863  
ibm-864  
ibm-865  
ibm-866  
ibm-867  
ibm-868  
ibm-869  
ibm-874  
ibm-875  
ibm-901  
ibm-902  
ibm-9067  
ibm-916  
ibm-921  
ibm-922

ibm-930  
ibm-933  
ibm-935  
ibm-937  
ibm-939  
ibm-9447  
ibm-9448  
ibm-9449  
ibm-949\_P110-1999  
ibm-949\_P11A-1999  
ibm-950\_P110-1999  
ibm-954\_P101-2000  
ibm-971\_P100-1995  
ibm-eucKR  
IBM-Thai  
IMAP-mailbox-name  
ISO\_2022,locale=ja,version=0  
ISO\_2022,locale=ja,version=1 (JIS)  
ISO\_2022,locale=ja,version=2  
ISO\_2022,locale=ja,version=3 (JIS7)  
ISO\_2022,locale=ja,version=4 (JIS8)  
ISO\_2022,locale=ko,version=0  
ISO\_2022,locale=ko,version=1  
ISO\_2022,locale=zh,version=0  
ISO\_2022,locale=zh,version=1  
ISO\_2022,locale=zh,version=2  
JIS (deprecated)  
KOI8-R  
KOI8-U  
Latin-9  
Latin1  
Latin1 (deprecated)  
Latin2  
Latin3  
Latin4  
Latin5  
Latin6  
Latin8  
LMBCS-1  
macintosh  
MS\_Kanji  
Native

SCSU  
Shift\_JIS  
shift\_jis78  
SJIS (deprecated)  
Thai8  
UNICODE Big Endian (deprecated)  
UNICODE Little Endian (deprecated)  
US-ASCII  
UTF-16  
UTF-16 Big Endian  
UTF-16 Little Endian  
UTF-16 Opposite Endian  
UTF-16 Platform Endian  
UTF-16,version=1  
UTF-16,version=2  
UTF-16BE,version=1  
UTF-16LE,version=1  
UTF-32  
UTF-32 Big Endian  
UTF-32 Little Endian  
UTF-32 Opposite Endian  
UTF-32 Platform Endian  
UTF-7  
UTF-8  
UTF-8 (deprecated)  
Windows 874  
Windows 949 (korean)  
x-iscii-be  
x-iscii-de  
x-iscii-gu  
x-iscii-ka  
x-iscii-ma  
x-iscii-or  
x-iscii-pa  
x-iscii-ta  
x-iscii-te  
x-mac-ce  
x-mac-cyrillic  
x-mac-greek  
x-mac-turkish  
x11-compound-text

## NONE and Zero

When the **Item Subclass** has a **Number** value and the **Interpret as** property has a **Character** value, the **NONE** and **Zero** properties are available. Expand the **NONE** or **Zero** property to specify an override data value to define the **Special Value** and **Required on input** properties.

### NONE > Special Value and Zero > Special Value

Enter a value in the **Special Value** property if the item is **NONE** or **Zero**.

For example, if you specify \* in the **Special Value** property for **NONE**, and a number object has the value \*, it is interpreted as **NONE**.

If you enter a value in the **Special Value** property for **NONE** or **Zero**, enter the exact characters to be validated in the input data and built in the output data.

The maximum element length that is allowed for the **Special Value** property is 3000 bytes.

### NONE > Required on input and Zero > Required on input

Select a value from the **Required on input** property list.

#### Yes

If the data object is **NONE** or **Zero**, the map uses the value in the **Special Value** property when it validates the item in the input.

#### No

Do not use the value in the **Special Value** property. No special values or input requirements are defined when data is **NONE** or **Zero**.

When the map builds the item in the output, the item might be either the value in the **Special Value** property or the default value for **NONE** or **Zero**. For example, if an item contains all pad characters and no actual data, the default value for **NONE** is used when that item is built in the output.

## Places

The **Places** property allows you to specify the number of implied decimal places.

For item types defined with an **Item Subclass** of **Number**, a **Decimal** presentation, and a **Separator**, the **Places** property allows you to specify the minimum and maximum number of decimal places and whole number places. The minimum number of places must be less than or equal to the maximum number of places.

If the decimal number has no separator, use the **Places** property to specify the number of implied decimal places.

## Text item subclass properties

You can interpret **Text** item subclass properties can be interpreted with a **Character** or **Binary** value. Depending on the value, character or binary, the following are text item subclass properties:

- ["Interpret as Binary" on page 644](#) or ["Interpret as Character" on page 645](#)
- ["Size \(content\)" on page 666](#)
- ["Pad" on page 652](#)
- ["Restrictions" on page 658](#)
- ["National language" on page 658](#)

## Interpret as binary

Binary text items have content size and pad properties. Binary data is required to be sized or of a fixed size.

Binary number items interpret the data as a binary number or as a byte stream.

Items with an item subclass of **Text** can be interpreted as character or binary.

The binary presentation options are Integer, Float, Packed, or BCD.

## Interpret as character

Character number items interpret the data as symbolic data. Symbolic data has the same meaning on different computers. For example, a comma is symbolic because it has the same meaning, regardless of the computer type.

Character text items can have content size and pad properties.

When you select **Character**, the **Release** property is specific to the **Character** value and is *not* available as an option for a **Binary** value. For more information, see ["Release characters" on page 637](#).

When interpret is set to **Character**, integer, decimal, and zoned presentation options are available.

## Size (content)

Each text or syntax object can be defined with the minimum and maximum sizes in bytes and/or characters. The product determines size for text or syntax objects according to the following rules.

- When the size of an object is in bytes only, size and validation are calculated according to the size limitations set for bytes.
- When the size of an object is in characters only, size and validation are calculated according to the size limitations set for characters.
- When an object has size values defined for both bytes and characters, the object size is determined by characters (based on any character size limitations). Bytes are used to verify that the size of the data did not exceed any byte constraints that were set.

When the content size of a text or syntax item is specified in bytes, initiators, terminators, release characters, and pad characters are excluded from the count.

### Min

The minimum size of the data object. You can enter values for bytes and/or characters.

The default value is 0. The largest value allowed is 65535.

### Max

The maximum size of the data object. You can enter values for bytes and/or characters.

The default value is 0. The largest value allowed is 65535.

When there is no maximum content size limitation, this value is not required.



**Note:** If you delete the value in the Max Characters field, the maximum content size is set to zero.

During schema analysis, checks are performed on byte and/or character **Min** and **Max** values according to the following methods:

- When values are present for both **Min > Bytes** and **Min > Characters**, the **Min > Bytes** value must be greater than or equal to **Min Characters**.
- When values are present for both **Max Bytes** and **Max Characters**, **Max Bytes** must be greater than or equal to **Max Characters**.
- When no size limitations are present, no verification takes place.



### Remember:

- Some character sets use two or more bytes per character.
- The trace file and audit log lengths are reported in bytes.

## Pad

Pad implementation determines how the pad value is sized during validation and how the pad value is sized when written as output.

When the data value of the target item is smaller than the minimum length of that item, pad characters can be used to pad the data to that minimum length. Output data is built according to the pad definitions of the types. Bytes are the default measurement for sizing, however, both byte and character sizing options are available.

### Yes

Enables the **Pad** option. Allows the item to contain both content and pad characters.

**No**

All data is assumed to be content on input; no pad characters are built on output.

## Regular expressions

Regular expression is a sequence of characters to define a pattern.

Use the **Regular expression** property to generate text or numerical data that matches the given regular expression.

The maximum length of the regular expression is 2000 characters.

**Note:**

- Regular expressions do not support certain notations like \\d and \\w. You cannot use regular expressions in JSON and XSD schemas.
- When you use @, #, < symbol, you need to escape it with backslash (\).

## Restrictions

Restrictions of an item are the valid values of that item. When the **Interpret as** value is defined as **Character**, the **Restrictions** property is specific to the **Character** value and is not available as an option for a **Binary** value.

Depending on other **Item Subclass** settings, the **Restrictions** property can be set to **Value**, **Character**, or **Range**.

## National language

The **National language** default value is **Western**. For initiator, terminator, and release character **Literal** values, you can optionally specify a ["Data language" on page 659](#).

When the **Interpret as** value is defined as **Character**, the **National language** property is specific to the **Character** value and is not available as an option for a **Binary** value.

## NONE

When the **Item Subclass** has a **Text** value and the **Interpret as** property has a **Character** or **Binary** value, the **NONE** property is available. Expand the **NONE** property to specify an override data value to define the **Special Value** and **Required on Input** properties.

## NONE > Special Value

Enter a value in the **Special Value** property if the item is **NONE**.

For example, if you specify \* in the **Special Value** property for **NONE**, and a text object has the value \*, it is interpreted as **NONE**.



If you enter a value in the **Special Value** property for **NONE**, enter the exact characters to be validated in the input data and built in the output data.

The maximum element length that is allowed for the **Special Value** property is 3000 bytes.

## NONE > Required on Input

Select a value from the **Required on Input** property list.

### Yes

If the data object is **NONE**, the map uses the value in the **Special Value** property when it validates the item in the input.

### No

Do not use the value in the **Special Value** property. No special values or input requirements are defined when data is **NONE**.

When the map builds the item in the output, the item might be either the value in the **Special Value** property, or the default value for **NONE**. For example, if an item contains all pad characters and no actual data, the default value for **NONE** is used when that item is built in the output.

## Bidirectional

Use the **Item subclass > Bidirectional** property to define the data as bidirectional.

Certain languages that are read from right-to-left often contain numeric or other phrases that are read from left-to-right. This type of data is called "bidirectional" because it changes direction in the middle of a line.

The default setting is **No**.

To enable this setting, select **Yes**. When you enable this setting, additional options are available to further define the bidirectional data.

### Text items

The Bidirectional property for text item types provide options for symmetric swapping, ordering, orientation, and shaping.

#### [Symmetric Swapping on page 641](#)

Use this property to maintain the orientation of directional pairs of characters (such as parentheses, greater than/less than symbols, brackets, braces).

#### [Ordering Scheme on page 765](#)

This property pertains to the order of the data as stored in memory.

**Logical**, the default setting, indicates that the data is stored in memory from left-to-right, regardless of the orientation of the text. Using the **Visual** ordering scheme, the data is read and stored with the start character in the right-most position both visually and in memory.

#### Orientation on page 642

Orientation pertains to the direction of the text as read visually. The default direction is left-to-right (LTR), meaning the text object is read starting from the left. The direction can be right-to-left (RTL), meaning the object is read starting from the right. When you select RTL, the behavior of some text functions (such as LEFT, RIGHT, MID) can change.

Use the **Contextual LTR** and **Contextual RTL** settings when the orientation should be taken from the context of the data because the data contains "strong" characters that are either orientation left or orientation right. When no strong characters are present in the data, the orientation will be based on this setting.

For example, when you set the orientation to **Contextual LTR** and no strong characters are encountered (meaning the data is orientation-neutral), the data orientation will be considered left-to-right.

#### Text Shaping on page 642

Use this property to produce output in a different glyph (shape or bit pattern). The default setting is off, or **Unshaped**.

To enable text shaping, select **Shaped**. The output will be shaped according to the code page of the target object.

### Character number items

For character number item types, you can additionally define a numeric shaping characteristic.

#### Numeric Shaping on page 642

Use this property to indicate whether the output of numeric values will have the shapes that are used in English (Arabic digits 0123456789) or the national numerical shapes.

The default setting is **Regular**, which means Arabic digits (0123456789). Otherwise, select **Arabic-Indic** (٠١٢٣٤٥٦٧٨٩).

The Bidirectional property is not applicable to binary numbers.

### Date & Time item subclass properties

The Date & Time properties provide the flexibility to define multiple combinations of date-time formats.

Date & Time items can be interpreted as either binary or character. (See sections ["Interpret as Binary" on page 644](#) and ["Interpret as Character" on page 645](#)).

For binary **Date & Time** items, the **Presentation** property has two different values: **Packed** and **BCD**.

**Packed**

The **Packed** value is a number that has size, decimal places, and sign properties.

**BCD**

The **BCD** value is a binary coded decimal number that has size.

Other **Item Subclass** properties for **Date & Time** are displayed depending on the **Interpret as** setting (**Binary** or **Character**).

## Date

For binary **Date & Time** items, use the **Date** property to define whether the date format is enabled.

**Yes**

**Date** format is enabled for this type. Expand the **Format** property to select the date format.

**No**

**Date** is not defined for this type.

## Date > format

Select the date format that the data interpretation will be based on. For binary **Date & Time** items, the **Date > Format** property has four values.

**CCYYDDD** and **YYDDD** Julian date formats are supported.

- CCYYMMDD
- YYMMDD
- CCYYDDD
- YYDDD

## Time

For binary **Date & Time** items, use the **Time** property to define whether the time format is enabled. Select one of the following options:

**Yes**

**Time** format is enabled for this type. Expand the **Format** property to select the time format.

**No**

**Time** is not defined for this type.

## Time > format

Select the time format that the data interpretation will be based on. For binary **Date & Time** items, the **Time > Format** property has several values. The following choices are available from the drop-down list:

- HH24MMSS
- HH24MM
- HH24:MM:SS
- HH24:MM
- Custom

## Format

You can define a date-time format from within the type properties, except for native schema XML.

The output format for the native schema XML date/time field is always:

```
{CCYY-MM-DD}T{HH24:MM:SS.3-3+/-ZZ:ZZ}
```

To specify a different output format, use Xerces XML instead.

To define the Date & Time format:

1. Open the type properties.
2. In the **Item Subclass > Format** property, click the browse button.

The **Date Time** dialog box is displayed.

3. Make your format selections and click **OK**.

You can use alphabetical characters as separators. In the example, 2001-04-02T10:32:59-0500, **T** is the separator.

Separators are limited to 60 characters.

Supported date formats:

- CCYYMMDD
- YYMMDD
- MMDDCCYY
- MMDDYY
- CCYYDDD
- YYDDD
- DDMMCCYY

- DDMMYY



**Note:** The DDMMCCYY and DDMMYY formats did not exist prior to version 6.5 of the Design Studio. If you had defined either one of these formats prior to 6.5 as a custom format, it will be recognized in 6.5 as the respective new format, instead of the previously defined custom format.

## Use of format elements

If you select the same format element more than once for the same item, that item may not be validated separately. If you specify **MON** twice, the day of the month is not validated separately for both months. For example, if the date format element **MON** is used twice for a single item, with the following format string:

MON D-MON D

Suppose the data is this:

Feb 28-Mar 31

28 is not validated for the month of February.

Some combinations of reserved words are invalid. A separator must follow reserved words representing a variable number of digits (**D** and **M** for date) if data follows.

For example:

- D/M/CCYY is valid.
- CCYYM/D is valid.
- DMCCYY is invalid.

See the Design Studio Introduction documentation for a list of reserve words and symbols.

## Custom date format

After choosing **Custom** from the date format drop-down list, click browse. The **Date Format** dialog box is displayed.

You can only use non-alphanumeric characters (excluding the {, } and [, ] non-alphanumeric characters) as separators in the custom **Date Format** dialog box.

The following table provides examples of date formats.

Date Format	Description	Example
CCYY	4-digit Century + Year	1999
YY	2-digit Year (00-99)	99
MM	2-digit Month (01-12)	12
M	1- or 2-digit Month (1-12)	8

Date Format	Description	Example
MON	3-character Month (Jan to Dec)	JAN
MONTH	Full name of Month	January
DDD	3-digit Day of year (001-366)	32
DD	2-digit Day of Month (01-31)	31
D	1- or 2-digit Day of Month (1-31)	7
DY	3-character Day of Week (Sun-Sat)	Fri
DAY	Full Name of Day of Week (Sunday-Saturday)	Friday
WW	1- or 2-digit Week of Year	13
Qn	Quarter of Year (Q1-Q4)	Q2
Custom	User defined custom date format	

After you define and save a custom format, the custom format string is displayed in the **Properties** window. For example, the custom date format **CCYYMMDD**, and the custom time format **HH24MMSS** display as:

```
{CCYYMMDD}{HH24MMSS}
```

## Custom Time Format

After selecting **Custom** from the time format list, click browse. The **Time Format** window is displayed.

You can only use non-alphanumeric characters as separators, excluding the { , } and [ , ] non-alphanumeric characters, in the **Time Format** window.

The following table provides time format examples.

Time Format	Description	Example
HH24	2-digit hour in 24 hour format (00-23)	23
H24	1- or 2-digit hour in 24 hour format (0-23)	11
HH12	2-digit hour in 12 hour format (00-12)	08
H12	1- or 2-digit hour in 12 hour format (0-12)	8
MM	2-digit minute (00-59)	09
M	1- or 2-digit minute (0-59)	9
SS	2-digit second (00-59)	05
S	1- or 2-digit second (0-59)	5
AM/PM	Meridian (AM/PM)	AM

Time Format	Description	Example
ZZZ	A time zone abbreviation. See <a href="#">"Time Zones" on page 675</a> for a list of supported time zones.	EST
+/-ZZZZ	Hours and minutes before or after the Greenwich Mean Time (GMT). GMT is now referred to as Coordinated Universal Time (UTC).	+0500
+/-ZZ:ZZ	4-digit time where the format is a 2-digit hour and 2-digit minute, separated by a colon.	+05:00
+/-ZZ[:ZZ]	4-digit time where the format is a 2-digit hour and an optional 2-digit minute, separated by colon.	+05:00
+/-ZZ[ZZ]	4-digit time where the format is a 2-digit hour and an optional 2-digit minute.	+0500
TZD	4-digit time where the format is a 2-digit hour and 2-digit minute, separated by a colon. Z on output, if value is +/-00:00.	+05:00

After you define and save a custom format, the custom format string is displayed in the **Properties** window. For example, the custom date format **CCYYMMDD**, and the custom time format **HH24MMSS** display as:**{CCYYMMDD}{HH24MMSS}**

## Time zones

The following table lists the supported time zones.

Time Zone	Abbreviation	Hours before Greenwich Mean Time	Hours ahead of Greenwich Mean Time
Greenwich Mean Time (Zulu)	GMT	0	0
West African Time	WAT	-1	+23
Azores Time	AT	-2	+22
No name; Brasilia Time	###	-3	+21
Atlantic Standard Time	AST	-4	+20
Eastern Standard Time	EST	-5	+19
Central Standard Time	CST	-6	+18
Mountain Pacific Time	MST	-7	+17
Pacific Standard Time	PST	-8	+16
Yukon Standard Time	YST	-9	+15
Hawaii Standard Time	HST	-10	+14

Time Zone	Abbreviation	Hours before Greenwich Mean Time	Hours ahead of Greenwich Mean Time
Nome Time	NT	-11	+13
New Zealand Time	NZT	-12	+12
No name; no location	###	-13	+11
Guam Standard Time	GST	-14	+10
Japan Standard Time	JST	-15	+9
China Coast Time	CCT	-16	+8
West Australia Time	WAT	-17	+7
Zulu + 6 (Russia zone 6)	ZP6	-18	+6
Zulu + 5 (Russia zone 5)	ZP5	-19	+5
Zulu + 4 (Russia zone 4)	ZP4	-20	+4
Baghdad Time	BT	-21	+3
Eastern European Time	EET	-22	+2
Central European Time	CET	-23	+1

## Time zone format string for XML



**Note:** The output format for the native schema XML date/time field is always:

```
{CCYY-MM-DD}T{HH24:MM:SS.3-3+/-ZZ:ZZ}
```

To specify a different output format, use Xerces XML.

The following time zone strings support the specification of a single character Z to indicate Coordinated Universal Time (UTC), as described by the World Wide Web Consortium ([www.w3c.org](http://www.w3c.org)) and the ISO 8601 standard:

- +/-ZZZZ
- +/-ZZ:ZZ
- TZD

When the +/-ZZ:ZZ or +/-ZZZZ format string is specified, the data validation process works in the following manner:

- Valid data that corresponds to this format string will contain either a character literal Z (representing UTC) or a zone in the appropriate format: +/-ZZ:ZZ or +/-ZZZZ, as specified.
- If a character literal Z is present, it shall be interpreted, and treated at mapping time, as UTC, which is equivalent to +00:00 or +0000.



When the TZD format string is specified, the data validation and output generation processes work in the following manner:

- Valid data that corresponds to this format string will contain either a character literal Z, representing UTC, or a zone, in the +/-ZZ:ZZ format.
- If a character literal Z is present, it shall be interpreted, and treated at mapping time, as UTC, which is equivalent to +00:00.
- If the value is +/-00:00, it generates Z in the output. Otherwise, it generates the output in +/-ZZ:ZZ format.

The following time zone strings support the omission of the minute portion of the difference from UTC:

- +/-ZZ[:ZZ]
- +/-ZZ[ZZ]

## Optional time segments of the time format string

This section discusses how you can specify a portion of a time-format string to be optional. For example, you can specify that either the *time zone* portion or the *hours, minutes, seconds, fractional seconds, Meridian* portion is optional.

This flexibility is also applicable to time-format strings used in functions or in schema scripts imported by the Type Tree Maker.

Only the time portion **or** the zone portion can be optional-not both.

To specify a portion of the time-format string as optional:

1. The following procedure assumes that the **Item Subclass** property of your schema is **Date & Time**.
2. From within the schema properties, navigate to **Item Subclass > Format**.
3. Click the browse button to open the **Date Time** dialog.
4. From a Format field that is set to **Time**, go to the field directly below it and choose **Custom** from the drop-down menu.
5. Next to the Format field with **Custom** selected, click the browse button. The **Time Format** dialog is displayed.
6. Choose from the following tasks:
  - To specify the *hours, minutes, seconds, fractional seconds, and Meridian*-portion of the time format string as optional, enable the check box on the far left side of the dialog.
  - To specify the *time zone* portion of the time format string as optional, enable the check box.

Only one segment of the time format string can be specified as optional.

## Date and time format examples

The following list includes examples of popular standard date and time formats:

### Format

### Description

## X12 EDI

Fractional seconds format: HH24MM[SS[0-2]].

## SWIFT

Time Zone Designator or UTC Designator format: HH24MMSS[TZD].

TimeStamp format: CCYY-MM-DDTHH:MM[SS[.0-6]][TZD], which is a combination of date and time.

## HL7

Time format: HH24MM[SS[.0-6]][+/-ZZZZ].

TimeStamp format: CCYYMMDDHHMM[SS[.0-6]][+/-ZZZZ], which is a combination of date and time.

## SAP

Time format: HHMMSS.

## ODBC

Time format: HH:MM:SS[.0-9], which is the form most commonly used in **ODBC** mapping. The fractional part is optional and not often used.

## NONE and Zero

When the **Item Subclass** has a **Date & Time** value and the **Interpret as** property has a **Character** value, the **NONE** and **Zero** properties are available. Expand the **NONE** or **Zero** property to specify an override data value to define the **Special Value** and **Required on input** properties.

### NONE > Special Value and Zero > Special Value

Enter a value in the **Special Value** property if the item is **NONE** or **Zero**.

For example, if you specify \* in the **Special Value** property for **NONE**, and a data and time object has the value \*, it is interpreted as **NONE**.

If you enter a value in the **Special Value** property for **NONE** or **Zero**, enter the exact characters to be validated in the input data and built in the output data.

The maximum element length that is allowed for the **Special Value** property is 3000 bytes.

### NONE > Required on input and Zero > Required on input

Select a value from the **Required on input** property list.

**Yes**

If the data object is **NONE** or **Zero**, the map uses the value in the **Special Value** property when it validates the item in the input.

**No**

Do not use the value in the **Special Value** property. No special values or input requirements are defined when data is **NONE** or **Zero**.

When the map builds the item in the output, the item might be either the value in the **Special Value** property or the default value for **NONE** or **Zero**. For example, if an item contains all pad characters and no actual data, the default value for **NONE** is used when that item is built in the output.

## Syntax item subclass properties

Syntax objects are characters that precede, separate, or follow a particular data object. Item types with an **Item Subclass** of **Syntax** are defined and used to specify delimiters, initiators, terminators, and release characters. Syntax objects with variable values are defined as the syntax object's **Variable** property. Syntax objects appearing as actual data to be mapped as output data are defined as components of a type.

## Syntax objects with variable values

### About this task

Using a syntax item to specify delimiters, initiators, terminators, and release characters is required when the value of syntax objects (delimiters, initiators, terminators, and release characters) may vary.

The restrictions of syntax items define the variable literal values.

To define a variable syntax object:

1. Create an item type with an **Item Subclass** of **Syntax** to represent the syntax object.
2. Define the restrictions for the syntax item type.
3. For the item or group type with the variable syntax object, for the **Variable** delimiter, initiator, terminator, and release character, select the item from the **Item** drop-down list in the **Properties** view.
4. For the **Find** property, select **Yes**.

Only item types defined with an **Item Subclass** of **Syntax** appear in the drop-down list.

## Example of variable syntax object as an item type

In this example, the group type **Header** has variable delimiter values of: , \* + ~

1. Create an item type with the **Item Subclass** of **Syntax** and the name **Delimiter**. (The name can be any valid type name, but naming this item **Delimiter** is useful).
2. To define the item restrictions, double-click the **Delimiter** type. The item view is displayed.
3. Define the restrictions in the **Include** column.

4. In the **Properties** window, select **Syntax** for the **Item Subclass** property.
5. Open the properties for the delimited group type **Header**.
6. For the **Syntax** property, choose **Delimited**.
7. For the **Delimiter** property, choose **Variable**.
8. Expand the **Delimiter** property.
9. Define the **Delimiter > Default** literal value.
10. For the variable **Delimiter Item** property, choose the item type **Delimiter** from the drop-down list.
11. Indicate that the variable delimiter should be determined for each occurrence of the object by selecting **Yes** for the **Delimiter > Find** property.

## Syntax objects as data

The value of a syntax object (delimiters, initiators, terminators, and release characters) may appear as actual data that can be mapped. Syntax objects that appear as actual data are defined as components of a type.

## Example of syntax objects as components of a type

The following is an example of defining a syntax object as a component of a group type. The following data represents a message received from multiple departments.

Each message is made up of segments. Each department uses different segment delimiters and terminators. The message format specifies that the delimiter and terminator are the first two bytes of the message, followed by the actual data.

To define syntax objects for the delimiter and terminator:

1. Define two separate syntax objects with an **Item Subclass** of **Syntax**, one for the message delimiter and one for the message terminator. Appropriate type names might be **SegmentDelimiter** and **SegmentTerminator**.
2. Define the possible message delimiter values as restrictions of the **SegmentDelimiter** item type. Define the possible message terminator values as restrictions of the **SegmentTerminator** item type.
3. Define these item types as the first two components of the group type **Message**.
4. Expand the **Delimiter** property.
5. For the variable **Delimiter > Item** property, select **SegmentDelimiter** from the drop-down list.
6. For the variable **Terminator > Item** property, select **SegmentTerminator** from the drop-down list.

During the data validation process, the component **SegmentDelimiter** appears in the data with a value of \*. The group type **Segment** has a variable delimiter specified as the item type **SegmentDelimiter**, with the value \*. Therefore, it is understood that the segment has \* as the delimiter.

When the value of a syntax object appears as actual data, it can be mapped as data. For example, source data may use different delimiters from several different sources. Acknowledgments of this data must be sent back to the source using the delimiter used in the original data. This data can be mapped from the input to the output if these syntax objects are defined as components within the data.

## Delimiter > find

When syntax objects are **Variable**, the **Find** property must be defined. The **Find** property determines whether the value of syntax object is set on each occurrence or whether the current setting (or default value) is used.

When the **Initiator**, **Terminator**, **Release**, or **Delimiter** property value is **Variable**, select one of the following from the drop-down list in the **Value** column:

### Yes

Determine the value of the syntax object each time an occurrence of that type is found. After the value of that syntax object is found, that particular value is used until it is reset by another **Find** or by the occurrence of that syntax item as a component.

INPUT: The value of the object in the input data is determined by the location in the data stream and the restrictions of the syntax item.

OUTPUT: The default value is used for building the syntax object in the output data.

### No

The variable syntax object is defined as the current value or, if it is not set, as the default value.

INPUT: If the syntax object is not encountered in the data stream, the value of the syntax object is the default value.

OUTPUT: If the value of the syntax item has *not* been previously set, the default value is used.

## Group properties

Group properties include the group's **Subclass** and **Format**, which describes how to distinguish one component of that group from another component of that group.

## Group subclass

Group types have a subclass of **Sequence**, **Choice**, or **Unordered**.

Property	Description
Sequence	<p>A partially-ordered or sequenced group of data objects.</p> <p>Each component of a <b>Sequence</b> group is validated sequentially.</p>
Choice	<p><b>Choice</b> group provides the ability to define a selection from a set of components like a multiple-choice question on a test.</p> <p>A <b>Choice</b> group is valid when the data matches one of the components of the choice group. Validation of a <b>Choice</b> group is attempted in the order of the components until a single component is validated. If the <b>Choice</b> group has an initiator, the initiator is validated first.</p>

Property	Description
Unordered	An <b>Unordered</b> group has one or more components. <b>Unordered</b> group can only have an <b>Implicit</b> format property, with the same syntax options as sequence groups: <b>None</b> or <b>Delimited</b> .

## Properties of group subclasses

The distinction between **Sequence** and **Choice** group subclasses is that the **Choice** group has no **Partition** or **Format** properties.

Type syntax properties such as **Initiator**, **Terminator**, **Release**, and **Empty** can be applied to **Choice** groups. If a release character is defined, by default it applies to the **Terminator**.

## Choice group components

A **Choice** group can have both items and groups as components. A partitioned **Sequence** group can only have group subtypes.

The components of **Choice** group must be distinguishable from each other. The **Choice** group components cannot have a range other than (1:1). Only one component of a **Choice** group built in the output data.

A **Choice** group data type is only one of the group components. For example, the data type **Record** is a group type with a **Group Subclass** of **Choice**. The group type **Record** has three components: **Order**, **Invoice**, and **Sales**. The data validation of **Record** can have only one of the components: **Order**, **Invoice**, or **Sales**. For this reason, the components of a **Choice** group must have a component range of (1:1).

## Unordered group components

An **Unordered** group has one or more components that can appear in the data stream in any order. They allow many SWIFT and FIX message types, for example, to be defined in a more natural way.

**Unordered** groups have no partitioned property. They have **Implicit** format properties with the same syntax options as sequence groups: **None** and **Delimited**.

When a group is defined as **Unordered**, any component can appear in the data stream. A component can be an item or a group.

Unordered group components have a range property. For example, if the unordered group, **A**, has the following component list:

```
B(1:S)
C
D(S)
```

then **A** must have one **C**, at least one **B**, and possibly some **Ds**. They could appear in any order. For example, data for **A** could have the pattern: **CDDBDDD** or **BBDDCDBD**.

Component rules of an unordered group cannot reference other components of the same group. They can only reference the component to which the rule refers and the objects contained in that component.

## Sequence group formats

A **Sequence** group has either an **Explicit** or **Implicit** format.

### Explicit

The explicit format relies on syntax to separate components. Each component can be identified by its position or by a delimiter in the data. Delimiters appear for missing components.

### Implicit

The implicit format relies on the properties of the component types. The format is not fixed. If delimiters separate components, they do not appear for missing components.

For example, if each component of a fixed group has a fixed size, the component is distinguished from the next component by its position in the data. Or, a group may have delimiters that appear for missing components. In these cases, the format is apparent; the group has an explicit format.

If a group does not have an explicit format, it has an implicit format. An implicit format relies on the properties of the component types. In this example, the components make some pattern in the data and it is possible to distinguish between them, but the format is not fixed and if delimiters separate components, they do not appear for missing components.

When deciding what format a group has, it may help to ask first whether it is clear where one component ends and another begins. Generally, a group has an explicit format if the position of each component in the data stream is always the same or if a delimiter always marks the place for each component.

## Explicit format

To specify that a group has an explicit format, choose **Explicit** for the **Format** property. Select a setting for the **Track** property, and choose the group's syntax: **Fixed** or **Delimited**.

## Track

The **Track** property indicates whether the system should track only the components that have content or all components, including those that do not have content. The settings are **Content** and **Places**.

For example, if a group **StudyGroup** has the component **Name(s)**, and **Places** is specified for the **Track** property, any empty occurrences of **Name** are tracked.

Suppose **Name** has an \* initiator and a space pad character. This is the data for **StudyGroup**:

```
*Carolyn * *Stuart *Margaret
```

Notice that the second **Name** is missing.

If **Places** is specified for the **Track** property, and you want to map the third **Name**, the empty **Name** would be counted as an occurrence, so the third **Name** would be Stuart. However, if **Content** is specified for the **Track** property, the empty **Name** would not be counted as an occurrence because it does not have content and the third **Name** would be Margaret.

## Fixed syntax

If a group data object is always the same size (in bytes), it has a fixed syntax. For example, a record that is always 160 bytes has a fixed syntax.

Each component of a fixed group must be fixed. If you break down a fixed group, it ultimately consists of items that are fixed. Each is padded to a fixed size or its minimum and maximum content size are equal. Do not specify the size of a fixed group. The size is automatically calculated based on the size of the group's components.

## Guidelines for defining a fixed group

- Each component must be a group with a fixed syntax or a fixed item. The item is padded to a fixed size or its minimum and maximum content size are equal.
- Each component must have a specified range maximum. The maximum cannot be ∞.
- If a component range minimum is not equal to the maximum, content is not required for optional component occurrences. For example, if a component is the item **ShippingAddress** (0:1) and **ShippingAddress** has a minimum content size of two characters, data may either contain: 1) all pad characters, or 2) if content is in the data stream, there must be a minimum of two characters for a **ShippingAddress**. If a component is a group, no content is required for any of the items contained in an optional occurrence of that component.

## Explicit delimited syntax

An explicit-format group with a delimited syntax is one whose components are separated by a delimiter and the delimiter appears as a placeholder even when a component has no content.

### About delimiters and explicit-format groups:

- The only time a delimiter can be missing is when all components following the delimiter are optional and there is no data for the optional components.
- A delimiter is a character or series of characters that separates data objects.
- A delimiter cannot be longer than 500 bytes.
- The delimiter of a group appears inside the group, separating its components. When a group is delimited, that indicates something about the components of the group. The delimiter inside a group is delimiting the components.

## Example

The group **Employee** has an explicit-delimited format because a comma delimiter appears between **Employee's** components, the items that make up the **Employee** object. In addition, the delimiter is displayed when a component is missing and there is data for components following it.



The components of **Employee** are the items **ID#, Name, Department, Address,** and **Age**.

## Delimiter

Use the **Delimiter** property to specify the value and location of the delimiter. For specific information on specifying the delimiter, see [“Specifying a Delimiter” on page 687](#) .

In an explicit-delimited group, if the delimiter is whitespace <WSP>, the delimiter is interpreted as one byte long; each <WSP> character is interpreted as another delimiter. In the output, a <WSP> delimiter is one space.

The limit for <WSP> is 512 bytes.

## Implicit format

In a group with an implicit format, the components are distinguishable not by delimiter or position, but by their pattern; something in the definition of the component types themselves. The group has no syntax property that distinguishes one component from another.

For example, the type **File** consists of **Record(s)**. <CR><LF> appears at the end of each **Record**. It has been defined as the terminator of **Record**. **File**, however, has no syntax of its own. The **Record** terminator distinguishes one **Record** from another.

To specify that a group has an implicit format, choose **Implicit** for the **Format** property. Optionally, define a comment type, and choose the group's syntax: **Delimited** or **None**.

## Floating component

The floating component represents an object that may appear after any component of the group.

An implicit group can have a floating component; an explicit group cannot. If the group is prefix or infix delimited, the floating component is displayed before the delimiter. If the group is postfix delimited, the floating component is displayed after the delimiter.

A floating component can be an optional component that may appear after any other component. However, it is not included in the component list because it does not appear at a specific location.

If a group has a floating component, a component must be distinguishable from a floating component. For example, components and floating component could start with different initiators.



**Note:** When a floating component appears in the input data, it is validated during mapping. If there are floating components in your output data, define them as actual components of the output.

A floating component can appear after the initiator (when the type has an initiator), after each component, or both. For EDI and other existing floating components, trees will be converted as *“after each component”*.

A floating component can be specified for implicit sequence group, choice group, and unordered group definitions.

## Implicit whitespace syntax

Select **Whitespace** for the **Component Syntax** value when white spaces are not allowed in the data. When you select the **Whitespace** option, define the **Build As** and **Character Set** properties.



**Note:** Helpful for XML data.

### Build as

Enter the characters that will replace white spaces.

The **Build As** property is only available when the **Component Syntax** property is defined as **Whitespace**.

### Character set

Select a character set for the component from the drop-down list.

The default value is **Native**, where the literal values use the native character set of the computer.

[Supported character sets \(code pages\) on page 659](#)

## Implicit delimited syntax

If a delimiter separates the components of a group, but the delimiter does not appear when a component is missing, the group has an implicit format with a delimited syntax.

In certain data, the delimiter may not be a placeholder.

### Delimiter

Use the **Delimiter** property to specify the value and location of the delimiter. For specific information on specifying the delimiter, see [“Specifying a Delimiter” on page 687](#).

In an implicit-delimited group, when the delimiter is whitespace <WSP>, all contiguous <WSP> characters are treated as one delimiter. In the output, a <WSP> delimiter is built as one space.

The limit for <WSP> is 512 bytes.

### No syntax

To specify a group with an implicit format that has no syntax, choose **None** for the **Component Syntax** value.

## Distinguishable components of an implicit group

Each component of an implicit group needs to be recognizable. If either of two different components appears at the same place in a data stream, there must be a distinguishable difference between one component and the other.

Sometimes components of an implicit group may be distinguished because there is something in the data that distinguishes them.

After the first item **Line** of the **Form**, the next data object could be another item **Line**. Or, it could be a **Trailer Line**. When looking at a particular **Line**, there is something in the data that identifies that **Line** either as a **Header Line**, an item **Line**, or a **Trailer Line**. The type **Line** has been partitioned to distinguish between the different kinds of **Lines**.

## Specifying a delimiter

For the **Delimiter** property, specify whether the delimiter is a literal or a variable value.

### Literal

If the delimiter is a constant value, enter the delimiter value in the **Value** field. To enter a non-printable value, click the browse button and select a value from the **Symbols** dialog box.

### National language

The **National language** default value is **Western**. For initiator, terminator, and release character **Literal** values, you can optionally specify a ["Data language" on page 659](#).

When the **Interpret as** value is defined as **Character**, the **National language** property is specific to the **Character** value and is not available as an option for a **Binary** value.

### Data language

Use the **National language > Data language** property to define the data language or character set.

[Supported code pages on page 659](#)

### Variable

Sometimes you do not know what delimiter is used in the data source, especially if you receive that data from an outside source. However, you know all of the possible values that the delimiter could be. You can create an item to represent this delimiter and specify all of the possible values as restrictions of that item. The value of the delimiter in the data is found.

To specify an item as a Variable Delimiter:

1. In the **Properties** view for the delimited group, click in the **Delimiter > Item** value field.
2. Press **ALT** and drag the item from the schema editor into the **Item** field.

### Location

The **Location** property specifies the location of the delimiter with respect to the components. The options are prefix (the delimiter appears before the component), postfix (the delimiter appears after the component), and infix (the delimiter appears between components).

The following table explains each option.

Property	Description	Example
Prefix	Before each component.	*a*a*b*c
	Before each member of a component in a series.	
Postfix	After each component.	a*a*b*c*
	After each member of a component in a series.	
Infix	Between components.	a*a*b*c
	Between members of a component in a series.	

## Delimiter value appears as data

Delimiters do not appear as part of the actual data. For example, if the delimiter is specified as a comma, the comma in the following text item would be considered a delimiter, rather than part of the data itself:

Tom Smith, Jr.

If the data does contain the delimiter value, there are ways to format the data so that both the data and the delimiter are distinguishable. For example, text items can be enclosed in quotation marks. Or, a release character may be used. For information about release characters, see ["Release Characters" on page 637](#).

## Allow Excess Trailing Delimiters

The schema validation process allows data that contains excess trailing delimiters on sequence groups.

You can change the default setting for the **Allow Excess Trailing Delimiters** property to **No** so that the schema validation process, instead, fails data that contains excess trailing delimiters.

The schema validation process interprets a delimited sequence group that contains optional components at the end of its sequence, as having excess trailing delimiters.

For example, `GROUPA` contains the following components: `COMPA` (mandatory) , `COMPB` (optional) , `COMPC` (optional), and `COMPD` (optional). The group is infix-delimited by a tilde. The following data stream examples are valid for `GROUPA`:

```
A~B~C~D
A~B~C~
A~B~~
A~~~
```

For this example, you want the schema validation process to disallow all of the data stream examples that contain unnecessary or excess trailing delimiters. To disallow the excess trailing delimiters, you can change the setting for

the **Allow Excess Trailing Delimiters** property to **No**. The result is that the validation process fails all of the examples except the first one.

## Defining exclude characters

During the run time, the data validation process uses the **Exclude Character List** group property to define a set of characters that are not allowed to exist in the data stream as data content.

The data validation process allows these excluded characters to exist in the data stream as syntax, such as delimiters, if they are not being used as data content.

The group exclude characters might be derived from previously parsed values in the data stream or might be defined as literal values.

When you define exclude characters at the group level, the data validation process handles these specified characters as if they were in-scope data syntax.

For example, you have a schema which defines two groups, `GROUPA` and `GROUPB`. The tree defines the asterisk (\*) character as a group delimiter for `GROUPA`. It defines the at-sign (@) character as a group delimiter for `GROUPB`. The tree also defines the `FILE` group, which contains both `GROUPA` records and `GROUPB` records.

The data stream contains content corresponding to the `FILE` group, which is composed of a series of both `GROUPA` and `GROUPB` records.

When the data validation process starts, it parses and validates this data stream. Since the asterisk is the delimiter of `GROUPA`, the data content of `GROUPA` records is only allowed to contain asterisk characters that are escaped. If the data content contains non-escaped asterisk characters, those `GROUPA` records fail data validation. Similarly, since the at-sign is the delimiter of `GROUPB`, the data content of `GROUPB` records is not allowed to contain non-escaped at-sign characters. If it does, those `GROUPB` records fail data validation.

The schema defines the delimiters, or markup, for the `GROUPA` and `GROUPB` records. Asterisk characters are considered to be in-scope markup while the data validation process handles `GROUPA` records. At-sign characters are considered to be in-scope markup while the data validation process handles `GROUPB` records.

Conversely, `GROUPA` data content is allowed to contain at-sign characters, as the `GROUPB` at-sign delimiter does not apply to `GROUPA` records. Therefore, at-sign characters are considered out-of-scope markup for `GROUPA` records. `GROUPB` data content is allowed to contain asterisk characters, as the `GROUPA` asterisk delimiter does not apply to `GROUPB` records. Therefore, asterisk characters are considered out-of-scope markup for `GROUPB` records.

You might require the data validation process to exclude asterisk and at-sign characters in all of the data content for both `GROUPA` and `GROUPB` records, except when those characters are used as delimiters.

If so, use the **Exclude Character List** property to specify the asterisk and at-sign literal values at the `FILE` group level. When the validation process evaluates the data, it throws an exception if there is an asterisk (\*) or an at-sign (@) character in the data content within the `FILE` group. The asterisk and at-sign characters are still valid if they are used as group delimiters; they are invalid if they are used in data content.



**Performance tip:** The **Exclude Character List** property causes the map validation process to use more resources as it searches for each of the exclude group characters in every data element under the group on which it is defined.

Open the **Exclude Character List** window by clicking the **Value** field on the **Exclude Character List** property, and then clicking the button that appears at the end of the field. Specify the characters in the list as literal characters or syntax items. You can use the **Add Literal**, **Add Syntax Item**, **Edit**, **Remove**, and **Remove All** functions to add literal characters and syntax items to the list, edit them, and remove them from the list.

## Escaping an excluded character in the data content

Select **Release Exclude Characters** when you configure the exclude character list to enable the release character to escape excluded characters. When enabled, a release character that precedes an excluded character in data content escapes the excluded character, and the data content passes validation.

---

Related reference

[Release characters on page 637](#)

## XML properties in the schema

When you open a schema in the schema designer after you imported it from a DTD or XML Schema, the XML-specific properties are displayed as read-only fields.

For each type in the resulting schema, the **Intent** property is either **General** (indicating non-XML) or **XML**. All category types that are created during the import process are considered non-XML properties. All group and item types that are created during the import process are considered XML properties.

Any types that you manually add to the XML type tree are considered non-XML and have **General** intent. In such a case where there are both XML and non-XML types present (such as EDI data), the appropriate method of validation is determined automatically.

## Support for XML constructs

HCL® OneTest™ Data validation supports the following XML constructs.

### Character Data

During validation, character data is mapped by the XML parser to HCL® OneTest™ Data types. This data includes both parsed character data (PCDATA) and unparsed character data (CDATA).

### Comments and Processor Instructions

XML comments and processing instructions (PI) are mapped to floating components in schemas.

## Namespaces

The XML Schema importer supports the specification of arbitrary prefixes for namespaces declared in the input grammar.

## XSDL Hints

The **xsi:schemaLocation** and **xsi:noNamespaceSchemaLocation** attributes are collectively known as XML Schema Definition Language (XSDL) hints. These attributes specify the location of the XML Schema(s) that the XML parser uses for validation.

An XML Schema allows either or both of these attributes to display within any element tag. But the XML parser respects the location values only when the XSDL hint is specified in the root element of the schema.

The **Doc** group type of schemas that are created with the XML Schema or XML DTD importer contains the location of the Xerces DTD or Xerces XML Schema that is used for Xerces XML validation. The **Intent > Validate As > Location** property can be modified. However, use the **Schema > Type > Metadata** card setting in the map to change the location of the DTD or Schema if needed.

## Empty Elements

Only one set of initiators and terminators for the Empty element is required for validation.

## Nillable Elements

A nillable element can have one of three states: absent, present with content, or present with nil content. Both the DTD and XML Schema allow the definition of optional elements in the instance document. Additionally, the XML Schema allows nillable elements where the content can be empty when it contains an **xsi:nil** attribute with a value of "true", despite the fact the element's content is mandatory.

The XML Schema Importer can create the content of a nillable element within a group with a range of (0:1), which makes the element content optional.

## Mixed Content

When parsing elements with a mixed content model (containing both character data and child elements), the DTD and XML Schema importers generate text items within the mixed content (instead of character sequences).

## Regular Expressions

An XML Schema allows the restriction of the values of simple types that are based on regular expressions or pattern facets. During XML validation, the parser enforces the pattern facet. The XML Schema Importer fills the appropriate type properties with pattern facets encountered in the input schema, which are viewable in the schema designer.

## XML schema datatypes

This section describes the schema constructs that correspond to XML Schema datatypes.

The following table lists some common XML Schema datatypes with their corresponding type property values in the schema designer.

XML Schema Datatype	Type Property	Value
simpleType	Intent > Validate As	XML_SIMPLETYPE
complexType	Intent > Validate As	XML_COMPLEXTYPE
element	Intent > Validate As	XML_ELEMENT
attribute	Intent > Validate As	XML_ATTRIBUTE
group	Intent > Validate As	XML_GROUP

## Simple types

Elements that have assigned simple types have character data content but not child elements or attributes.

Simple type elements are represented in the Type Designer type properties with the value XML\_SIMPLETYPE.

## Complex types

Elements that have assigned complex types can have both child elements and attributes. The order and structure of the child elements of a complex type are known as its content model. Content models are defined using a combination of model groups, element declarations or references, and wild cards. There are three kinds of model groups: **all**, **choice** and **sequence**.

Complex type elements, including the complex types using compositor elements, are represented in the type properties with the value XML\_COMPLEXTYPE.

## Elements

Element declarations are either local or global. Local elements are represented in the schema designer type properties with the value: XML\_ELEMENT.

When the schema defines several global elements, the generated schema includes a Global choice group that contains all of the global elements. A Global choice group is represented in the type properties with the value XML\_BODY.

## Attributes

Attributes are another building block of XML. The import process uses attribute declarations to name attributes and associate them to particular simple types. Attribute declarations can be either local or global.

Attributes, including both local and global, are represented in the type properties with the value XML\_ATTRIBUTE.



## Groups and substitution groups

One way an XML Schema allows the creation of reusable content is by using named model groups. These global groups can be referenced throughout a schema.

Substitution groups are a flexible way to designate element declarations as substitutes for other element declarations in a content model. New element declarations can easily be designated as substitutes from other schema documents or namespaces without changing the original content model.

Groups and substitution groups are represented in the type properties with the value XML\_GROUP.

## Identity constraints

The XML parser in HCL® OneTest™ Data enforces declared identity constraints. The XML Schema Importer sets these properties based on the XML Schema. In general, the identity constraint definition serves in one of the roles listed below:

### Identity Constraint

#### Unique

Enforces that a value (or combination of values) is unique within a given scope. For example, all product numbers must be unique within a catalogue.

#### Key

Enforces uniqueness and requires that all values be present. For example, every product must have a number and it must be unique within the catalogue.

#### Key References

Enforces that a value (or combination of values) corresponds to a value represented by a key or uniqueness constraint. For example, for every product number that is displayed as an item in a purchase order there must be a corresponding product number in the product description section.

## Namespaces

The XML Schema Importer supports namespace and prefix properties for all elements and attributes. The importer assigns the values of these properties based on the input grammar. For each namespace in the input grammar, the importer allows you to specify the namespace prefix to be used for output documents.

During the import process, a **Location** value is specified in the schema without a fully qualified path. In this case, an execution process would, by default, look for the XML Schema at this location. However, if you need to change this path, you can do so.

For example, if you refer to the proper location of the schema in the XML document, then you can manually remove the path location from the type properties. When the execution process does not find a **Location** value specified in the schema, the process then looks to the ["XSDL Hints" on page 691](#) specified in the XML document. When XSDL hints are not present, validation fails.

The value specified in the schema takes precedence over the XML document.

## Element type declarations

The XML DTD and XML Schema importers can generate a simple schema when encountering character content within mixed data. During the import process, when character data (CDATA/PCDATA) is encountered, the importer incorporates it into existing text items within the schema.

## Element and attribute wildcards

DTDs and XML Schemas allow the specification of wildcard elements within a grammar. The HCL® OneTest™ Data XML Schema and DTD importers recognize element wildcards, build the appropriate types, and set the appropriate properties for those types to represent element wildcards in the schema.

During the import process, when a wildcard element resolves to an element that is not defined in the imported grammar, it is represented by a text item.

The value of these text items is the text string from the input buffer that runs from the start of the open tag to the end of the close tag for that element.

XML Schemas allow the specification of an attribute wildcard that matches any number of undeclared attributes within the element tag. The XML Schema Importer recognizes the attribute wildcard and creates a sequence of name and value text item pairs within the attribute list of the enclosing element. The XML validation library fills this sequence with the names and values of attributes that do not match any declared attributes in the attribute list.

## Item restrictions

Item restrictions are an optional feature you can use to specify valid or invalid data objects for an item type. Item restrictions are grouped into three categories: **Value**, **Character**, and **Range**.

Restrictions of an item type are the valid or invalid values for that item. "Include" restrictions are all of the valid values for an item. "Exclude" restrictions specify all of the invalid values. For example, a unit of measure field in the data must be one of a set of values: `CN`, `BX`, `PK`, `BR`. These values should be defined as "include" restrictions of the item **UnitOfMeasure**.

Defining restrictions for an item restricts the valid data for that item. Partial lists of the valid values are not possible. For example, it is not possible to define the values for a certain item to be `{A, B, C, "some other possible values"}`.

While you can define restrictions for any item, most items will not have restrictions. For example, you would not want to restrict the valid values of a name field because you would probably want to accept any name as valid data for that field. However, you could assign restrictions to it if needed.

## Restrictions settings

Using the **Restrictions** property, you can create a "restriction list" to limit an item to a particular value or set of values. Depending on the **Item Subclass**, you can set value, character, or range restrictions.

## Range restrictions

*Include range restrictions* define the minimum and maximum values valid values of a range. *Exclude range restrictions* define the minimum and maximum values of the range that are to be excluded or considered invalid.



**Note:** Unbound restrictions are not supported.

## Value not in range

### About this task

The minimum value and the maximum values can be specified as not included in the range. For example, when a number in an **Include Minimum** field displayed the **Value NOT In Range** icon, it indicates that the number is not included as the minimum value. The range will therefore extend from any value that is greater than that number to the maximum value specified. Similarly, if the **Include Maximum** value is designated as “*not in range*”, the valid range would extend up to any value this is less than the maximum value.

To specify a value as “*not in range*”:

1. Select the field (cell) that contains the value to be designated as “*not in range*”.
2. Right click on the value and choose the **Value NOT In Range** menu item.

## Inserting symbols

### About this task

Symbols are used to indicate non-printable characters. You can insert a symbol by typing it or by using the **Symbols** dialog.

For example, to define a carriage return/line feed as an item restriction, from the item restriction view area, right-click and select **Insert Symbols** from the context menu. From the list of symbols, double-click `CR` and double-click `LF` and select **OK**. The carriage return/line feed symbols are displayed in angle brackets in the item restriction view area:

```
<CR><LF>
```

## Ignoring restrictions

There might be instances when you do not require the data for an item to match any of its restrictions. Suppose you defined restrictions for your data, but you want to run a test on some test data, and you do not want to use the restrictions for this time only. You can ignore the restrictions for a given execution of a map.

In another example, you have a list of valid part numbers that can appear in any data circulated within your company. Suppose you receive data from some other company in the same format as your internal data. The other company may be using different part numbers, so you do not want to make their data conform to the restrictions. When you map the other company's data, you could ignore the restrictions.

## Components

A component represents a data object that is part of another data object.

### Components are required for group types

Categories and groups can have components. Components of group and category types display in the group and category views. Item types do not have components.

Group types represent actual data objects, so groups must have components. The one exception is a partitioned group which is explained in ["Partitioning" on page 705](#).

By contrast, a category is used for organizing types and for type property inheritance reasons. Categories do not define actual data objects in detail. A category does not need components.

Each group must have at least one component, unless it is partitioned.

### Components must be in the same schema

A component must be a type in the same schema as the type that contains the component.

You cannot define the components of a type by opening up a different schema and dragging components from that tree. You can, however, copy types from one schema to another.

### Importance of component order

In the group view, components are listed from top to bottom in the order they appear in the data stream. The component in the first cell appears first in the data stream. The component in the second cell appears next, and so forth.

### Component range

A component range can be specified for any component. A component range defines the number of consecutive occurrences of that component. The range (  $\geq$  ) represents some or any number greater than one.

When a component is selected, the component name is displayed in the rule bar. Click in the rule bar after the component name to type the component range.

The range is displayed in parentheses immediately after the component name. The range is two numbers separated by a colon. The first number indicates the minimum number of consecutive occurrences of that component. The second number indicates the maximum number of consecutive occurrences of that component. The syntax of the component name and range is:

Component (MIN:MAX)

To enter a range after a component name, type in the rule bar or use the Set Range command.

The maximum component range is 2147483647.

Whenever a component has a range, each occurrence of that component may be referred to as a "member" of a series. The words "occurrence" and "member" are used interchangeably in the documentation.

## Indefinite number

When there is no maximum number of occurrences of a component (the maximum is indefinite) use the letter **s** to stand for "Some - you do not know how many". Therefore, a file that contains at least one record and has no maximum number of records has this component:

```
Record(1:s)
```

If the range minimum is zero, omit the **0** and only enter **s**. If a file has a minimum of zero records and no maximum number of records, it would have this as its component:

```
Record(s)
```

Remember that when you see **(s)**, the minimum of zero is implied. Therefore,

```
Record(0:s) is the same as Record(s)
```

If you enter any number alone in the parentheses, the range changes to a minimum of 0 and a maximum of that number. For example, if you enter **(5)** for a range and save and close the group view, the next time you open it, you see the range displays:

```
(0:5)
```

## Single occurrence

If there is a minimum and a maximum of one consecutive occurrence of a component, its range is **(1:1)**. This is the default. If there is no range after a component name, the range **(1:1)** is implied. When you drag a type to make it a component, it is automatically created with the default component range of **(1:1)**.

## Defining components

Most groups need at least one component. Categories do not have components, however, you can define components for a category for inheritance purposes.

To determine the components of a group, ask the question: *This group type consists of what?*

For example, "The file consists of what?" The answer might be "records". So, **Record(s)** is a component of that group or category type.

Suppose you have a data file containing order records for an office supply store. You need to define the components of the type **File**.

## Guidelines for defining components

The guidelines below are numbered to allow references to each other. The numbering does not suggest a priority or a sequence to be followed in observing the guidelines.

1. Categories cannot have components.

A category is only used for organizing your schema and for setting common properties.

2. A partitioned group cannot have components.

A partitioned group always represents a choice among the subtypes of that group. You never map a partitioned group without its subtree, so it does not need components.



**Note:** A *subtree* is a branch of a schema that includes a type and all of the subtypes that stem underneath it.

3. If a group is not partitioned, it must have at least one component.

Nonpartitioned groups are sequences of data objects rather than choices. A sequence must contain at least one component.

4. A type and one of its subtypes cannot be in the same component list.
5. If a type has components, a subtype can inherit any of those components or any type in the subtree of one of those components.
6. If a type has no components, a subtype can inherit any type that could be in that type's component list.
7. A type that has an initiator and a terminator can have itself or one of its ancestors as a component.
8. A type cannot have one of its subtypes as a component.

## Complete type name

The complete name of a group is displayed in the title bar of its window. A type's complete name is similar to a path name, beginning at the type and including all types in the path up to the root. Spaces appear between types in a complete type name.

You can have duplicate type names as long as they are not on the same level. Each type has a unique complete name, so there is no doubt as to which particular type is being referenced.

## Relative type names

A component name is similar to a relative path name. Component names exclude types that the defined type and component type have in common in their complete names.

The relative type name excludes the names that appear in the complete names of both the type and the component type. For example, because **ROOT** is included in the complete type name of both **Row ROOT** and **Product Column ROOT**, it is excluded from the relative type name **Product Column**.

If more than one type in a schema has the same relative type name as another in a group, then the full path of the type is exported instead of the relative type name. No two types that have different full type names but identical relative type names can be added to a component list.

## Moving types with the same relative type name

In a schema, there could be two types with the same name that exist in different places within the tree. The relative type names could both evaluate to be the same, with respect to the component.

For example, the relative type **Group Category ROOT** might have a type named "Item" in both **ROOT** and **Category ROOT**. Both evaluate to "**Item**" with respect to the **Group** component. In this scenario, you can drag only one component named **Item** into the component list. An attempt to add a second component with the same name will be blocked.

## Ambiguous type names

A component name can refer to more than one type. In this situation, you must change type names so that the component name is no longer ambiguous. The schema designer does not allow components with the same relative type name to be added to component lists.

If you delete a type that is used as a component of another type, and then perform an "Undo", thus adding the type back, and the component name is ambiguous; the component name remains unresolved in the component list. In this scenario, the "Undo" operation does not result in a complete reversal of the action.

## Specifying minimum and maximum consecutive occurrences in the component list

The following table contains examples of how to specify in the component list the minimum and maximum consecutive occurrences for specific data objects:

Data Object	Min	Max	How to Specify
DateField	1	5	DateField (1:5)
DetailRecord	1	100	DetailRecord (1:100)
AddressField	2	3	AddressField (2:3)

## Fixed and variable ranges

Sometimes the range of a component is described as fixed or variable. As an example, a fixed range has the same minimum and maximum, (5:5). A variable range has a different minimum and maximum, (1:10) or (s) for example.

## Specifying a component range

### About this task

You can specify a component range by using Set Range to enter the minimum and maximum number of occurrences.

You can select multiple components and use **Set Range** to apply the same range to each component. A range of (1:s) means that there will always be at least 1 occurrence, but could occur an infinite number of times. A range of (0:s) means that an occurrence is optional, but could occur an infinite number of times.

## Variable component names

A component can refer to more than one type. To refer to all possible types whose names could appear at a certain place in the component name, use the word `ANY`.

The word `ANY` is like a wild card. It represents any type whose name could appear in that place.

The use of `ANY` is restricted to **Categories** and partitioned groups. For more information about using `ANY` in a partitioned group, see ["Partitioning" on page 705](#). For a list of reserved words and symbols, see the Design Studio Introduction documentation.

## Required and optional data

Sometimes, a certain data object is optional; it does not have to be present in the data. For example, in purchase order data, there might be a billing address and a shipping address. If the company wants the items shipped to the billing address, the shipping address would not appear in the data. The shipping address would be optional; it might not appear in the data.

Another example is a middle name field. Some people do not have a middle name, so the middle name field might be optional.

The schema designer needs to know what data is optional. This is evident from the component range. The range minimum tells how many occurrences of that object must be present in the data. These are the required occurrences. Optional occurrences are the ones that are not required.

For example, for the following component, the range minimum is zero. No occurrences must be present. It is optional data:

```
DateField (0:1)
```

Suppose the component looks like this:

```
DateField (1:5)
```

The range is between one and five occurrences. This means that one occurrence of **DateField** is required and the remaining four occurrences are optional.

The following table lists examples of components and explanations of their status.

Component	Status	Reason
LineItem (1:s)	1 occurrence required	Range minimum is 1



Component	Status	Reason
Note Field (5:5)	5 occurrences required	Range minimum is 5
RecordID	1 occurrence required	Range minimum is 1
ShipTo (0:1)	0 occurrences required (Optional)	Range minimum is 0
OrderRecord (s)	0 occurrences required (Optional)	Range minimum is 0

## Significance of required data

If you define an occurrence of a component as required, you are saying that, for the data containing the component to be *valid*, this component must exist. If it does not exist, the data is *invalid*.

For example, if you define **Record** as having the component **Field (3:3)**, you are saying that there must be three **Fields** in the **Record**. If there are not three **Fields**, then either it is a **Record** in error or it is not a **Record**.

There are other factors, besides the existence of all required components that make data valid. The existence of required components is necessary, but not sufficient, for data to be valid.

## Defining component rules

A component rule is an expression about one or more components. It indicates what must be true for that component to be valid. For given data, it evaluates to either "true" or "false". A component rule is similar to a test. If the data does not pass the test, it is invalid.

Component rules are used for validating data. Some important points about component rules are:

- Only components of a group can have component rules. Components of a category cannot have component rules.
- A component rule cannot be longer than 32K.
- If a component is optional and does not appear in the data, the component rule is evaluated after determining that the data is missing. In a component rule, you can specify relationships that depend on existence or nonexistence of data.

Sometimes components have relationships among each other. For example, an address field in purchase order data is preceded by a qualifier field which tells whether the address is a bill-to or a ship-to address. If the value of the qualifier field is `BT`, the address field following it is a bill-to address. If the value is `ST`, the following address is a ship-to address.

The qualifier and address fields are dependent on each other. They only make sense as a pair. If one of these fields is optional and is missing, the other field is not meaningful. If the qualifier is missing, you do not know whether the address is a bill-to or a ship-to. If the address is missing, the qualifier does not qualify anything!

You might want to define this kind of relationship and other relationships among data objects. To do this, use a component rule. For example, use a rule on the address field component to indicate that the qualifier must be present if the address is present.

## Examples of component rules

A component rule can limit the acceptable values of a component as shown in the following example:

```
Quantity < 10000
Interest Rate > .13 & Interest Rate < .20
WHEN (PurposeCode != "PF", ShipValue = 200|ShipCode < 0)
```

The following example illustrates how a component rule can make the presence of one component mandatory, if another component is present:

```
WHEN (PRESENT (Address Field), PRESENT (Qualifier Field))
WHEN (PRESENT (PhoneNumber), PRESENT (AreaCode), ABSENT (AreaCode))
```

A component rule can compare a component to the result of an arithmetic operation as shown in the following example:

```
SUM (((QuantityOrdered:Item Record:Detail) = TotalQuantity:Summary Record:Detail
Account Balance = Credits - Debits
Extension = Quantity*Price
#Items Field = COUNT (Item Record IN Invoice)
```

## Component rule syntax

A component rule is a complete expression that evaluates to either "true" or "false". It can contain functions (for example, PRESENT, COUNT, and SUM). It can also contain arithmetic operators (such as - + / \*).

A component rule is a statement, so it does not start with an equal sign.

For more information about the syntax of expressions, see the Functions and Expressions documentation.

## Entering object names in component rules

### About this task

A component rule can refer to a component within a component. The syntax for a component is the component name, followed by a colon (:), followed by the object of which the component is a part. Whenever you see the colon (:) in an expression, it can be interpreted as "component of" or simply "of".

For example, the following expression means *"The item number of the order record of the order"*:

```
Item#:OrderRecord:Order
```

All of the objects that can be used in component rules are shown in the group view. You can enter an object name into a component rule by pressing `Alt` and dragging the object into the edit area. The complete object name is automatically entered.

A component rule can refer to:

- The component it applies to and any nested components.
- Any component above the given component in the component list, and its nested components.

You can enter an object name in a component rule by typing it; however, the recommended method is to press `Alt` and drag the component into the edit area.

## Shorthand notation

In a component rule, the dollar sign (\$) represents the component itself. When you enter an object name in a rule by pressing `Alt` and dragging the object, the dollar sign is automatically entered in the rule to represent the given component.

## Component rules are context-sensitive

Component rules apply to components, not types. It applies to data in a certain context when the data is a component of a given group.

Suppose you have some order data that contains two kinds of records: a regular order record and a bulk order record. In the bulk order record, the quantity ordered must be greater than 1000. In the regular order record, the quantity can be any number. You could put a rule on the quantity ordered component of a bulk order, but not on the quantity ordered of a regular order.

## Special characters in component rules

### About this task

To enter the actual value of a special character in a component rule, you must enter the Hex value for one of the characters. For example, to enter the text value `<WSP>`, enter the Hex value for the less than sign `<<3C>>`, and then the rest:

```
<<3C>>WSP>
```

See the Design Studio Introduction documentation for a list of non-printable Hex and decimal values.

## Comments in component rules

### About this task

You can add comments to component rules. Comments do not affect how component rules are evaluated.

A comment begins with the characters `/*` and ends with the characters `*/`. A comment can appear anywhere in a rule as long as it does not separate object names.

For example, the following component rule has a comment:

```
SIZE (Phone# Field:$) >=7
/* Phone numbers must include area code */
```

## Component attributes

Attributes can be assigned to a component in a component list.

### Identifier attribute

The identifier attribute can be used on a component of a group. The identifier indicates the components that can be used to identify the type to which a data object belongs. All the components, from the first, up to and including the component with the identifier attribute, are used for type identification.

When this data is validated, it knows that, when it reaches the identifier, it has found a specific group. That group, therefore, is known to exist, even if part of the group following the identifier is missing. The map designer documentation discusses how data validation occurs.

The identifier attribute is also useful when identifying an object of a partitioned group. For information about using an identifier with partitioned data, see ["Partitioning" on page 705](#).

In a component list, there can be only one identifier attribute.

### Restart attribute

#### About this task

To continue processing your input data when a data object of a component is invalid, assign the restart attribute to that component.

Do not put the restart attribute on a required component. There must be a sufficient number of valid instances to cover all required components. If you have a required component that is not valid, the restart attribute does not validate the data.

For additional information about using the Restart attribute, see the map designer documentation.

### Sized attribute

The sized attribute is used on a component in which the value specifies the size (in bytes) of the component immediately following it. The sized attribute can be used on more than one component of a group.

For example, you might have a variable length component with a number immediately preceding it that indicates the length of the component: `10Washington`. The size of the component would be 10.

Some important points about using the sized attribute are:

- The component with the sized attribute must be defined as an unsigned integer.
- If a binary byte stream item does not have a fixed size, the component preceding it must specify its size and the sized attribute must be used on that component.

The size of a component is the number of bytes from the beginning of that component, up to and including the end of the component. If a component has a series range (such as [1:3]), the size includes all of the members in the series of that component. If a delimiter separates each member of that series, the delimiters must be included in the size. Also, if release characters appear in the component, they must be included in the size.

The size does *not* include delimiters that separate one component type from the next.

## Include self in size

If the value of the component with the sized attribute includes the size of the component, use the **Include Self in Size** option. For example, suppose the component with the sized attribute has a length of seven bytes. The value of the component with the sized attribute would be seven. So, its data would be one byte long, to hold the character seven. If its value includes the length of itself, its value would be  $7 + 1 = 8$ .

For example, a sized component is 7 bytes.

+ 1 byte (the length of the character 7)

If **Include Self in Size** is selected: 8 bytes

## Partitioning

By partitioning, you can define your data to distinguish the difference between data objects based on values in the data or differences in the syntax.

Partitioning is a method of subdividing objects into mutually exclusive subtypes. The partitioned type maintains the same class.

## Determining when to partition

There are some cases in which you will *need* to partition your data and others in which you will *want* to partition your data. You are *required* to partition for unordered data when a data object at a certain place in the data stream can be any number of types and each type has different definitions. *Choose* to partition for convenience, either to simplify rules or to put additional logic into your data's definition.

## Required partitioning

Partitioning is required when components are randomly or partially ordered. The following example represents unordered data:

BGI - 13100,REM,931104,19970424...

AXR - 10930,INV,003X114,19970422...

PVY - 19496,ORD,PO-104-1499,19970425...

BGI - 13100,ORD,PO-182-2587,19970425...

AXR - 10930,INV,003X-114,19970422...

PVY - 19496,REM,931104,19970424...

The file would contain three different types of transactions: **Invoice**, **Order**, and **Remittance**. Each transaction has a different definition based on the type of information it represents.

## Partitioning for convenience

You might decide to use partitioning in your schema to build additional logic into the definition of your data. You may also use partitioning to simplify the rules.

The following is an example of partitioning to simplify rules. The example compares the differences between rules needed *with* and *without* partitioning. In the rule without partitioning, you would specify a condition for each state abbreviation in each region. This could make your rules long, difficult to read, and difficult to maintain. The rule with partitioning is more concise, self-documenting, and easier to maintain.

Rule without partitioning:

```
=IF(ShipToCode Field::Input="NY"|
  ShipToCode Field::Input="NJ"|
  ShipToCode Field::Input="PA",
  F_MapEast (Record:Input), NONE)
```

Rule with partitioning:

```
=F_MapEast (EXTRACT (Record:Input,
PARTITION (ShipToCode Field::Input, East)))
```

## Benefits of partitioning

Using the [previous example on page 706](#), explore the benefits of partitioning.

- The rule with partitioning is shorter than the rule without partitioning. The knowledge of which states belong to each region is maintained in the type tree rather than the rule.
- It is easy to read this rule and understand the mapping function being performed. For example, if the state belongs to the list of states in the eastern region, execute the **MapEast** functional map.
- The partitioning method is easier to maintain. If a value for **State** is added or moves from one region to another, it can be easily changed in the schema and automatically reflected in any rules that reference the partitioned object.
- Partitioning using a restriction list used with the **Ignore Case** setting eliminates the need for **PROPER**, **LOWERCASE**, or **UPPERCASE** functions to compare each state with a literal.

## Consideration for HCL® OneTest™ Data

- Do not attempt to generate sample test data for a partitioned group. Sample test data should instead be generated for the specific subtype.

## Partitioning types

When data objects of different types appear in the same place in the data, the types must be distinguishable. This means that the data needs to be distinguishable by their definitions in the schema.

When the data object at any given point in the data may belong to any of a number of different types, there must be some way to tell the difference between them. To do this, you create a type and define a mutually exclusive subtype for each data object that may appear in the same place in the data. Once the subtypes are created, the subtypes also need to be distinguishable. Subtypes are distinguishable based on a value in the data or in the syntax of the different types.

## Partitioning items

Use one of the following three methods to partition items in schemas:

- Initiators
- Restrictions
- Format

## Partitioning an item type using initiators

To partition by initiator, each subtype must have an initiator and the value of the initiator must be unique for each subtype.

Partitioning by initiator is the most efficient method of partitioning.

## Partitioning an item type using restrictions

If an item has restrictions, you can partition that type, create mutually exclusive subtypes, and divide the restrictions between subtypes. A restriction cannot appear in more than one subtype of that item.

## Example of using restrictions

You have new data that needs to be entered into the Type Designer. Each new record contains the name of the employee and his or her department.

The **Employee List** schema illustrates components of **Record**.

The following is the new data containing the name of the employee and the department.

Steven Barlow,Doc

Heather Proust,Qa

Mary Whiting,Doc

Genie Elks,Sup

Francine Maxwell,Dev

Mark Brown,Sup

Daryl Schwartz,Acc

Harry O'Brian,Sal

Ellen Randolph,Dev

Paula Keller,Qa

Define the values of **Department** as a character text item.

Define the example data as valid restrictions (enter in the **Include** column) of **Department**:

If the departments are located in different offices, you can divide the data into separate files; one file per office. To do this, map the data from the main office to one file, the data from the development office to one file, and the data from the support office to another file. Then create subtypes of **Department**: **MainOffice**, **DevelopmentOffice**, and **SupportOffice** and partition **Department**. When you partition **Department** and create subtypes, you are saying that a given department data object belongs to only one of the subtypes based on its value.

The subtypes of **Department** inherit the restrictions of **Department**. Now allocate restrictions among subtypes. To do this, delete the restrictions from the subtype that do not apply to that particular office. For example, the **MainOffice** item has only the departments in that office, **DevelopmentOffice** item has only the departments in that office, and the **SupportOffice** item has only the departments in that office.

## Partitioning an item type by format

Subtypes that differ by their format are distinguishable from each other.

## Partitioning groups

Use one of the following three methods to partition groups in schemas:

- Initiators
- Identifiers
- Component rules



## Partitioning a group type using initiators

To partition by initiator, each subtype must have an initiator and the value of the initiator must be unique for each subtype.

The method of partitioning by initiators for group types is similar to item types.

## Partitioning a group type using identifiers

### About this task

In a component list, only one component can have the identifier attribute.

The identifier attribute distinguishes the components that can be used to identify the type to which a data object belongs. Typically, use this technique to distinguish group partitions when components following the identifier are different for each partition, or, if you have a multilevel partitioned subtree. In the latter case, using an identifier accelerates data validation.

A partition is valid when each component up to and including the identifier is validated. If the set of components is valid, the partition exists.

If the identifier set of components is not valid, the partition is determined not to exist. Either validation occurs for the next partition at the same level (if there is one) or it is determined that the partitioned group does not exist.

If the partition exists, what occurs next depends on the position of the partition type in the subtree.

- If the partition is partitioned (that is, it has subtypes), the rest of the components are skipped and the process begins to validate subtypes until a subtype is valid, exists and is in error, or does not exist.
- If the partition has no subtypes, the remaining components are validated. If all remaining components are valid, the partition not only exists, but also is valid. If one or more components are found to be in error, the partition exists, but its type is in error. If the partition exists, but its type is in error, the error is propagated back up the partitioned subtree until the group being validated is reached. When a partition is found to exist, the system will not continue to search for partitions.

To specify a component as the identifier:

1. From the schema editor, double-click a component.

#### Result

The Component view opens.

2. In the Component column, right-click on the component and choose **Identifier** from the context menu.

### Results

The identifier symbol appears in the component column next to the component name.

## Partitioning a group type using component rules

Component rules are used to partition data when a value or range of values can be used to distinguish one partition from another.

## Type inheritance

Properties, components, and restrictions of a type can be inherited by the types created as subtypes under it. Some properties of a type can also be propagated to already existing subtypes.

When you create a type, it becomes a subtype of whatever type is selected at the time. Everything that defines a type gets passed down: properties (with the exception of the Partitioned property), components, and restrictions. After a type is created, you can then modify any aspect of its definition.

When a group is created within a category, the item properties do not apply, so they are not inherited. When an item is created under a category, the group properties are not inherited.

## Inheritance of item properties and restrictions

Properties and restrictions of items are inherited when a new item is created.

For example, an item named **Department** is defined as a character text item that has a content size minimum of 2 and maximum of 3. Department has a list of "include" restrictions that consists of valid departments: ACC (Accounting), SLS (Sales), and MKT (Marketing).

The type **MainOffice** is created as a subtype of **Department**. **MainOffice** inherits the properties and restrictions of **Department**.

You can delete any restrictions that are not applicable to **MainOffice**.

## Inheritance of category properties and components

Categories can be used for organizing types and for inheritance reasons. Generally, you would use categories when you want to put items, groups, and possibly other categories under it as subtypes.

## Organizing types under a category

If you have two tables defined in one schema, you could divide the types of the two tables into different categories. A benefit of using a category type is that you can have any class of subtype: groups, items, and other categories.

## Using categories for inheritance

The properties of a category include group and item properties. Assign properties to a category that you want types beneath the category to inherit.

Any group created under a category inherits the property of the category. Any item created under a category inherits the category's item properties. A category created under a category inherits both the group and item properties. Each type created under a category inherits the other properties of the category, such as initiator and terminator.

As an example, if most groups in your **LabInfo** data are infix delimited with ~ and most items in your **LabInfo** data are unsigned integers, you can define these as the properties of the category **LabInfo**. Any types you create under **LabInfo** inherit its properties.

You can change the properties of the root type, which is a category. Categories can have components, so you can also use them for inheriting components.

## When not to use categories

It is best not to use a category type when the subtypes will specifically be item types or specifically be group types.

For example, you have a type named **Field** for which the only subtypes will be items. If you make **Field** a category type, each time you create a subtype, the new type will also be a category type by default. You will have to change the **Class** property from **Category** to **Item** for each subtype that you create. Instead, make **Field** an item type so that the **Class** property for any subtypes you create will automatically be defined as item types.

## Propagation of type properties

Creation time is not the only time type properties can be passed from a type to its subtypes. You can use propagation to pass along certain type properties.

*Propagation* passes the setting of a particular type property from the given type to all of the types in the subtree as applicable and defined by their context. When you select a type property for propagation that does not apply to a given subtype, it is not propagated.

## Propagation of type properties common to all types

You can propagate some type properties that are common to all types such as **Item Subclass**, **Description** and **Type Syntax**, to each of its subtypes, without the process checking additional criteria to determine if the type property can be propagated.

The type properties that fall into this category are the properties in the root of the tree, or are children properties that apply to all types to which their parent property is set, and are global in that these properties apply to all items.

The propagation process checks that the settings for some type properties such as the **Item Subclass** property, the **Interpret as** property, and if applicable, the **Presentation** property in a type, match the settings for these same properties in its subtypes, and if they match, the selected property is propagated. But, the propagation process does not check the settings of other properties to determine if the selected property exists in the item, because the selected property in this case, applies to all item types.

## Examples of propagating a type property that has no children properties

A given category type that is in the root of the tree has the following type property setting: **Description > An example**. The **Description** type property has no children properties. When you select the **Description** type property setting of **An example** to be propagated, the process propagates the **Description** setting of **An example** to all types in the tree. The **Description** property is a type property that is common to all types.

A given category type has the following type property settings: **Item Subclass > Text** and for its child property, **Interpret as > Character**. When you select the child **Interpret as** type property to be propagated, the process propagates the **Interpret as** setting of **Character** to each subtype of that category that also has the same **Text** type property setting. The **Interpret as** property is a type property that is common to all types except syntax types.

## Examples of propagating a type property that has children properties

A given category type has the following type property settings: **Item Subclass > Number**, and for its children properties, **Interpret as > Binary**, and **Presentation > Integer**. When you select the parent **Item Subclass** type property to be propagated, the process propagates the **Item Subclass** setting of **Number**, as well as the settings for its children properties, **Interpret as** setting of **Binary**, and the **Presentation** setting of **Integer**, to each subtype of that category. The **Item Subclass** property is a type property that is common to all types.

A given category type has the following type property settings: **Item Subclass > Text** and for its child property, **Interpret as > Character**. When you select the parent **Item Subclass** type property to be propagated, the process propagates the **Item Subclass** setting of **Text**, as well as the setting for its child property, **Interpret as** setting of **Character**, to each subtype of that category. The **Item Subclass** property is a type property that is common to all types.

## Propagation of type properties for specific types

You can propagate some other type properties such as **Separators** and **Restrictions**, to each of its subtypes for specific types, with the process checking additional criteria to determine if the type property can be propagated.

The type properties that fall into this category are children properties that do not apply to all types to which their parent property is set, and are specific to a type, in that these properties do not apply to all item types.

The propagation process checks that the settings for some type properties such as the **Item Subclass** property, the **Interpret as** property, and if applicable, the **Presentation** property in a type, match the settings for these same properties in its subtypes, and if they match, the selected property is propagated. But, depending on the property being propagated and the context, the process also checks other property settings. If you are propagating a child property, depending on the context, the process also checks the settings of the parent property, because the selected property might only apply to specific item types.

## Examples of propagating a type property that applies to a specific type

A given category type has the following type property settings: **Item Subclass > Number**, and for its children properties, **Interpret as > Character**, and **Presentation > Decimal**. When you propagate the **Separators** child property, the process affects only types in the schema that also have those same **Number**, **Character**, **Decimal** type property

settings. The **Separators** property is a type property that applies to the specific **Presentation** types, **Decimal** and **Integer**. For those types that have the **Presentation** type property setting of **Zoned**, the **Separators** property does not apply; it is not an available type property.

A given category type has the following type property settings: **Restrictions > Value** and for its child property, **Ignore case > No**. The **Ignore case** type property has no children properties. When you select the child **Ignore case** type property setting of **No** to be propagated, the process propagates the **Ignore case** setting of **No** to all types in the category whose parent **Restrictions** property setting is **Value**. The **Ignore case** property is a type property that applies to a specific **Restrictions** type, **Value**. For those types that have the **Restrictions** type property setting of **Range**, the **Ignore case** property does not apply; it is not an available type property.

## Schema analyzer

Use the schema analyzer to analyze your type definitions.

The schema analyzer analyzes type definitions and ensures internal consistency. For example, if you defined a group as fixed, but accidentally defined one of its items with no **Padded To** length, errors occur during analysis.

The analyzer checks your data definitions for logical consistency. It does not compare your definitions to your actual data. The resulting analyzer messages indicate whether your schema definitions are acceptable; not whether they match your data.



**Note:** The HCL® OneTest™ Data does not support schema analysis of JSON or XSD format schema.

## Mapping effects

The analyzer helps you define your data by locating objects in your input data and creating the objects in your output data. The analysis indicates whether there is something in your definition that may prevent a correct mapping of your data.

If you do not analyze a tree before you map the data or you analyze a tree and do not resolve the errors, you will be warned that you may receive unpredictable results when you map.

## Internal consistency

The analyzer checks the logic of your data definitions. For example, suppose you defined a group **PO** as fixed and consists of **Line(s)**. For the **PO** to be fixed, it cannot have an indefinite number of **Lines**. An analysis error would occur, indicating that you defined **PO** as fixed but it has a variable number of components.

## Logical analysis

Logical analysis addresses the integrity of the relationships that you define. Logical analysis detects, for example, undefined components, components that are not distinguishable from one another, item restrictions that do not match the properties of that item, and circular type definitions. The analyzer also checks delimiter relationships to each other and to components, undefined inherited relationships, and logic errors contained in component rules.

## Structural analysis

Structural analysis addresses the integrity of the underlying database. Generally, you should not encounter structural analysis errors. Structural analysis might be able to detect and possibly correct defects caused by system environment failures.

## Error and warning messages

Analysis results might contain error and warning messages.

Warnings are relatively insignificant. They indicate an inconsistency that occurred when you changed something in the tree that was resolved. For example, if you change a group to an item, the analyzer removes the components of that type because items do not have components. To remind you of this change, the analyzer issues a warning.

Errors are important. An error is a problem in your type definitions that you should correct. An error may result in unpredictable results in your mapping.

Occasionally there are errors that prevent the analysis of the rest of your definitions (for example, when a component type cannot be found in the tree). `Analysis halted before completion` is displayed. This error might be due to one of the following:

- Undefined COMPONENT found: ending analysis.
- Circular reference found in COMPONENT list: ending analysis.

When this occurs, correct the errors and analyze the tree again.

## Distinguishable objects

Differences can be identified for objects in a data stream. This information is helpful when a schema analysis produced an error message concerning objects that are not distinguishable.

## Objects in a data stream

A data stream is a byte-by-byte flow of data. Objects in a data stream include both data objects and syntax objects. Syntax objects indicate where a data object begins or ends. They include separators, initiators, terminators, delimiters, release characters, and pad characters. Sometimes, data values are used to identify where another data object begins or ends.

It might be necessary to distinguish between data objects in a component series, data objects of different types, or even syntax objects.

## Schema analyzer and distinguishable objects

The schema analyzer indicates whether your data definitions are sufficient to distinguish the objects in your data stream. The following discussion explains how you can define your data so that the objects that need to be

distinguishable are distinguishable. Chances are, if you analyzed your tree and received a message that involves distinguishable objects, you need to define the types differently or more specifically.

## Bound types

A type is bound if its definition makes it clear where an instance of that type ends. If a type is bound, different objects of that type can be distinguished in a data stream. A bound type is easier to distinguish between an object of that type and an object of another type. The following tables describe how types may be bound.

### **An object of this type:**

#### **Is bound if any of the following is true:**

##### **Item**

It is padded to a fixed size or its minimum and maximum content size are equal.

It has a terminator.

It has an include restriction list.

##### **Partitioned item**

Each non-partitioned item in the subtree is bound.

##### **Sequence Group**

It has an explicit fixed format.

It has a terminator.

Its last component is bound.

It is postfix delimited and its last component has a fixed range. For example, **Comment Field (3:3)** has a fixed range of **(3:3)**.

##### **Partitioned Group**

Each non-partitioned group in its subtree is bound.

##### **Choice Group**

It has a terminator.

The type of each selection component is bound.

##### **Unordered Group**

It has a terminator.

## Bound components

A component is bound if it is possible to tell if a data object belongs to that component without comparing it to a component that follows it.

## Component of a fixed group

### Condition

#### Example

**A range maximum is specified (it is not "s"), and its type is either a fixed group or an item whose length is fixed.**

The component **InventorySection (0:3)** is bound because it is assumed there will be spaces in the data stream for 3 **InventorySections**.

## Component of an explicit delimited group

### Condition

#### Example

**A range maximum is specified (it is not "s") if a component of the same group follows.**

In the component list, **Security Sequence (0:10) Trailer** is bound in an explicit delimited group. It is assumed there will be a delimiter for all 10 **Security Sequences** in the data stream because **Trailer** is required.

**The component is the last one and the explicit group has a terminator.**

In the component list, **Security Sequence (s)** is bound in an explicit delimited group with a terminator. The system assumes the terminator will appear to indicate that there are no more **Security Sequences**.

## Component of an implicit group

### Condition

#### Example

**Its range is fixed and its type is bound.**

The component **Inventory Record (3:3)** is bound, if the type **Inventory Record** is bound according to any of the conditions listed under ["Bound Types" on page 715](#) .

**It has a component rule that binds it.**

The component **PO Record (s)** is bound, if it has the following component rule which binds it:

```
PO# Field:PO Record = PO# Field:PO Record[LAST]
```

The schema analysis checks only that the component has a rule that refers to the component itself. The analysis does not check that the rule binds the component. You must ensure that the rule is one that binds the component.

**It is sized (using the Sized attribute) by the component that precedes it.**

The component **Name Field (0:2)** is bound, if the previous component in the group has the Sized attribute.



## Component of a choice group

### Condition

#### Example

**A component is bound if the type of a component is bound.**

The component **Customer Record** is bound if the type **Customer Record** is bound.

## Component of an unordered group

### Condition

#### Example

**A component is bound if its range is fixed and its type is bound.**

The component **Address Field** is bound if its range is fixed, for example **(3:3)**, and the type **Address Field** is bound.

## Group starting set

Data objects in a data stream need to be distinguishable when they could belong to two different groups. Specifically, the difference between the data that can come first in one group and the data that can come first in the other group. All of the possible types of data that may appear first in a group are referred to as the group's starting set.

The following describes the starting set of a group based on its group format:

Group Format	Starting Component Set
<b>Explicit</b> <ul style="list-style-type: none"> <li>• Delimited</li> <li>• Fixed</li> </ul>	Includes the type of its first component.
<b>Implicit</b> <ul style="list-style-type: none"> <li>• Sequence</li> <li>• Unordered</li> </ul>	<ul style="list-style-type: none"> <li>• Includes the type of all components up to and including the first component that has a minimum range of at least one.</li> <li>• Includes the type of each component.</li> </ul>
<b>Choice</b>	Includes the type of each component.

## Group unbound set

Based on the nature of a bound group, the end of a bound group is determined without analyzing the data that follows it. However, if a group is not bound, the data that belongs to that group must be distinguishable from data that belongs to another type that might follow it in the data stream.

The unbound set of a choice or unordered group consists of the type of each component. The unbound set of a partitioned group consists of the unbound set of each unbound partition.

## Unbound set of a sequence group

The unbound set does not include a type that could come last, if that type is a required occurrence. For example, if the type **Comment Record** could appear last and the component is specified as **Comment Record (2:2)**, it is not in the unbound set, because the two occurrences of **Comment Record** are required. However, if **Comment Record** could appear last and the component is specified as **Comment Record (1:2)**, then **Comment Record** is in the unbound set, since its second occurrence is optional.

To determine the unbound set of a group, start with the last component of the group and proceed backward up the component list. If a component is unbound or has a minimum range of zero, continue up the list. Stop at the first component that is bound or that is unbound but has a minimum range greater than zero. Essentially, a group's unbound set is everything that is not clearly defined at the end of the group.

## Initiator-distinguishable types

Initiator-distinguishable types are used during validation to determine existence of a type object based on the existence of its initiator.

## Determining if a component is initiator-distinguishable from its following set

A component is initiator-distinguishable from its following set if:

- The component is not a member of the identifier set and
- The type of the component has an initiator and
- The following set is empty or
- The type of the component is initiator-distinguishable from each type in its following set.

Schemas are analyzed to determine if components are initiator-distinguishable. Each implicit sequence group, choice group, and unordered group is analyzed to determine if their components are initiator-distinguishable. A component is marked as initiator-distinguishable when that component is initiator-distinguishable from its following set. The basis for this determination is found in ["Determining If Two Types are Initiator-Distinguishable" on page 719](#) .

## Determining if a partition is initiator-distinguishable from its following set

In a partitioned type, a partition is initiator-distinguishable from its following set if:

- The type of a partition has an initiator and
- The following set is empty or
- The type of the partition is initiator-distinguishable from each partition in its following set and the following set of a partition is the type of each partition that may follow.

Schemas are analyzed to determine if partitions are initiator-distinguishable. Each partitioned type is analyzed to determine if its partitions are initiator-distinguishable.

## Determining if two types are initiator-distinguishable

The following table lists ways two types may be initiator-distinguishable. This is helpful if data validation errors indicate a type does not exist.

Type1	Type2	How to define them as initiator-distinguishable
item	item	If Type1 and Type2 have an initiator, and the initiators are different.
item	sequence group	Either: <ul style="list-style-type: none"> <li>• Type1 and Type2 both have an initiator and the initiators are different.</li> </ul> or <ul style="list-style-type: none"> <li>• Type1 has an initiator, Type2 does not, Type2 has no delimiter, and Type1 is initiator distinguishable from type of each component in the starting component set of Type2.</li> </ul>
item	choice group or unordered group	Either: <ul style="list-style-type: none"> <li>• The Type1 and Type2 both have an initiator and the initiators are different.</li> </ul> or <ul style="list-style-type: none"> <li>• Type1 has an initiator, Type2 does not, Type2 has no delimiter, and Type1 is initiator distinguishable from type of each component of Type2.</li> </ul>
item	partitioned item or partitioned group	Type1 has an initiator and is initiator distinguishable from each partition of Type2.
partitioned item	item or sequence group	Type1 has an initiator and each partition is initiator distinguishable from Type2.

Type1	Type2	How to define them as initiator-distinguishable
partitioned item	partitioned item or partitioned group	Type1 has an initiator and each partition is initiator distinguishable from each partition of Type2.
sequence group	item	Type1 has no identifier, Type1 and Type2 both have initiators, and the initiators are different.
sequence group	sequence group	<p>Type1 has no identifier and</p> <ul style="list-style-type: none"> <li>• Type1 and Type2 both have initiators and the initiators are different</li> </ul> <p>or</p> <ul style="list-style-type: none"> <li>• Type1 has an initiator, Type2 does not have an initiator, Type2 has no delimiter, and Type1 is initiator distinguishable from the starting component set of Type2.</li> </ul>
sequence group	choice group or unordered group	<p>Type1 has no identifier and</p> <ul style="list-style-type: none"> <li>• Type1 and Type2 both have initiators and the initiators are different.</li> </ul> <p>or</p> <ul style="list-style-type: none"> <li>• Type1 has an initiator, Type2 does not have an initiator, Type2 has no delimiter, and Type1 is initiator distinguishable from the type of each component of the Type2.</li> </ul>
sequence group	partitioned item	Type1 has no identifier and Type1 must be initiator distinguishable from each partition of Type2.
sequence group	partitioned group	Type1 has no identifier and the Type1 must be initiator distinguishable from each partition of Type2.
partitioned group	partitioned item	Type1 has no identifier and each partition Type1 must be initiator distinguishable from Type2.
partitioned group	sequence group	Type1 has no identifier and each partition of the Type1 must be initiator distinguishable from Type2.
partitioned group	partitioned item	Type1 has no identifier and each partition of Type1 must be initiator distinguishable from each partition of Type2.
partitioned group	partitioned group	Type1 has no identifier and each partition of Type1 must be initiator distinguishable from each partition of Type2.

Type1	Type2	How to define them as initiator-distinguishable
choice group or un-ordered group	item	Type1 and Type2 both have an initiator and the initiators are different.
choice group or un-ordered group	partitioned item	Type1 is initiator-distinguishable from each partition of Type2.
choice group or un-ordered group	choice group or un-ordered group	Either: <ul style="list-style-type: none"> <li>• Type1 and Type2 both have an initiator and the initiators are different.</li> </ul> or <ul style="list-style-type: none"> <li>• Type1 has an initiator and Type2 does not, Type2 has no delimiter, and Type1 is initiator-distinguishable from the type of each component of Type2.</li> </ul>
choice group or un-ordered group	sequence group	Either: <ul style="list-style-type: none"> <li>• Type1 and Type2 both have an initiator and they are different.</li> </ul> or <ul style="list-style-type: none"> <li>• Type1 has an initiator and Type2 has no initiator and no delimiter, and Type1 is initiator-distinguishable from the type of each component in the starting component set of Type2.</li> </ul>
choice group or un-ordered group	partitioned group	Type1 is initiator distinguishable from each partition of the Type2.

## Distinguishable objects of the same component

When a component has a series range, for example, (1:10) or (s), one occurrence of that component must be distinguishable from the next occurrence of that same component.

Different data objects of a component are distinguishable if any of the following is true:

- The component type is bound.
- The component type is a non-partitioned group with an unbound set that is content distinguishable from the component type as a whole.
- The component type is a partitioned group and the unbound set of each non-partitioned group in its subtree is content-distinguishable from the component type as a whole.

## Content-distinguishable components

A component is distinguishable from its following set when the component is:

- Initiator-distinguishable from its following set.

or

- Content- distinguishable from its following set.

A component is content-distinguishable from its following set when:

- The following set is empty.

or

- The type of the component is content-distinguishable from each component in its following set.

## Content-distinguishable types

The following table lists the ways in which two types may be content-distinguishable. The table is helpful particularly if you analyzed your tree and received an error indicating that two types are not distinguishable. Look up the combination of types in the first two columns and read the list of ways to define them that would make them distinguishable.

For example, if you get an error indicating that **X** is not distinguishable from **Y**, where **X** is an item, and **Y** is a partitioned group, you would find the row in the table where **Type1** is an item and **Type2** is a partitioned group.

Remember that the order in which you compare two types matters. When you ask the question *"Is type A content-distinguishable from type B?"*, this is not the same question as *"Is type B content-distinguishable from type A?"*

If you are trying to determine whether two types are distinguishable and you follow the guidelines in the following tables, you may encounter a situation in which you are comparing a type to itself. A type is never distinguishable from itself.

Another thing to keep in mind is that the context in which a type is used matters. When you ask the question, "Is type A, as a component of type C, content-distinguishable from type B?" this is not the same question as, "Is type A, as a component of type D, content-distinguishable from type B?"

The following table shows how to define types as content-distinguishable.

Type 1	Type 2	How to define them as content-distinguishable
item	item	<p>The first component or partition is either:</p> <ul style="list-style-type: none"> <li>• An item and marked as initiator distinguishable and</li> </ul> <p>Both items are different partitions of the same partitioned subtree, or The second type has an initiator and those initiators are mutually exclusive, or Both types have the same initiator and the value of the first item is distinguishable from the value of the second item, or The second type has no initiator and the initiator of the first type is distinguishable from the value of the second type.</p> <ul style="list-style-type: none"> <li>• An item and not marked as initiator distinguishable and</li> </ul> <p>Both items are different partitions of the same partitioned subtree, or The first item has an initiator, second has an initiator and initiator value of first is distinguishable from initiator value of the second. Both types have initiators and they are the same, or both types have no initiator and the value of the first item is distinguishable from the value of the second item. The first item has an initiator, second does not, and value of initiator distinguishable from the value of second item. The first item has no initiator, second does, and the first item's value is distinguishable from the value of the second item's initiator.</p>
item	sequence group, choice group, or unordered group	<p>The first component or partition is either:</p> <ul style="list-style-type: none"> <li>• An item marked as initiator distinguishable and</li> </ul> <p>The second type has an initiator and those initiators are mutually exclusive, or The second type has no initiator and the item is initiator-distinguishable from each type in the starting component set of the group.</p> <ul style="list-style-type: none"> <li>• An item not marked as initiator distinguishable and</li> </ul> <p>The item and group both have an initiator and the initiator of the item is distinguishable from the initiator of the group, or The item has no initiator, the group has an initiator, and the item's value is distinguishable from the value of group's initiator, or Both types have initiators and they are the same or both types have no initiator, and the item's value is distinguishable from the type of each component in the starting component set of the group, or The</p>

Type 1	Type 2	How to define them as content-distinguishable
item	partitioned item or partitioned group	item has an initiator, the group has no initiator, and the item is content-distinguishable from the type of each component in the starting component set of the group, or The item has no initiator, the group has an initiator, and the item's value is distinguishable from each type in the starting component set of the group.
partitioned item	item	Each partition of the first type is content-distinguishable from the second type.
partitioned item	sequence group	Each partition of the first type is content-distinguishable from the second type.
partitioned item	choice group	Each partition of the first type is content-distinguishable from second type.
partitioned item	unordered group	Each partition of the first type is content-distinguishable from second type.
partitioned item	item	Each partition of the first type is content-distinguishable from second type.
partitioned item	partitioned group	Each partition of the first type is content-distinguishable from second type.
group	item	Each partition of the partitioned group is content-distinguishable from the item.
partitioned group	partitioned item	Each partition of the partitioned group is content-distinguishable from each partition of the item.
partitioned group	sequence group	Each partition of the partitioned group is content-distinguishable from the second group.
partitioned group	partitioned group	Each partition of the first partitioned group is content-distinguishable from each partition of the second partitioned group.
partitioned group	choice group	Each partition of the partitioned group is content-distinguishable from the choice group.
partitioned group	unordered group	Each partition of the partitioned group is content-distinguishable from the unordered group.
sequence group	item	The first component is either: <ul style="list-style-type: none"> <li>• A group marked as initiator distinguishable and</li> </ul>



Type 1	Type 2	<b>How to define them as content-distinguishable</b>
sequence group	sequence group	<p>The item has an initiator and the initiator value of the group is distinguishable from the initiator value of the item, or The second type has no initiator and the initiator value of the first type is distinguishable from the value of the second type.</p> <ul style="list-style-type: none"> <li>• A group not marked as initiator distinguishable and</li> </ul> <p>The group has an initiator, the item has an initiator and the initiator value of the group is distinguishable from initiator value of the item, or The group has an initiator, the item does not, and value of the initiator is distinguishable from value of the item, or Both types have initiators and they are the same, or both types have no initiator and each type in the starting component set of the group is content-distinguishable from the value of the second item.</p> <p>The first component is either:</p> <ul style="list-style-type: none"> <li>• A group marked as initiator distinguishable, and</li> </ul> <p>Both groups are different partitions of the same partitioned subtree, or The second type has an initiator and those initiators are mutually exclusive, or The second type has no initiator and the first group is initiator-distinguishable from each type in the starting component set of the second group.</p> <ul style="list-style-type: none"> <li>• A group not marked as initiator distinguishable, and</li> </ul> <p>Both groups are different partitions of the same partitioned subtree, or Both groups have an initiator and the initiator value of the first group is distinguishable from the initiator value of the second group, or The first group has no initiator, the second group has an initiator and the type of each component in the starting component set of the first group is content-distinguishable from the second group, or The first group has an initiator, the second group has no initiator and the first group is content-distinguishable from the type of each component in the starting component set of the group. Both types have initiators and they are the same or both types have no initiator and The first group is content-distinguishable from the starting component set of the second group, or The second group is content-distinguishable from the starting component set of the first group, or The starting component set of the first group is con-</p>

Type 1	Type 2	How to define them as content-distinguishable
sequence group	choice group or group	<p>tent-distinguishable from the starting component set of the second group.</p> <p>The first component is either:</p> <ul style="list-style-type: none"> <li>• A group marked as initiator distinguishable and</li> </ul> <p>Both groups are different partitions of the same partitioned subtree, or The second type has an initiator and those initiators are mutually exclusive, or The second group has no initiator and the first group is initiator-distinguishable from the type of each component of the second group.</p> <ul style="list-style-type: none"> <li>• A group not marked as initiator distinguishable and</li> </ul> <p>Both groups are different partitions of the same partitioned subtree, or Both groups have an initiator and the initiator of the first group is distinguishable from the initiator of the second group, or The first group has an initiator, the second group has no initiator and the first group is content-distinguishable from the type of each component of the second group, or The first group has no initiator the second group has an initiator, and each component in the starting component set of the first group is distinguishable from the second group, or Both types have initiators, either the initiators are the same or both types have no initiator, and each component in the starting component set of the first group is content-distinguishable from the type of each component of the second group.</p>
sequence group	partitioned item or partitioned group	<p>The first component is either:</p> <ul style="list-style-type: none"> <li>• A group marked as initiator-distinguishable and the group is initiator-distinguishable from each partition of the second type, or</li> <li>• A group not marked as initiator-distinguishable and the group is content-distinguishable from each partition of the second type.</li> </ul>
choice group or unordered group	item	<p>The first component is either:</p> <ul style="list-style-type: none"> <li>• A group marked as initiator distinguishable and</li> </ul> <p>The item has an initiator and the initiator value of the group is distinguishable from the initiator value of the item, or</p>

Type 1	Type 2	<b>How to define them as content-distinguishable</b>
choice group or group	sequence group	<p>The second type has no initiator and the initiator value of the first type is distinguishable from the value of the second type.</p> <ul style="list-style-type: none"> <li>• A group not marked as initiator distinguishable and</li> </ul> <p>Both types have an initiator and the initiator value of the group is distinguishable from initiator value of the item, or</p> <p>The group has an initiator, the item does not, and the value of the initiator is distinguishable from value of the item, or</p> <p>The group has no initiator, the item has an initiator and the starting component set of the group is content-distinguishable from the item, or</p> <p>Both types have initiators and they are the same, both types have no initiator and the type of each component of the group is distinguishable from the value of the second item.</p>
		<p>The first component is either:</p> <ul style="list-style-type: none"> <li>• A group marked as initiator distinguishable and</li> </ul> <p>Both groups are different partitions of the same partitioned subtree, or The second type has an initiator and those initiators are mutually exclusive, or The second type has no initiator and the first group is initiator-distinguishable from each type in the starting set of the second group.</p> <ul style="list-style-type: none"> <li>• A group not marked as initiator distinguishable, and</li> </ul> <p>Both groups are different partitions of the same partitioned subtree, or Both groups have an initiator and the initiator value of the first group is distinguishable from the initiator value of the second group, or The first group has an initiator, the second group has no initiator and the first group is content-distinguishable from the type of each component in the starting component set of the second group. The first group has no initiator, the second group has an initiator and the type of each component of the first group is content-distinguishable from the second group, or. Both types have initiators and they are the same or both types have no initiator, and the type of each component of the first group is content-distin-</p>

Type 1	Type 2	How to define them as content-distinguishable
choice group or group	choice group or group	<p>guishable from the type of each component in the starting component set of the second group.</p> <p>The first component is either:</p> <ul style="list-style-type: none"> <li>• A group marked as initiator distinguishable and</li> </ul> <p>Both groups are different partitions of the same partitioned subtree, or The second type has an initiator and those initiators are mutually exclusive, or The second group has no initiator and the first group is initiator-distinguishable from the type of each component of the second group.</p> <ul style="list-style-type: none"> <li>• A group not marked as initiator distinguishable and</li> </ul> <p>Both groups are different partitions of the same partitioned subtree, or Both groups have an initiator and the initiator of the first group is distinguishable from the initiator of the second group, or The first group has an initiator, the second group has no initiator and the first group is content-distinguishable from the type of each component of the second group, or The first group has no initiator the second group has an initiator, and each component of the first group is distinguishable from the second group, or Both types have initiators and the initiators are the same or both types have no initiator, and the type of each component of the first group is content-distinguishable from the type of each component of the second group.</p>
choice group or group	partitioned item or partitioned group	<p>The first component is either:</p> <ul style="list-style-type: none"> <li>• A group marked as initiator-distinguishable and the group is initiator-distinguishable from each partition of the second type, or</li> <li>• A group not marked as initiator-distinguishable and the group is content-distinguishable from each partition of the second type.</li> </ul>

## Ending-distinguishable types

Ending-distinguishability is used to determine if the end of a data object is distinguishable from the start of any other data object that could be next in the data.

The following table describes how two types may be ending-distinguishable. This is helpful if you are validating data and you receive a message that says a type exists, but it belongs to the wrong component.

<b>Type1</b>	<b>Type2</b>	<b>How to define them as ending-distinguishable</b>
item	item or sequence group or choice group or unordered group	Type1 is bound or the value of Type1 is content-distinguishable from Type2.
item	partitioned item or partitioned group	Type1 is bound or the value of Type1 is content-distinguishable from each partition of Type2.
sequence group	item or sequence group or choice group or unordered group	<p>Either:</p> <ul style="list-style-type: none"> <li>• Type1 is bound.</li> </ul> <p>or</p> <p>For each component in the unbound set of Type1</p> <ul style="list-style-type: none"> <li>• If the component has a fixed range it must be ending-distinguishable from Type2.</li> <li>• If the component range is variable, it must be content-distinguishable from Type2 and ending-distinguishable from Type2.</li> </ul>
sequence group	partitioned item or partitioned group	<p>Either:</p> <ul style="list-style-type: none"> <li>• Type1 is bound.</li> </ul> <p>or</p> <p>For each component in the unbound set of Type1</p> <ul style="list-style-type: none"> <li>• If the component has a fixed range it must be ending-distinguishable from each partition of Type2.</li> <li>• If the component range is variable, it must be content-distinguishable from each partition of Type2 and ending-distinguishable from each partition of Type2.</li> </ul>
choice group or unordered group	item or sequence group or choice group or unordered group	<p>Either:</p> <ul style="list-style-type: none"> <li>• Type1 is bound.</li> </ul> <p>or</p>

Type1	Type2	How to define them as ending-distinguishable
choice group or unordered group	partitioned item or partitioned group	<p data-bbox="857 260 1192 285">• For each component of Type1,</p> <p data-bbox="792 331 1398 436">The type of that component is bound, or The type of that component is unbound, and that type must be ending-distinguishable from Type2.</p>
		<p data-bbox="792 470 862 495">Either:</p> <ul data-bbox="857 541 1040 569" style="list-style-type: none"> <li data-bbox="857 541 1040 569">• Type1 is bound.</li> </ul>
		<p data-bbox="792 617 813 642">or</p> <ul data-bbox="857 688 1192 716" style="list-style-type: none"> <li data-bbox="857 688 1192 716">• For each component of Type1,</li> </ul>
		<p data-bbox="792 758 1398 863">The type of that component is bound, or The type of that component is unbound, and that type must be ending-distinguishable from each partition of Type2.</p>
partitioned item or partitioned group	item, sequence group, choice group, or unordered group	<p data-bbox="792 894 1154 919">For each partition of Type1, either:</p> <ul data-bbox="857 968 1398 1157" style="list-style-type: none"> <li data-bbox="857 968 1398 1031">• The partition is a partitioned type and ending-distinguishable from the second type, or</li> <li data-bbox="857 1045 1300 1073">• The partition of the first type is bound, or</li> <li data-bbox="857 1087 1398 1157">• The partition of the first type is unbound and ending-distinguishable from the second type.</li> </ul>
partitioned item or partitioned group	partitioned item or partitioned group	<p data-bbox="792 1209 1219 1234">For each partition of the first type, either:</p> <ul data-bbox="857 1283 1398 1476" style="list-style-type: none"> <li data-bbox="857 1283 1398 1346">• The partition is a partitioned type and ending-distinguishable from the second type, or</li> <li data-bbox="857 1360 1300 1388">• The partition of the first type is bound, or</li> <li data-bbox="857 1402 1398 1476">• The partition of the first type is unbound and ending-distinguishable from the second type.</li> </ul>

## Distinguishable data objects of an implicit group

When a data object belongs to an implicit sequence, determining the component a data object belongs to depends on the format of the sequence, the type definition of a component, and the component properties.

## Guidelines for defining an implicit delimited sequence

- If the type of a component has a terminator, that terminator must be different from the delimiter of the explicit group. If the delimiter and terminator are both defined as literal values, a type analysis confirms both have different values.
- The type of a component cannot be a binary item whose length is not fixed or sized.
- If the type of a component or a contained component is not bound, the type of the component cannot have a delimiter that is the same as the delimiter of the implicit sequence.
- If the range of a component is not bound, the type of that component must be content-distinguishable from the type of each component in the following set of that component.
- If the type of a component has both an initiator and terminator, the nested delimiters do not have restrictions. If this is not the case, all contained delimiters must be different.

## Guidelines for defining an implicit sequence that has no delimiter

- The type of a component cannot be a binary item whose length is not fixed or sized.
- If the range of a component is bound, but the type of the component is unbound, the type must be ending-distinguishable from each type in the component's following set.
- If the range of a component is not bound, all the following rules apply:
  - The type of the component must be content-distinguishable from each type in the component's following set.
  - If the maximum range for that component is greater than one, and the type is unbound, the type must be ending-distinguishable from itself.
  - If the type of the component is unbound, the type must be end-distinguishable from each type in the component's following set.

## Guidelines for defining an implicit unordered group that is delimited

- The type of a component cannot be a binary item whose length is not fixed.
- The type of a component must be content-distinguishable from the type of each other component.
- If the type of a component has both an initiator and terminator, the nested delimiters do not have restrictions. If this is not the case, all contained delimiters must be different.

## Guidelines for defining an implicit unordered group that has no delimiter

- The type of a component cannot be a binary item whose length is not fixed.
- The type of a component must be content-distinguishable from the type of each other component.
- If the type of a component is unbound, the type must be ending-distinguishable from the type of each other component.
- If the maximum range is greater than one and the type is unbound, the type must be ending-distinguishable from itself.

## Distinguishable data objects of an explicit group

When a data object belongs to an unordered group, distinguishing one data object from another depends on the format of the unordered group and the type definition of each component.

### Guidelines for defining an explicit fixed group

- In a fixed group, each component must be either an item of fixed size, an item padded to a fixed length, or an explicit fixed group.
- The range maximum for each component must have a specific value; it cannot be **s**. A fixed amount of space is assumed in the data stream for each series member of a component in a fixed group.

### Guidelines for defining an explicit delimited group

A component of an explicit delimited group can be a group or item, with the following restrictions:

- If the type of a component has a terminator, that terminator must be different from the delimiter of the explicit group. If the delimiter and terminator are both defined as a literal, analysis confirms if both have different values.
- If a component of an explicit delimited group is a binary item whose length is not fixed, that component must be sized by the component that precedes it.
- If the type of a component, or a contained component, does not have a terminator and the type of the component has a delimiter that is the same as the delimiter of the explicit group, the type of the component *must* be an explicit sequence. In this case, the delimiter appears as a placeholder for every data object if data of the same delimiter follows it.
- If an explicit group has a delimiter, the range maximum of any component other than the last one must have a specific value; it cannot be **s**. A delimiter is assumed in the data stream for each series member of a component of an explicit group with a delimiter.

### Objects of a choice group

When a data object belongs to a choice group, which component the data object belongs to is based on the order the components appear in the type definition.

In a choice group, if a component is unbound it must be content-distinguishable from the type of each component that follows it in the component list.

### Objects of a partitioned type

When data objects of different types appear in the same place in the data, the types must be distinguishable. The data must be distinguishable in the type definition. When a partition is part of a partitioned object, the process of cycling through the partitioned subtree begins to make the determination of which partition the data object belongs to.

In a partitioned type, each partition must be content-distinguishable from the type of each partition that follows it.



## Distinguishable syntax objects

The system must be able to distinguish between different syntax objects contained within a group. To ensure that the syntax objects are distinguishable, use the following guidelines:

- If a component is unbound, make sure its delimiter and any delimiter contained in it is distinguishable from the delimiter of the group. If you get an analysis error, look for missing placeholders or components with a range maximum of **s**.

In a component or a contained component with the same delimiter as the type delimiter, the system assumes that the delimiter appears as a placeholder for every data object if data with the same delimiter follows it.

- If the delimited group has a terminator, make sure that the delimiter and any contained delimiter is distinguishable from the terminator.

The schema analysis verifies if the above requirement is met in a non-partitioned group.

## Functions and expressions

### Expressions and evaluations

You can use functions and expressions to create component rules in the Type Designer.

The **Examples** directory, located in the product installation directory, contains type trees that can help you learn how to accomplish your tasks. These type trees and maps contain component rules that show the use of many functions and expressions.

### Expressions

An expression is a statement about data objects. Expressions are used in component rules in the schema designer and in map rules in the map designer.

The following are examples of expressions:

```
Account Balance = Credits - Debits
TrunkRoom < 15
PRESENT (Store#:StoreInfo)
```

As you enter an expression, you can add spaces before and after component names and operators to make the expression more readable. Additional spaces have no effect on an expression.

Expressions are a combination of literals, object names, operators, functions, and map names.

### Component rule expressions evaluate to true or false

Component rules are expressions that evaluate to "true" or "false". For example, the result of each of the following component rules is "true" or "false".

```
WHEN (PRESENT (Qualifier), PRESENT (Address))
COUNT (Exchange) < COUNT (Purchase)
Invoice#:Item = Invoice#:Item[1]
```

## Map rule expressions evaluate to data

Map rules are expressions that evaluate to data.

For example, the result of each of the following map rules is data:

```
= "Florida"
= Price:Input
= COUNT ( Name:Roster )
= RecordMap ( FixedRecord )
```

## Literals

A literal is a constant value. A literal may be a number or a text string.

The rules for using numeric literals are:

- A numeric literal is in integer or decimal format and can be signed.
- A comma separator should not be included in a numeric literal.
- A numeric literal cannot be greater than 254 digits.

The rules for using text literals are:

- A text literal is enclosed in double quotes.
- A double-quote included in a text literal is released by another double quote.

Literals are encoded in UTF-16 (Unicode).

The following are some examples of literals:

```
"This is a text literal"
"ABC Company"
"Some ""quoted data"" to show use of a double double-quote."
1.23
-9
1045
```

Carriage return/line feeds cannot be within a quoted string. To use a carriage return/line feed the string must be broken into two strings with <CR><LF> between them. For example:

```
= "username=?????" + "<CR><LF>" + "&password=?????"
```

will place the output username and password information on separate lines.

<CR><LF> must be within quotation marks.

## Data object names

A data object is referenced in an expression by its name. An object name can be as simple as a card name or a local type name. It can also be complex depending on the object you want to reference.

## Object names in map rules

In a map rule, data object names always refer to the card name that contains the data object name. So, in a map rule, a data object name ends with a card name. In the following example image, the object name for the component **Contact** of the card **ContactFile** is **Contact:ContactFile**.

The general definition of an object name in a map rule can be divided into three parts as described in the following table. If one of these parts is used, it must be sequenced according to the order in the table.

### Parts of an Object Name

#### Required

#### Component name

No

#### Set of component-paths, each ending with either a colon or the reserved word **IN**

No

#### Card name

Yes

When **IN** is used, there must be a space before and after it, such as `Dependent IN Claim`.

## Object names in component rules

In a component rule, data object names always refer to components in the same component list. In a component rule, a data object name ends with a component name. For example, if **LinItem** is in a component list, the object name for the **Qty Info** component of **LinItem** is **Qty Info:LinItem**.

In the preceding **Qty Info:LinItem** example, the colon would be interpreted as *"is a component of"*. If the example was **Qty Info LinItem**, the space between **Info** and **LinItem** would be interpreted as *"is a subtype of"*.

The general definition of a component rule object name can be divided into the two parts described in the following table. If one of these parts is used, it must be sequenced according to the order in the table.

### Parts of an Object Name

### Required

Set of one or more component-paths

At least one is required

, infix separated with either a colon or the reserved word, **IN**

The reserved word, **COMPONENT**

No

IN and COMPONENT are reserved words. Reserved words are not case sensitive.

## Card name

A map rule can reference the object of an entire card. A card name is a simple type name from 1 to 32 bytes in length in UTF-8 encoding. For example, if the input card name is **Invoice**, the name for the card object would be **Invoice**.

## Local type name

An object name can be a local type name. A local type name is specified as one or more simple type names separated by spaces, such as:

```
City Field
```

Local type names are typically used to refer to a component. If used in a component rule, for example, the component referenced in a rule is specified as a local type name. In a map rule, local type names refer to components contained in a card object.

Local type names can also be used to refer to the partitioned type of a component. When a type is partitioned, you can refer to all of its partitions by simply referring to the partitioned type.

For this example, the type **Record** is partitioned into **Header**, **Detail**, and **Trailer**. The object name that references **Header**, **Detail**, and **Trailer Record** is: **Record**

Here is another example: the type **Transcript** has **Header Record**, **Detail Record**, and **Trailer Record** as components contained within it. The object name that references all **Header**, **Detail**, and **Trailer Records** within **Transcript** would be **Record IN Transcript**.

## Partition list

A partition list is a set of partition names separated by the symbols <>. A partition name is the simple name that represents the partitions found under the parent, or partitioned, type. For example, the partition Header of the type Record would have this component path: **Header<>Record**

## Component path

A component path is specified as an optional partition list, followed by a local type name and then followed by an optional index. Here are some examples of component paths:

```
Grand<>Ball Room[5]
Ball Room[LAST]
Really<>Grand<>Ball Room
Ball Room
```

## Indexed object names

An object name can refer to a particular occurrence of a data object. The index of the occurrence appears in square brackets immediately after its name. For example, the object name of the third **Note** would be:

```
Note[3]
```

The index between the square brackets can be an integer or the reserved word, "LAST" (a special index value that refers to the last data object of a particular series). The index cannot be another object name. To use a variable index, use the CHOOSE function.

LAST is interpreted in the context in which it is used.

## Using LAST in a map rule to reference an input

In a map rule, when referencing an input, LAST refers to the very last occurrence of that input. For example, the last

**Note** of the input **Report** would have the name:

```
Note[LAST]:Report
```

In this context, all of the input has been validated before map rules are evaluated.

## Using LAST in a component rule

In a component rule, LAST refers to the last occurrence found. For example, in the rule:

```
PO# Field:Order = PO# Field:Order[LAST]
```

LAST refers to previous occurrence of **Order** because when this rule is evaluated, the last known **Order** is the last one found.

## Using LAST in a map rule to reference an output

In a map rule, when referencing an output, LAST refers to the last built occurrence of that object. For example, using the same expression:

```
PO# Field:Order = PO# Field:Order[LAST]
```

LAST refers to the previous occurrence of **Order** because, when this rule is evaluated, the last known **Order** is the last one built.

LAST is a reserved word and is not case sensitive.

## Component paths separated by a colon

A component in the component list of another object is always referenced by a colon (:). The local type name of the nested component precedes the colon. The name of the object that contains the component follows the colon. For example, the **Line Item** component of **PO** would have this name:

```
Line Item:PO
```

The **Last Name** component of **Client**, which is in turn a component of **Account**, would have this name:

```
Last Name:Client:Account
```

## Component paths separated by IN

To reference all occurrences of an object contained in another object, use the keyword IN.

For this example, **ShipSchedules** appears as a component of different components within **OrderAck**.

The object name for all occurrences of **ShipSchedules** within **ORD\_ADR\_OUT\_WD Record** is:

```
ShipSchedules IN ORD_ADR_OUT_WD Record:OrderAck
```

When IN is used, the contained object name must refer to the same type. In the example, the **ShipSchedules** component of **ORD\_ADR\_OUT\_WD Record** and the **ShipSchedules** component of **OrderLine Structure** must be of the same type.

In the map designer, you can refer to all occurrences of an object contained in a card object. The object name for all occurrences of **ShipSchedules** within **OrderAck** is:

```
ShipSchedules IN OrderAck
```

## Referring to all occurrences with IN COMPONENT

In the schema designer, you can refer to all occurrences of an object contained in a component list. To refer to all components in a component list, up to a given component, the word COMPONENT is used.

For example, the rule on the component **Conclusion Topic** needs to count all of the valid **Topics** up to, and including, **Conclusion Topic**.

The `IN COMPONENT` expression can be used:

```
#Topic Field:$ = COUNT (Topic IN COMPONENT)
```

The above component rule works only if **Topic** is partitioned.

## Comment object name

A comment name can be either an in-comment name or an @-comment name.

An in-comment name references all comment objects within a specified object. An in-comment name is specified as a component-path, a space, the reserved word IN, followed by another space, followed by the component path of the component on which the floating component type is defined. For example, given the data structure shown below, to reference all **INPUT Parameters** for a particular **Input**, regardless of whether they followed **Method OpenFile** or **Method GetEntry**, the object name would be **INPUT Parameters IN Input**.

An @-comment name references all comment objects that follow immediately after a specified object. An @-comment name is specified as the component path of the comment, followed by the reserved symbol "@", followed by the component path of the component that follows the comment. For example, to reference only the **INPUT Parameters** that follow immediately after the **Method OpenFile** (and before **Method GetEntry**), the object name would be **INPUT Parameters @ Method OpenFile:Input**.

## Shorthand notation

In a rule, a dollar sign (\$) can be used to refer to the object to the left of the rule cell. In the schema designer, the \$ sign refers to the object you are qualifying. In the map designer, the \$ refers to the object to which the rule evaluates.

In certain cases, you can also use a period between colons to shorten the reference to a data object name. The Use Ellipses command can be used to automatically shorten data object names you drag into rule expressions.

## Using the dollar sign (\$)

The dollar sign can be used to refer to the object to the left of a rule cell.

For example, the component rule below applies to the component **RunRule**.

The **RunRule** in the rule can be replaced by \$. You can remember it this way: a dollar sign always means “*whatever component appears to the left of the rule*”.

Here is an example of a map designer rule that uses the dollar sign:

```
Order:Invoice = MapMyOrder ( Order:PO , Index ( $ ) )
```

In this example, each time this rule is evaluated, the index of the **Order:Invoice** being built is the object referred to by the \$.

## Using ellipses

A single period can be used, like ellipses, in place of object names, as long as that object name can be interpreted as a unique object.

For example, if you have the following object name:

```
City Field:BillTo Customer:Account:File
```

you can replace it with

```
City Field::File
```

If file also contains another **City Field**, such as

```
City Field:ShipTo Customer:Account:File
```

and you use the **Use Ellipses** option, the system returns

```
City Field:BillTo Customer::File
```

## Evaluating expressions

Expressions are evaluated based on the context in which data object names, literals, operators, function names and map names appear in the expression, and the data on which it operates. A component rule always evaluates once for each occurrence of a component. A map rule can evaluate many times.

When a map rule evaluates, the map designer selects an evaluation set of values and then evaluates the rule. Then it selects another evaluation set of values, if there is one, and evaluates the rule again. This continues until all the different evaluation sets have been used.

The same number of evaluations is always produced, but output objects can be ordered differently, depending on how you define your map.

For a selected evaluation set, a particular instance of a rule is evaluated using parentheses, operator precedence rules, and left-to-right precedence rules to get one result.

## Card order can influence the order of evaluation sets

When an evaluation set is selected, card object names are ordered by their position in the map: card 1 before card 2, and so on. The card order for a map can affect the order of output results.

For example, suppose you want to evaluate the following map rule.

```
NumberSet ( s ) = B:Card2 + A:Card1
```

In this example, an evaluation set for this rule consists of one value for B and one value for A. One evaluation set produces one **NumberSet**.

As an example, here there are three As and three Bs in the data:

	A Values	B Values
1	3	
2	2	
3	1	

There are nine evaluation sets. The order of the cards might affect the order of the evaluation sets.

	Evaluation #	A	B	Result
1	1	3	4	
2	2	3	5	
3	3	3	6	
4	1	2	3	
5	2	2	4	
6	3	2	5	
7	1	1	2	
8	2	1	3	



	<b>Evaluation #</b>	<b>A</b>	<b>B</b>	<b>Result</b>
9		3	1	4

	<b>Evaluation #</b>	<b>B</b>	<b>A</b>	<b>Result</b>
1		3	1	4
2		2	1	3
3		1	1	2
4		3	2	5
5		2	2	4
6		1	2	3
7		3	3	6
8		2	3	5
9		1	3	4

If both A and B are contained in the same card object, evaluation sets are selected based on the order A and B appear in the rule. Leftmost objects are selected first. If you change the rule to this:

```
NumberSet ( s ) = B:Card2 + A:Card2
```

the same results are produced as when card 2 is the first card.

## Functions influence the number of evaluation sets

For this example, you have the following rule:

```
NumberSet ( s ) = A:Card1 + SUM ( B:Card1 )
```

Using the same values for A and B, there are only three evaluation sets as shown.

	<b>A Values</b>	<b>B Values</b>
<b>1</b>	<b>3</b>	
<b>2</b>	<b>2</b>	
<b>3</b>	<b>1</b>	

	<b>A Values</b>	<b>B Values</b>
<b>1</b>	<b>3</b>	

	<b>A Values</b>	<b>B Values</b>
<b>2</b>		<b>2</b>
<b>3</b>		<b>1</b>

	<b>A Values</b>	<b>B Values</b>
<b>1</b>		<b>3</b>
<b>2</b>		<b>2</b>
<b>3</b>		<b>1</b>

If your rule looks like:

```
NumberSet ( s ) = SUM ( A:Card1 ) + SUM ( B:Card1 )
```

there is one evaluation set: NumberSet[1] = 12

	<b>A Values</b>	<b>B Values</b>
<b>1</b>		<b>3</b>
<b>2</b>		<b>2</b>
<b>3</b>		<b>1</b>

If you have the following rule:

```
NumberSet ( s ) = A:Card1 + EXTRACT ( B:Card2 , B:Card2 != 2 )
```

the order of the operands does not affect the order of evaluation set selected. This is because an A is selected, then the EXTRACT is evaluated, then evaluation sets are determined for that A and a given EXTRACT result. When all EXTRACT results are used, the next A is selected, and so on.

Given the same data, this time you get the following results:

	<b>Evaluation #</b>	<b>A</b>	<b>B Extract Values</b>	<b>Result</b>
<b>1</b>	1	3		4
<b>2</b>	1	1		2
<b>3</b>	2	3		5
<b>4</b>	2	1		3
<b>5</b>	3	3		6
<b>6</b>	3	1		4

As another example, the following map rule applies to the output Status(s):

```
= IF ( Quantity:Order Record:Order > 10 &
OR ( Store:Order Record:Order = "A" ) ,
    "Accept" , "Reject" )
```

The first argument of the IF function is an expression. One evaluation set of this expression requires:

- the **Order**
- one **Order Record**
- one **Quantity** of the selected **Order Record**
- all the **Stores** of the selected **Order Record**

The IF function evaluates once for each **Quantity** of each **Order Record**. Other than the first argument of the IF function, nothing else in the map rule affects the number of times the map rule will be evaluated. So, if there are 1000 **Quantity** data objects contained in **Order**, this map rule will evaluate 1000 times.

## Object names influence the number of evaluation sets

In some expressions, the same data object name might appear more than once. The same data object name always refers to the same data object. The same data object name influences the combinations of evaluation sets that are selected. For example:

```
BackOrder(s) = Qty Ordered:Record:Order - Qty Received:Record:Order
```

When a **BackOrder** evaluation set is selected, the **Record** that contains **Qty Ordered** is always the same **Record** that contains **Qty Received**. And an **Order** is always the same one for one evaluation set. When the same object name is used more than once in the same expression, both references bind to the same object.

If there is always one **Qty Ordered** per **Record** and one **Qty Received** per **Record**, the following would be sample data:

	Record #	Qty Ordered	Qty Received
1		5	4
2		4	4
3		6	2

This would produce three **Back Orders**, one for each **Record**.

You can coordinate nested data objects easily by the way an object name is used.

## Using object names to coordinate evaluation sets

In the following example, common object names are used to coordinate **Qty** and **UnitPrice** so they do not get mixed up across different **Records**:

```
Extension = SUM ( Qty:Detail Record:Order * UnitPrice:Detail Record:Order )
```

This rule has only one evaluation set because the SUM consumes all the objects referenced in the rule.

## Using IN to decouple object name coordination

Suppose you have a situation in which you do not want to coordinate common objects. You might, for example, want to count different objects contained in the same object, as in:

```
Summary = COUNTabs ( Header:Order:Message ) + COUNTabs ( Line Item:Order:Message )
```

If there are multiple **Orders** in your data, the **Summary** would not be evaluated correctly. This rule would only count the **Headers** of the first **Order** and add it to the count of all Line Items of the first **Order**. To count all the **Headers** and all the **Line Items** in all the **Orders**, use IN:

```
Summary = COUNTabs ( Header IN Message ) + COUNTabs ( Line Item IN Message )
```

## Using IN to decouple different partitions in the same rule

If data objects in an expression are different partitions of the same type, the expression might evaluate to "none".

For example:

```
MyMap ( Good<>Design , Best<>Design )
```

**Good** and **Best** are partitions of the same type, **Design**. A given **Design** could never be both a **Good<>Design** and a **Best<>Design**; therefore, the second argument of **MyMap, Best<>Design**, evaluates to "none".

To reference both the **Good** and **Best** partitions, use IN:

```
MyMap ( Good IN Design , Best IN Design )
```

## Operators

Operators are used for arithmetic functions and text functions on items. If an operator requires two operands, the operator symbol appears between them. For example, in the expression **a + b**, the operator symbol "+" appears between the operands **a** and **b**.

Operators are reserved symbols that cannot be used in type names.

## Arithmetic operators

Operator	Description
+	addition
-	subtraction
*	multiplication

/  
division

## Text operators

<b>Operator</b>	<b>Description</b>
+	concatenation

## Logical operators

Logical operators are used in expressions that evaluate to "true" or "false".

<b>Operator</b>	<b>Description</b>
^	exclusive or
&	and
	or
!	not
=	equals

## Comparison operators

<b>Operator</b>	<b>Description</b>
=	equal to
>	greater than

&lt;

less than

&gt;=

greater than or equal to

&lt;=

less than or equal to

!=

not equal to

**NOT=**

alternative for not equal to

The results of comparison operators are based on the native collating sequence of the machine used to run the map.

## Order of operator evaluation

Use parentheses to group operations. Expressions within parentheses are evaluated before performing other operations. For example:

```
(UnitPrice - Discount) * Tax
```

The expression **UnitPrice - Discount** is evaluated before the rest of the expression.

Some operators are evaluated before others according to standard rules of precedence. Operators are evaluated in the following order:

Precedence	Operators	Description
first	()	Operations within parentheses
second	+ and -	unary plus and minus
		The sign indicates a positive or negative expression, for example: - quantity*rate
third	=, <, >, <=, >=, !=	comparative operators
fourth	^	logical operator "exclusive or"
fifth	&	logical operator "and"
sixth		logical operator "or"
seventh	* and /	multiplication and division
eighth	+ and -	addition and subtraction

Operators of equal precedence are evaluated left to right.

## Operands

Operands are the arguments for an operator. For example, in the expression **a + b**, the **+** is the operator symbol, and **a** and **b** are the operands.

An operand can be any of the following types:

- a literal
- a data object name
- a function
- another operator

## Using functions in expressions

Functions perform a particular action on its input arguments. A function is written like this:

```
FUNCTION (argument1, argument2, ... argumentn)
```

The arguments of the function appear inside the parentheses, separated by commas. A function might require a fixed number of arguments or might allow a varying number of arguments. You can use functions anywhere in an expression.

## Function arguments

Input arguments themselves can be expressions. Some functions limit the type of expression that can be used as an input argument. The syntax specification tells you the number of data objects that can be used for one function evaluation.

There is *always* one output argument for a function. The specification of the output argument tells you: 1) the type of the result produced by a function and 2) the number of objects that can be produced when the function evaluates once.

You need to know the type of the output argument because functions can be used as other arguments. The result of a function can also be used to directly produce an output data object contained in an output destination.

When the map designer analyzes the interfaces of expressions in rules during the build process, it ensures that the output argument of a function matches the input argument it is used for. The analysis also checks the type of the output argument when that function is used to directly produce an output data object.

The output argument of a function specifies the result that the function produces for one evaluation. For example, one evaluation of the ABS function produces one positive number. One function evaluation might produce multiple outputs. If it does, the expression that contains that function might produce multiple evaluation sets.

## Input arguments

The input arguments for a function specify the information the function uses to return a result. Some functions require an exact number of input arguments. For other functions, the number of input arguments can vary. Optional arguments for a function are shown in brackets [ ].

To correctly use a function, you must use the correct arguments in the correct order.

Each argument can, itself, be an expression. In general, an input argument might be any of the following:

- an object name
- a literal
- a function-name and its arguments
- an operator and its arguments
- an enumerated series of literals, such as {a, b, c}
- a map name and its arguments

For example, some valid expressions that use the ABS function are:

```
ABS ( Quantity:LineItem )
ABS ( UNIQUE ( Quantity:LineItem ) )
ABS ( Quantity:LineItem - 300 )
```

Each function has its own expression syntax for its input arguments. For example, the ABS function cannot use a map as an argument. See [Syntax of a Function on page 754](#) for notation used to specify valid expression syntax.

## Nested input arguments

When a function, operator, or map is used as an input argument, each of these has arguments of its own. Arguments of other arguments are said to be *nested*. For example, in the use of the ABS function, you can use a function as an argument:

```
ABS ( UNIQUE ( Quantity:LineItem ) )
```

The output of the UNIQUE function becomes the input argument to the ABS function.

## Output arguments

The output argument of a function indicates the type of object the function produces and the number of objects a function can produce. An output argument is specified as one of the following object types or values:

- A single data object of a type
- A series of data objects of the same type
- "True" or "False"

If the output of a function is a series, that function can be used only in a map rule in the map designer. It cannot be used in a component rule in the schema designer. For example, you cannot use the CLONE, EXTRACT, REJECT, SORTDOWN, SORTUP or UNIQUE functions in the schema designer.



## Function arguments and evaluation

When a function is evaluated, the number of argument objects used for one evaluation depends on the function. In the specification of the syntax for a function, the number of input objects that can be used for each argument for one function evaluation is expressed as "single" or "series".

Some functions use a single object as the value of an argument for one evaluation. For example, the ABS function uses a single object as the value of its argument for one evaluation.

```
ABS ( Quantity:LineItem )
```

When a single object is used for one evaluation, the function itself can evaluate many times if there is more than one data object that fits the argument definition. For example, if there were ten **Quantity:LineItem** data objects, the ABS function could evaluate ten times.

Some functions use an entire series of objects as the value of an argument for one evaluation. For example, the COUNT function uses an entire series of data objects as the value of its argument for one evaluation.

```
COUNT ( LineItem:PO )
```

When a series of data objects is used for one evaluation, the function evaluates just once when there is more than one data object that fits the argument definition. For example, if there are ten **LineItem:PO** data objects, the COUNT function would evaluate only once.

Some functions use an entire series as input and produce a series as output. For example, the evaluation set for the EXTRACT function produces a series from an input series. When a function produces a series, each output can be selected for different evaluation sets of the expression that contains that function.

Consider the following expression:

```
Line Item(s) = EXTRACT ( Line Item:Order, Qty:Line Item:Order > 1000 )
```

Each **Line Item** that fits the specified criteria produces a **Line Item of Order**. In this example, the map rule is evaluated many times, once for each **Line Item** produced by the EXTRACT function.

When functions are part of other expressions, the number of evaluation sets for that function depends on the object names used in the entire expression. For example:

```
Debit ( s ) = ABS ( Debit:Account:Input )
```

There might be many evaluation sets for the above expression. The ABS function might be evaluated many times, once for each **Debit** of each **Account** of **Input**. However, if the ABS function is part of a more complex expression, common objects of the expression might determine the number of ABS evaluations.

## Expressions that evaluate to NONE

The following discussion explains how the map designer evaluates an expression when expressions within it evaluate to "none".

## When an operand evaluates to NONE

The following table explains how an operator expression evaluates when an operand evaluates to `none`.

The symbol **A** represents one operand of the expression and **B** represents the other operand. The symbol **!=** means "not equal to".

Operator Expression	Condition	Evaluates to:
A / B	A = NONE B = NONE	NONE
A*B	A = NONE B = NONE	NONE
A / B	A != NONE B = NONE	0
A*B	A != NONE B = NONE	0
A + B	A = NONE B = NONE	NONE
A - B	A = NONE B = NONE	NONE
A + B	A = NONE B != NONE	B
A - B	A = NONE B != NONE	-B
A + B	A != NONE B = NONE	A
A - B	A != NONE B = NONE	A
A > B	A = NONE B = NONE	FALSE
A < B	A = NONE B = NONE	FALSE
A = B	A = NONE B = NONE	TRUE
A > B	A = NONE B != NONE	FALSE
A < B	A = NONE B != NONE	TRUE
A = B	A = NONE B != NONE	FALSE
A > B	A != NONE B = NONE	TRUE
A < B	A != NONE B = NONE	FALSE
A = B	A != NONE B = NONE	FALSE

For example, in this operator expression:

```
Total = #Guests + 1
```

if **#Guests** evaluates to "none", the value for **Total** will be 1.

## When an input argument of a function evaluates to NONE

If any input argument of a function evaluates to "none", the function might not evaluate to "none". For example, `ABS (NONE)=NONE`, but `COUNT (NONE)=0`.

## When an input argument of a functional map evaluates to NONE

If any argument of a functional map evaluates to "none", the functional map will not be evaluated for that evaluation set.

For example:

```
MakeForm ( EntryForm:Input , GroupInfo:Input )
```

Where the object **GroupInfo** is optional; it has a component range of (0:3). If there are no occurrences of **GroupInfo** in the **Input** data object, the map **MakeForm** will not be evaluated at all.

Here is another example:

```
OrderMap ( OrderRecord:Order:Input , SummaryRecord:Order:Input )
```

If **SummaryRecord** has a component range of (0:1) and it is missing in a particular **Order**. The map **OrderMap** will not be evaluated for that **Order**.

If you want **OrderMap** to be evaluated even when **SummaryRecord** is missing, use the entire **Order** as an input argument, rather than **SummaryRecord**. This will ensure that **OrderMap** will be evaluated, even if **SummaryRecord** is missing. Then, in the map **OrderMap**, you can still map from **SummaryRecord**.

The rule you should use is this:

```
OrderMap ( OrderRecord:Order:Input , Order:Input )
```

## When an input of an executable map evaluates to NONE

The evaluation of a map referenced in an expression differs from the evaluation of that map when it is executed as a functional map and the evaluation of a function because of how the NONE is treated.

The evaluation of a map used as a function is different from the evaluation of that map when it is run as an executable map. When a map is run as an executable, if any input card data object has no content, the map will still run. This is not true when that map is referenced as a function. If any input argument of a functional map evaluates to "none", the map will not evaluate.

## Impact of track setting on order of output

This property setting determines whether objects are tracked according to content or according to their position within a series. This setting will determine the sequence in which those objects are evaluated when referenced.

For example, assume that **RunsByInning** is a component of **BallGame**. **BallGame** is defined as an explicit format, delimited (by a colon) group. The data for **BallGame** is as follows:

```
2::1:1::1::3
```

- **Track Content.** If **BallGame** has a **Track** setting of **Content**, all instances of **RunsByInning** with content will be mapped before any empty instances. So, if **BallGame** was mapped to itself, the output would be:

```
2:1:1:1:3
```

- **Track Places.** If **BallGame** has a **Track** setting of Places, instances of **RunsByInning** will be mapped in the sequence in which they occur, including those instances that are empty. So, if **BallGame** was mapped to itself, the output would match the input.

```
2::1:1::1::3
```

**Track Places** will only have an impact on the sequence of the output when *both* the input group object and output group object are defined as **Track Places**.

However, **Track Places** will have an impact in situations in which an input defined with **Track Places** is referenced by index. For instance, in the above example,

- If **RunsByInning** was defined as Track Places, **RunsByInning[2]:BallGame** would have a value of **2** because the value in the second "place" in **BallGame** is a **2**.
- On the other hand, if **RunsByInning** was defined as **Track Content**, **RunsByInning[2]:BallGame** would have a value of **1** because the second instance of **BallGame** with content has a value of **1**.

## Evaluated expressions assigned to output items

### NONE assigned to an output number

When an output item is defined as a number, its value is determined from the result of the evaluated expression that is then converted to its output form as follows:

Output is specified as:	Evaluated value	Output value
Required	0	0
Required	NONE	0
Optional	0	0
Optional	NONE	NONE

### Automatic item format conversions

When an item is assigned an output value, the map designer will automatically convert an item with the same interpretation (either number, text, date, or time) from one presentation to another. For example, if you want to map an item that is a number and whose presentation is integer to an item that is also a number but whose presentation is decimal, the map designer will automatically convert it.

The map designer fills in missing information in date and time conversions in the following way:

- **Dates with a year, but no century.** The century will be determined using the **CenturyDerivation** map setting.
- **Missing part of an output date format.** If a portion of the format of an output date cannot be derived from the input date, the missing part will have pound characters (#) in its place. For example, if *09/98* is the date for an

input with a format string of MM/YY and that date was mapped to an output date with a format of MM/DD/YY, the output date would be built as **09/##/98**.

- **Missing part of an output time format.** If a portion of the format of an output time cannot be derived from the input time, the missing part will have zeroes (0) in its place. For example, if `12:14` is the data for an input with a format string of HH12:MM and that time was mapped to an output time with a format of HH:MM:SS, the output date would be built as **12:14:00**.

To convert a number to text, a date or time, or vice versa, you must use a function to perform the conversion. For example, to convert a date to text, use the function DATETOTEXT.

## Numeric precision

All numbers in the input and numeric literals in map rules are converted to multiple precision and converted to the format required at output time.

- If the multiple precision number does not overflow but does not fit into the maximum size of a numeric item, that item is filled with the number sign character (#) up to its maximum size.
- An arithmetic overflow is recognized when an arithmetic expression includes:
  - Divide by zero
  - A multiple precision number greater than 1.0E254

Arithmetic overflow is designated in the output object with up to a maximum of ten number sign (#) characters.

## Generating line numbers in test data

You can generate index line numbers for the CSV or excel format test data.

- If you want line numbers in the text output, apply `=TEXT(COUNTER)`
- If you want line numbers in the numeral output, apply, `= COUNTER`

## Using functions

A function is an expression that generates an output by performing a certain operation on one or more inputs. Most functions can be used in both component rules and map rules. Those that produce a series can only be used in map rules.

### Functions in a component rule

For example, the object **TotalQty** in your data is the sum of all the **Qty** objects in your data; in a component rule for **TotalQty**, you can use the SUM function to verify this relationship.

### Functions in a map rule

Suppose you want to map your data differently according to the presence of a certain input data object. In your map rule, to check the presence of the data object, use the function PRESENT. To create the output according to whether

the input data was present, use the function IF. Examples of the IF function can be found in ["Logical Functions" on page 862](#). Examples of the PRESENT function can be found in ["Inspection Functions" on page 847](#).

## Syntax of a function

The following is an example of a function:

```
FUNCTION ( argument1, argument2, ... argumentn )
```

When you use a function, substitute the name of a specific function for FUNCTION and substitute a specific expression for each input argument. Each input argument is separated by a comma and the set of arguments is surrounded by parentheses.

## Function argument syntax

Arguments typically have some restrictions on the expression used for that argument. For example, the input argument to the INDEX function can only be an object name. Depending on the function, one or many data objects can be used for an input argument for one function evaluation.

For each function listed in this documentation, the input argument has a syntax definition that indicates:

- the number of objects that can be used for each argument for one function evaluation, expressed as either single or series

followed by

- the type of expression that can be used for each argument.

For the output argument, the syntax specification is similar to the input argument syntax, except that the output is always used as a specific type of object, not an expression.

For example, single-item-expression means one evaluation operates on a single item that can be defined as an item expression. Series-item-expression means one evaluation operates on one or more input argument items that can be defined as an item expression.

In this documentation, the terms listed in the following table are used to describe the syntax of function arguments.

### Term

#### Explanation

#### **general-expression**

Any valid combination of object names, literals, operators, function names, and map names.

#### **condition-expression**

Any valid combinations of object names, literals, conditional operators, and functions whose output arguments are specified as `true` or `false`.

**item-expression**

Any valid combination of object names, literals, operators, and functions whose output argument is specified as an item.

If an item must be interpreted as a date, time, or number, the item is referenced by its interpretation. For example, if an item-expression must be text, it is specified as text-expression.

**object-expression**

An object name or a series producing function.

If an object expression is further limited to be an item, it is specified as item-object-expression.

**object-name**

The name of a data object, as defined in [Data Object Names on page 735](#). If an object name is further limited, it is specified as a prefix. For example, simple-object-name refers to a simple type name.

...

Indicates that the function can take on any number of additional similar arguments.

[]

Indicates that the argument is optional.

{ literal , literal ... }

An enumerated series of literals.

The syntax specification shows each input argument, using terms as described in the previous table, preceded by the word single or series. For example:

**Syntax:**

ALL ( series-condition-expression )

The arguments are defined using a name that describes its use by the function. A simple example is:

**Meaning:**

DATETONUMBER ( date\_to\_convert )

Function names are shown in uppercase letters in this documentation. However, function names are not case sensitive.

There is also a description of what each function returns.

**Returns:**

A single integer

## Functions that convert character-set encoding to the native code page

Functions that use the EXIT architecture convert data to the native character set of the platform as part of the function call. These functions use the EXIT architecture:

- DBLOOKUP
- DBQUERY
- DDEQUERY
- EXIT
- JEXIT
- GET
- PUT
- RUN
- XQUERY
- XSLT
- MATHLIB functions
- XMLLIB functions (except XPATH and XPATHEX)
- RESOURCELIB functions
- Customer-developed functions that use the EXIT architecture

Maps that use these functions must take the data conversion into account. For example, a map that uses the EXIT architecture converts non-EBCDIC data to EBCDIC when the map runs on a z/OS® platform. To prevent the data conversion, the map must use a function like:

```
CTEXT(object, "Native")
```

If the function returns the data from the z/OS® platform to another platform, the map must specify that the data to return is non-EBCDIC by using a function like:

```
CTEXT(object, "data language")
```

## General functions

### ASFUNCTION

When you run a functional map with the ASFUNCTION function, the output of the functional map returns to the map rule that invoked the functional map. The invoking rule can work with the output from the functional map. The type of the output card must be an item.

Without ASFUNCTION, a functional map writes to the output card before the rest of the invoking rule is evaluated.

If ASFUNCTION runs a functional map that calls a second functional map, the output of the second map also returns to the map rule, as if the second map had been invoked by ASFUNCTION. If the output of each map is a number and the outputs are concatenated, ASFUNCTION adds the numbers instead of concatenating them.

**Syntax:**

```
ASFUNCTION ( general-expression )
```

**Meaning:**

```
ASFUNCTION ( FunctionalMapName ( argument-1 , argument-2 , ... argument-n ) )
```



**Returns:***item***Example 1****Rule:**

```
Output_name = FMAP1 (rule) + " and " + Fmap2 (rule) + "."
```

**Result:**

```
Output_of_FMap1Output_of_FMap2 and .
```

**Rule:**

```
Output_name = ASFUNCTION(FMAP1 (rule) + " and " + Fmap2 (rule) + ".")
```

**Result:**

```
Output_of_FMap1 and Output_of_FMap2.
```

**Example 2****Rule:**

```
Output_name = IF (FMap1 (rule) = "abc", "equal", "not equal")
```

**Result:**

- If FMap1 produces **abc**:

```
abcnot equal
```

- If FMap1 produces **def**:

```
defnot equal
```

**Rule:**

```
Output_name = ASFUNCTION(IF (FMap1 (rule) = "abc", "equal", "not equal"))
```

**Result:**

- If FMap1 produces **abc**:

```
equal
```

- If FMap1 produces **def**:

```
not equal
```

**Example 3****Rule:**

```
Output_name = ASFUNCTION(FMAP1(rule1) + FMAP2(rule2))
```

Result:

- If FMAP1 produces 1 and FMAP2 produces 2:

3

- If FMAP1 produces A and FMAP2 produces B:

AB

## CLONE

The CLONE function creates a specified number of copies of some object.

This function can be useful when the number of output objects to be built depends on a data value, rather than the number of objects that exist in the data.

### Syntax:

CLONE (single-object-name , single-integer-expression )

### Meaning:

CLONE (object\_to\_copy , number\_of\_copies)

### Returns:

A series-object

The CLONE function returns a series of the object specified by *object\_to\_copy*. The output series consists of as many copies of the object as specified by *number\_of\_copies*. The value of each member of the resulting output series is the same as *object\_to\_copy*.

## Examples

- LineItem Segment (s) = LotsOf ( CLONE ( Detail Row , Quantity:Detail Row ) )

In this example, the input contains an object named **Detail Row** that has a **Quantity** item component. You want to create multiple **Line Item Segments** for every **Detail Row** in your input based on the value of **Quantity**. So, if **Quantity** has the value 5, you want to build five **Line Item Segments** for that **Detail Row**.

## DEFAULT

The DEFAULT function allows a predefined value to be introduced into an expression.

DEFAULT, which is typically used with the EITHER function, returns the value assigned as the default value when the type was defined.

### Syntax:

DEFAULT(single-type-expression)

### Meaning:

DEFAULT(type\_whose\_defined\_default\_value\_is\_desired)

**Returns:**

A single-object

**Examples**

- `DEFAULT(ZipCode)`

In this example, the defined default value for `ZipCode` is returned. When used with `EITHER`, `DEFAULT` might appear like this:

```
EITHER( ZipCode,DEFAULT( ZipCode))
```

If the first argument evaluates to a value other than "none", that value is used. If the first argument evaluates to "none", the default value for **ZipCode** is used.

**ECHOIN**

The `ECHOIN` function returns a command (or property) to be used in a `RUN` function argument.

**Syntax:**

```
ECHOIN (single-integer-expression, single-text-expression)
```

**Meaning:**

```
ECHOIN (card#, item_or_group_interpreted_as_text)
```

**Returns:**

A single object

**Examples**

Before `ECHOIN` was introduced, the `RUN` function was used in this way:

```
RUN ("Mymap", "-IE1S"+NUMBERTOTEXT (SIZE(Data))+ " " + TEXT(Data))
```

You can use `ECHOIN` with `RUN`, similar to the following example:

```
RUN ("Mymap", ECHOIN(1,(Data)))
```

**Related functions**

`HANDLEIN`

**HANDLEIN**

The `HANDLEIN` function returns a command (or property) to be used in a `RUN` function argument.

You can use the HANDLEIN function when deriving large amounts of data from a parent map for use in a RUN map. The HANDLEIN function works by using the same instance of the parent map's data set in the RUN map, instead of duplicating it as the ECHOIN function does.

The result of the HANDLEIN function is a text object that defines the characteristics of the data to be used. This includes an internal handle, offset, and length of the data. The result of this can be seen in an execution audit log:

```
<ExecutionSummary MapStatus="Valid" mapreturn="0" ElapsedSec="0.1803"
BurstRestartCount="0">
  <Message>Map completed successfully</Message>
  <CommandLine>install_dir\sdq20.mmc -IH1 4:512:1024 -ae</CommandLine>
  <ObjectsFound>85</ObjectsFound>
  <ObjectsBuilt>23</ObjectsBuilt>
  <SourceReport card="1" adapter="Handle" bytes="1024" adapterreturn="0">
    <Message>Success</Message>
    <Settings>4:512:1024</Settings>
    <TimeStamp>05:06:07 January 27, 2004</TimeStamp>
  </SourceReport>
```

You cannot use a GETANDSET function against an input that is passed using HANDLEIN.

**Syntax:**

HANDLEIN (single-integer-expression, single-text-expression)

**Meaning:**

HANDLEIN (card\_to\_override, data\_object)

**Returns:**

A character text item

When the HANDLEIN function returns a character text item, the string is prefixed with: **IHx**, where x is the first parameter.

String format:

-IHx <internal-handle>.<offset>.<length>

Example:

-IH1 4.4096.512

A *single-integer-expression* is the number item that specifies the card number to override. A *single-text-expression* specifies the data object to use for the override.

In a RUN map rule, you can override a source card with a handle of a current map's card by specifying the card to override and the object to override it with.

**Examples**

In the following rule, the data is object c:b:a for input card 2 of the RUN map:

```
HANDLEIN(2, c:b:a)
```

The second parameter can be more complex if needed. For example,

```
HANDLEIN(2, SUBSTITUE(c:b:a, "a", "A"))
```

Any data in object **c:b:a** that contains a lowercase "a" would change it to an uppercase "A" for the call to the RUN map. In cases such as this, where the data cannot be expressed contiguously within the original card or cards, a scratch file is used to hold the expression and the calling map's handle is that of the scratch file.

There is no correlation between the handle returned in the output of HANDLEIN and the actual card number of the second parameter.

### Related functions

Generally, the ECHOIN function is used to return a command or property to be used in a RUN function argument. However, for large amounts of data, using the HANDLEIN function can be more efficient.

For best results, use the ECHOIN function for small quantities of data, such as less than 100K, and the HANDLEIN function for larger quantities of data of more than 100K.

## PARSE

The PARSE function analyzes a data object and validates it against the metadata of a component type. The resulting data structure is available to the calling map for further processing.

The input to the PARSE function is any data that can be processed as a text blob, for example, the result of a functional map invocation, a GET function, or a RUN map.

### Syntax

```
PARSE (general_expression)
```

### Meaning

```
PARSE (data_to_parse)
```

### Returns

A mapped data object in a work file

## Examples

The data structure that results from the PARSE function is available to the calling map. When you use a RUN map to parse data, the resulting data structure is available to the RUN map rather than to the calling map. By using the PARSE function instead of the RUN map implementation, you have fewer maps to configure and maintain.

The following examples:

1. Receive an input document from the SPE adapter
2. Extract the data objects from the document envelope
3. Return a response

### Example 1: RUN map implementation

In this example, the DEENVELOPE map calls the RUN map named DUMPRESPONSE to extract the data from the document.

The DUMPRESPONSE map has two input cards:

- SpeResponse (Response global XSD)
- Test Root (text root)

The output card of the DUMPRESPONSE map has the following rule:

```
=WriteDocument(Document:sequence:Documents:sequence:SpeResponse,
TestRoot;
TEXT(INDEX(Document:sequence:Documents:sequence:SpeResponse))) + "<NL>"
```

The DEENVELOPE map has two input cards:

- DataIn (text root)
- Test Root (text root)

The DEENVELOPE map has two output cards:

- DOCOUTPUT:

```
= RUN ("DumpResponse", ECHOIN( 1 , Deenvelope ) + ECHOIN( 2 , TestRoot ) + " -OE1" )
```

- DEENVELOPE:

```
=VALID(GET("SPE", "-DURL jdbc:derby://localhost:1527/spe2
-DBUSER derbyuser -DBPSWD derbypw
-DBDRIVER org.apache.derby.jdbc.ClientDriver -DENV", DataIn),
FAIL ("SPE Deenvelope failed: " + LASTERRORMSG() ))

/*
=VALID(GET("SPE", "-DURL jdbc:db2://bobdb9.bcr.ibm.com:50000/spetest
-DBDRIVER com.ibm.db2.jcc.DB2Driver -DBUSER spetest -DBPSWD Seer5r12
-DENV -T ", DataIn),
FAIL ("SPE Deenvelope failed: " + LASTERRORMSG() ))
*/

/*
=VALID(GET("SPE", "-DURL jdbc:db2://localhost:60000/SPETEST
-DBDRIVER com.ibm.db2.jcc.DB2Driver -DBUSER fred -DBPSWD org13asd
-DENV -T ", DataIn),
FAIL ("SPE Deenvelope failed: " + LASTERRORMSG() ))
*/
```

## Example 2: PARSE function implementation

In this example, a single map uses the PARSE function to perform the processing of the two maps in Example 1.

The DEENVELOPE\_PARSE map has two input cards:

- DataIn (text root)
- Test Root (text root)

The DEENVELOPE\_PARSE map has two output cards:

- RESPONSEINFO:

```
= WriteDocument( Document:sequence:Documents:sequence:SpeResponse,
TestRoot,
TEXT( INDEX( Document:sequence:Documents:sequence:SpeResponse ) ) ) + "<NL>"

= "Response - Status: " + Status:sequence:SpeResponse + ",
Advanced status: " + AdvancedStatus:sequence:SpeResponse + "<NL>"
```

- SPERESPONSE (Response Global XSD):

```
=PARSE( VALID(GET("SPE", "-DURL jdbc:derby://localhost:1527/spe2
-DBUSER derbyuser -DBPSWD derbypw
-DBDRIVER org.apache.derby.jdbc.ClientDriver -DENV", DataIn),
FAIL ("SPE Deenvelope failed: " + LASTERRORMSG() ) ) )
```

## REFORMAT

The REFORMAT function returns a type object that results from replacing the syntax of the input type with the syntax of the output type.

The initiator and terminator of the output type are built. The content result rules are determined based on the Input type and the Output type. Groups and Items can be used with REFORMAT.

REFORMAT cannot be used as an argument to a function, operator, or functional map.

### Syntax:

```
REFORMAT (single-object-expression)
```

### Meaning:

```
REFORMAT (type-to-convert)
```

### Returns:

A type object

REFORMAT returns a type object whose content matches the input type object, but whose syntax matches the output type object. The following table details the expected results based on the input and output.

Input Type	Output Type	Result
Group	Group	<p>The content for each output component results from the content of the corresponding input group component content. For example, output component 1 is matched with input component 1, output component 2 is matched with input component 2, and so on. The components of the groups must be items, not another group. If there is no matching input component for an output component, the output component's required occurrences are built as "none"s. If there is no matching output component for an input component, the data of that input component is ignored.</p> <p>When a component occurrence is built, the delimiter (if any) of the contained output group is built and an object of the component's output type is built from the corresponding input type using the REFORMAT algorithm.</p>
Group	Item	The text of a corresponding input group content.
Item	Group	The input item content is applied as though it were the first component of the output group.
Item	Item	The input item content is applied to the output item content.

## Examples

- MyXMLPurchaseOrder = REFORMAT(MyCOBOL\_PurchaseOrder)

This function is useful when the same type structure might come from different type trees, such as different versions of the same EDI, or when converting traditional formats to XML.

## RESOLVETYPE

The RESOLVETYPE function resolves the provided component path to a specific type tree reference when the map is being compiled.

This function returns the type of the component for the component path that you provided as an argument. It is used only with the IN keyword, to resolve the provided component path to a specific type tree reference. Use it when more than one type that has the same name occurs within (IN) the containing component or card. Specify this function only in map rules, not in component rules.

### Syntax:

RESOLVETYPE (single-text-expression)

### Meaning:

RESOLVETYPE (component\_path\_to\_resolve)

### Returns:

A single type reference



For more information about using the IN keyword to reference all occurrences of an object contained in another object, see [Component paths separated by IN on page 738](#).

## Example

```
NetWorth = SUM (RESOLVETYPE (Price:Cars:East:Regions) IN Inventory)
```

In the example, you have an inventory of cars and trucks in dealerships in different geographic regions: **East**, **West**, **North**, and **South**. The price of the cars and trucks are represented by two different types in the type tree with the same name, **Price**. You want to determine the net worth of the inventory of just cars from all of the dealerships in all of the regions.

The RESOLVETYPE function is used to return the type of the component path. In this example, the function is set up to return the type, **Price**, for **Cars**, for the component path, `Price:Cars:East:Regions`, specified in the argument of the function. This function also returns the same type, **Price**, for **Cars**, if the argument of the function specifies any of the **Regions** in the following component paths: `Price:Cars:West:Regions`, `Price:Cars:North:Regions`, and `Price:Cars:South:Regions`. Then **Price**, for **Cars**, is used as the criteria for the selection process in your total **Inventory**. The mapping process calculates the net worth by summing the values of **Price** for all **Cars** in all of the **Regions** in the total **Inventory**.

If you do not specify the RESOLVETYPE function, and instead, specify a rule such as `NetWorth = SUM (Price IN Inventory)`, the mapping process uses either one of the types, **Price** for **Cars**, or **Price** for **Trucks**, as the criteria for the selection process in your total **Inventory**. If you do specify the RESOLVETYPE function, the mapping process uses the returned type of the component path, specified in the argument of the function, as the criteria for the selection process in your total **Inventory**.

If you want to determine the net worth of just **Trucks** in the **Inventory** from all of the dealerships in all of the **Regions**, you can specify the RESOLVETYPE function in the following way: `NetWorth = SUM (RESOLVETYPE (Price:Trucks:North:Regions) IN Inventory)`. Again, the function also returns the same type, **Price**, for **Trucks**, if the argument of the function specifies any of the **Regions** in the following component paths: `Price:Trucks:West:Regions`, `Price:Trucks:East:Regions`, and `Price:Trucks:South:Regions`.

## Bidirectional functions

Bidirectional languages such as Arabic and Hebrew are languages in which the text is presented to the user ordered from right to left, but numbers and Latin alphabetic strings within the text are presented left to right. In addition, the order in which characters appear within program variables can vary. The text is usually stored in logical order, the order in which the characters are entered in the input field. These differences in ordering and in other associated presentation characteristics require the program to have the ability to convert bidirectional text strings from one format to another.

Use the SETLOGICALORDER, SETVISUALORDER, SETORIENTATIONLTR, SETORIENTATIONRTL, SETTEXTSHAPING, and SETTEXTSHAPINGOFF functions for specific transformation needs when dealing with bidirectional data.

## About the order of text

The order of text generally corresponds with way it is stored in memory or on disk. In general, the order of text is displayed, on the window for example, in visual order so that a person can read it the way it is subjectively meant to be read. That is, left-to-right characters are displayed from left-to-right, and right-to-left characters are displayed with the first character of the text block beginning from the right.

Storage in memory or on disk generally occurs in a logical manner so that the left-to-right characters are stored left-to-right (that is, one after another), and require no processing to display properly. However, when right-to-left characters are stored in a logical manner, they are also stored one after another from left-to-right. These text blocks, when in Logical order, are usually enclosed by markers that identify the start and end positions of the right-to-left text blocks.

## Logical to visual reordering

To display these logical sequences on a window would require that any left-to-right text is simply read and displayed. When right-to-left text is encountered, it must be turned around to the appropriate direction. This is logical-to-visual reordering.

## Examples

In the following example, lowercase characters are considered left-to-right (LTR) and uppercase characters are considered right-to-left (RTL).

<b>Text=</b>	<b>a</b>	<b>b</b>	<b>c</b>	<b>F</b>	<b>E</b>	<b>D</b>	<b>g</b>	<b>h</b>	<b>i</b>
Visual order	0x61	0x62	0x63	0x46	0x45	0x44	0x67	0x68	0x69
Logical order	0x61	0x62	0x63	0x44	0x45	0x46	0x67	0x68	0x69

## SETLOGICALORDER

Use this function to convert a bidirectional data object from visually ordered (characters ordered as they are presented for reading) to logically ordered (the order in which text is typed on a keyboard or phonetic order).

This function is only for use with text objects that have bidirectional properties. The function will have no effect on objects that have no bidirectional properties.

### Syntax:

```
SETLOGICALORDER (single-text-expression , single-text-expression)
```

### Meaning:

```
SETLOGICALORDER (visually_ordered_text_object , TRUE_or_FALSE_text_object)
```

### Returns:

A logically ordered text object.

The SETLOGICALORDER function changes the order of the data as held in memory (order scheme) of a text object that is being passed as a parameter. The function returns a text object that reflects the order scheme change.

The output of the function will be the data from the original bidirectional item, transformed to match the specified layout (as it would be if it were assigned to a bidirectional item with the specified characteristics). This allows you to use text manipulation functions, such as LEFT or MID, on bidirectional data using the proper understanding of the bidirectional data. For example, LEFT of the logical ordering of a bidirectional item would actually take the rightmost characters of the visual ordering of that item.

### Example

The properties of the text objects define the orientation and memory order of data objects found in the data stream.

Text objects defined as right-to-left would be considered as starting from the right character and progressing to the left, until the end of the data. However, the data can be held in memory with the start character (the right most character) being held in the left most position. This memory ordering is defined by the ordering scheme of the object.

## SETORIENTATIONLTR

The SETORIENTATIONLTR function converts the orientation of a text object from right-to-left (RTL) to left-to-right (LTR).

#### Syntax:

```
SETORIENTATIONLTR (single-text-expression, Boolean-expression)
```

#### Meaning:

```
SETORIENTATIONLTR (right_to_left_oriented_text_object, true_or_false_text_object)
```

#### Returns:

A text object with left-to-right orientation.

The second parameter, a Boolean expression, indicates whether symmetric swapping should be applied.

### Example

#### Example

```
SETORIENTATIONRTL( VLTRData:In1, "FALSE" ) )
```

## SETORIENTATIONRTL

Use the SETORIENTATIONRTL function to convert the orientation of a text object from left-to-right (LTR) to right-to-left (RTL).

#### Syntax:

```
SETORIENTATIONRTL (left_to_right_oriented_text_object, true_false_text_object,  
TRUE_or_FALSE_text_object)
```

**Meaning:**

SETORIENTATIONRTL (XML fragment)

**Returns:**

A text object with right-to-left orientation.

**Example**

**Example**

```
SETORIENTATIONRTL (VLTRData:In1, "FALSE")
```

## SETTEXTSHAPING

The SETTEXTSHAPING function converts a bidirectional text object from non-shaped to shaped.

**Syntax:**

SETTEXTSHAPING (single-text-expression)

**Meaning:**

SETTEXTSHAPING (unshaped\_text\_object)

**Returns:**

A shaped text object.

## SETTEXTSHAPINGOFF

The SETTEXTSHAPINGOFF function removes the shaping from a bidirectional text object.

**Syntax:**

SETTEXTSHAPINGOFF (single-text-expression)

**Meaning:**

SETTEXTSHAPINGOFF (shaped\_text\_object)

**Returns:**

A text object with no shaping.

## SETVISUALORDER

The SETVISUALORDER function converts a bidirectional data object from logically ordered to visually ordered.

This function can only be used with bidirectional data.

**Syntax:**

SETORDERVISUAL (single-text-expression , single\_text\_expression)

**Meaning:**

SETORDERVISUAL (text\_object , text\_object)

**Returns:**

A visually ordered text object.

When HCL® OneTest™ Data processes bidirectional data, this function changes the order scheme of the text object that is passed as a parameter.

**Example****Example****Example**

```
SETVISUALORDER(SETORIENTATIONRTL(LSRTLData:In1, "FALSE" ), "FALSE" )
```

True or false indicates whether the text object has symmetric swapping enabled. A visual ordering scheme would have the data being read and stored with the start character in the rightmost position in both memory and visually.

## Bit manipulation and testing functions

### SETOFF

The SETOFF function sets to zero a specified bit in a binary number. You can use this function to manipulate (turn off) a bit in a binary number.

**Syntax:**

SETOFF ( single-binary-number-expression, single-integer-expression )

**Meaning:**

SETOFF ( binary\_number\_to\_change, bit\_to\_turn\_off )

**Returns:**

A single binary number

The SETOFF function uses *bit\_to\_turn\_off* to specify the bit of *binary\_number\_to\_change* to be set to the value 0. The result is a binary number item of the same size as *binary\_number\_to\_change*.

The value of *bit\_to\_turn\_off* represents the position of the single bit in *binary\_number\_to\_change* to be set off. (Bits are numbered from left to right, with the leftmost bit being bit 1.) If *bit\_to\_turn\_off* is less than one or greater than the number of bits of *binary\_number\_to\_change*, SETOFF returns *binary\_number\_to\_change* unchanged.

**Examples**

- SETOFF ( A , 16 )

Assume **A** is the two-byte binary value of "1" (which is all zeros except for bit **16**). The binary representation of the value in **A** is 0001.

The result is the two-byte binary value, "0"

- SETOFF ( A , 38 )

Results in the two-byte binary value "1", the original value of **A**, because bit **38** does not exist in **A**.

## SETON

You can use the SETON function to "turn on" a specific bit in a binary number. SETON sets the specified bit in the number to "1".

### Syntax:

SETON ( single-binary-number-expression, single-integer-expression )

### Meaning:

SETON ( binary\_number\_to\_change, bit\_to\_turn\_on )

### Returns:

A single binary number

SETON uses the value of *bit\_to\_turn\_on* to specify the bit of *binary\_number\_to\_change* that should be set to the value 1. The result is a binary number item of the same size as *binary\_number\_to\_change*.

*Bit\_to\_turn\_on* represents the position of the single bit in *binary\_number\_to\_change* to be set on. (Bits are numbered from left to right, with the leftmost bit being bit 1.) If *bit\_to\_turn\_on* is less than one or greater than the number of bits of *binary\_number\_to\_change*, SETON returns *binary\_number\_to\_change*, unchanged.

## Examples

- SETON ( A , 15 )

In this example, assume **A** is the two-byte binary value of 1 (which is all zeros except for bit 16). The binary representation of the value in **A** is 0001.

The function returns the two-byte binary value, 0011, which is the decimal value of 3.

- SETON ( A , 40 )

Returns the two-byte binary value of 1-the original value of A- because bit 40 does not exist in **A**.

## TESTOFF

The TESTOFF function tests a specified bit in a binary number item to see whether it is off.

### Syntax:

TESTOFF ( single-binary-number-expression , single-integer-expression )

### Meaning:

TESTOFF ( binary\_number\_to\_test , bit\_to\_test )

**Returns:**

"True" or "false"

The value of *bit\_to\_test* specifies which bit of *binary\_number\_to\_test* should be tested for the value 0. If *bit\_to\_test* has the value 1, it refers to the leftmost bit of *binary\_number\_to\_test*.

The TESTOFF function returns "true" if the specified bit is off and returns "false" if the specified bit is on.

If *bit\_to\_test* is less than one or greater than the number of bits of *binary\_number\_to\_test*, TESTOFF returns "false".

**Examples**

- TESTOFF (A, 16)

Assume **A** is the two-byte binary value of "1", which is all zeros except for bit **16**. The binary representation of the value in **A** is 0001.

This example returns "false".

- TESTOFF (A, 20)

Returns "false" because bit **20** does not exist.

**TESTON**

The TESTON function tests a specified bit in a binary number to see if it is on.

**Syntax:**

TESTON ( single-binary-number-expression, single-integer-expression )

**Meaning:**

TESTON ( *binary\_number\_to\_test* ,*bit\_to\_test* )

**Returns:**

"True" or "false"

The value of *bit\_to\_test* specifies the bit of *binary\_number\_to\_test* to test for the value 1. If *bit\_to\_test* has the value 1, it refers to the leftmost bit of *binary\_number\_to\_test*.

The function returns "true" if the specified bit is on; it has the value 1. It returns "false" if the specified bit is off; it has the value 0.

If *bit\_to\_test* is less than one or greater than the number of bits of *binary\_number\_to\_test*, TESTON returns "false".

## Examples

- TESTON ( A , 16 )

Assume **A** is the two-byte binary value of "1", which is all zeros except for bit **16**. The binary representation of the value in **A** is 0001.

This example returns "true".

- TESTON ( A , 20 )

Returns "false" because bit **20** does not exist in a two-byte value.

## Conversion functions

### BASE64TOTEXT

Use the BASE64TOTEXT decoding function to convert previously encoded data from BASE64 back to the original text. The data is converted to the character set of the output object used. For example, if the output object is EBCDIC the output will be EBCDIC.

#### Syntax:

**BASE64TOTEXT (single-object-expression)**

#### Meaning:

BASE64TOTEXT (BASE64\_object\_to\_convert)

#### Returns:

A single-text-item

This function receives a text object in Base64-encoded format, converts the object to text, and returns a single data object that represents the original text object that was encoded. If an error occurs during conversion, no data is returned.

### Example

```
BASE64TOTEXT("U29tZSBFeGFtcGxIIERhdGE=")
```

Output: Some Example Data

### BCDTOHEX

The BCDTOHEX function converts an item from binary coded decimal (BCD) format to hexadecimal format. The binary value that is returned always has the high-order byte first, regardless of the operating system. This is useful for testing a specific bit position for a specific value.



**Syntax:**

**BCDTOHEX (single-text-expression, single-integer-expression)**

**Meaning:**

BCDTOHEX (BCD\_item\_to\_convert, length\_of\_output)

**Returns:**

A single binary bytestream

When using the BCDTOHEX function, the input argument is converted from BCD format to its binary value.

Numbers in BCD format have two decimal digits in each byte. Each half-byte, therefore, can contain a binary value from 0000 (which represents the digit 0) through 1001 (which represents the digit 9). Based on this definition, the following behavior applies:

- If any half-byte of the BCD number contains the binary values 1101 or 1111, that half-byte is ignored.
- If any half-byte contains the binary values 1010, 1011, 1100, or 1110, the output of the function is "none".

The length of the output argument is specified as the value of the second input argument. The length must be 1, 2, or 4. If it is any other value, the length is assumed to be 2.

**Examples**

- You can use BCDTOHEX to test a specific bit position for a specific value. For example, BCDTOHEX is useful when the BCD value represents flag bits. The conversion performed by BCDTOHEX results in a predictable location for the flag bits across operating systems where the bytes might otherwise be reversed.
- TESTON (BCDTOHEX ( ProcessIndicator, 2) 10)

Tests for the x`0040' bit in a two-byte result.



**Note:** This same test, using BCDTOINT on a PC, tests the x`4000' bit.

**Related functions**

- BCDTOTEXT
- BCDTOINT
- TEXTTOBCD

**BCDTOINT**

The BCDTOINT function converts an item from BCD (binary coded decimal) format to integer format. The binary value that is returned has a form native to the machine on which the map is being run.

**Syntax:**

BCDTOINT (single-text-expression , single-integer-expression)

**Meaning:**

BCDTOINT (BCD\_item\_to\_convert , length\_of\_output)

**Returns:**

A single-binary-number

*BCD\_item\_to\_convert* is converted from BCD (binary coded decimal) format to its integer value.

Numbers in BCD format have two decimal digits in each byte. Each half-byte, therefore, can contain a binary value from 0000 (which represents the digit 0) through 1001 (which represents the digit 9). Based on this definition, the following behavior applies:

- If any half-byte of the BCD number contains the binary values 1101 or 1111, that half-byte is ignored.
- If any half-byte contains the binary values 1010, 1011, 1100, or 1110, the output of the function is "none".

The length of the output binary number is specified by *length\_of\_output*. The length must be 1, 2, or 4. If it is any other value, the length is assumed to be 2.

BCDTOINT returns a binary value of the form native to the machine on which the map is being run. The value can therefore be used in arithmetic operations or compared to other numeric values. It should *not* be used for bit manipulations, because the order of the bytes in the number is dependent on the operating system on which the map is being run. For example, on a personal computer, the low-order byte is first, while on a mainframe, the high-order byte is first.

**Examples**

- BCDTOINT ( bcdAmount Field , 2 )

If the **bcdAmount Field** contains **x'37'** in BCD format, this example returns the integer value of 37.

**Related functions**

- BCDTOTEXT
- BCDTOHEX
- TEXTTOBCD

**BCDTOTEXT**

The BCDTOTEXT function converts the digits in a BCD (Binary Coded Decimal) item to a text item containing the digits of the BCD-encoded item as a string of characters.

**Syntax:**

BCDTOTEXT (single-text-expression)

**Meaning:**

BCDTOTEXT (BCD\_item\_to\_convert)

**Returns:**

A single text item

*BCD\_item\_to\_convert* is converted from BCD format to a text string containing the digits of the BCD-encoded value as a string of characters.

Numbers in BCD format have two decimal digits in each byte. Each half-byte, therefore, can contain a binary value from 0000 (which represents the digit 0) through 1001, which represents the digit 9). Based on this definition, the following behavior applies:

- If any half-byte of the BCD number contains the binary value 1101 or 1111, that half-byte is ignored.
- If the BCD item contains the binary value 1010, 1011, 1100, or 1110, the output of the function is "none".

**Examples**

- `BCDTOTEXT ( Qty:Item )`

If **Qty** is `x'1234'`, the result is 1234.

- `BCDTOTEXT ( DiscountAmt )`

If **DiscountAmt** is `x'0123'`, the result is 0123.

- `BCDTOTEXT ( TotalDollars )`

If **Total** is `x'F123'`, the result is 123.

**Related functions**

- `BCDTOHEX`
- `BCDTOINT`
- `TEXTTOBCD`

**CONVERT**

The `CONVERT` function replaces each byte of a byte stream or text expression with a byte from another byte stream or text expression. The decimal value of each byte of the first argument is used to locate the corresponding byte in the second argument that will replace it.

**Syntax:**

`CONVERT ( single-byte-stream-or-text-expression , single-byte-stream-or-text-expression )`

**Meaning:**

`CONVERT ( bytes_to_replace , replacement_bytes )`

**Returns:**

A single byte stream or text item

The CONVERT function replaces each byte of *bytes\_to\_replace* with a byte from *replacement\_bytes*. The byte chosen from *replacement\_bytes* is the one whose index is the decimal value of *bytes\_to\_replace*. The first byte of *replacement\_bytes* has the index value of zero. If there is no corresponding byte in *replacement\_bytes*, CONVERT returns "none".

You can use the CONVERT function to convert ASCII to EBCDIC or EBCDIC to ASCII data. See the **convert.mms** map source file in the **examples\general\portdata** folder of the product installation directory. One of the maps in **convert.mms** is **ASCII\_TO\_EBCDIC**, which converts ASCII to EBCDIC. The **EBCDIC\_TO\_ASCII** map converts EBCDIC to ASCII.

ASCII to EBCDIC conversion can be performed automatically by defining the appropriate data language (ASCII or EBCDIC) for each data item.

### Examples

- CONVERT ( SYMBOL ( 0 ) ,"AB" )

Returns A

- CONVERT ( SYMBOL ( 1 ) ,"AB" )

Returns B

- CONVERT ( SYMBOL ( 2 ) ,"AB" )

Returns "none"

- CONVERT ( ASCII , ATOETable )

Converts ASCII text to EBCDIC based on values in a table named **ATOETable**

## DATETONUMBER

Use DATETONUMBER to perform arithmetic on dates.

The DATETONUMBER function returns an integer that results from counting the number of days from December 31, 1864, to the specified date.

#### Syntax:

DATETONUMBER (single-date-expression)

#### Meaning:

DATETONUMBER (date\_to\_convert)

#### Returns:

A single integer

The DATETONUMBER function converts a date to an integer. The resulting integer represents the number of days since December 31, 1864, where using DATETONUMBER with a date of January 1, 1865 returns the integer value 1.

If the input argument is in error, the function returns the value "none".

If the date format specified by the input argument does not include century, the century is determined based on the **Century** map setting using the **CCLookup** parameter or the current century.

## Examples

- `DaysBetween = DATETONUMBER ( StopDate ) - DATETONUMBER ( StartDate )`

This expression could be used in a map rule to produce a value for **DaysBetween**. It converts the **StopDate** and the **StartDate** to integers, then subtracts the resulting two integers, and returns the result as **DaysBetween**.

## Related functions

- ADDBAYS
- NUMBERTODATE

## DATETOTEXT

The DATETOTEXT function converts a date object or expression to a text item.

### Syntax:

`DATETOTEXT (single-date-expression)`

### Meaning:

`DATETOTEXT (date_to_convert)`

### Returns:

A single text item

If *date\_to\_convert* is a date object name, this returns the date as a text item formatted according to the presentation of the date object.

If *date\_to\_convert* is a date expression produced by a function, this returns the date as a text item formatted according to the presentation of the output argument of that function.

## Examples

- `DATETOTEXT ( ShipDate )`

In this example, **ShipDate** is converted from a date to text. If **ShipDate** has a CCYYMMDD presentation, the resulting text item will have that presentation, as well.

- `DATETOTEXT ( CURRENTDATETIME ( "{MM/DD/CCYY}" ) )`

In this example, CURRENTDATETIME evaluates and returns a date in MM/DD/CCYY format. Then DATETOTEXT evaluates and returns a text string that is that date in MM/DD/CCYY format.

For example, use DATETOTEXT, to do text concatenation. The FROMDATETIME function provides greater flexibility in specifying the format of the resulting text item.

## Related Functions

- FROMDATETIME
- NUMBERTOTEXT
- TEXT
- TEXTTODATE
- TEXTTONUMBER
- TEXTTOTIME
- TIMETOTEXT
- TODATETIME

## FROMBASETEN

You can use FROMBASETEN when you need to convert numbers to a base other than 10.

The FROMBASETEN function converts an integer to a text item that can be interpreted as a number, using positional notation of the base specified.

### Syntax:

FROMBASETEN (single-integer-expression, single-integer-expression)

### Meaning:

FROMBASETEN (positive\_integer\_to\_convert, base\_to\_convert\_to)

### Returns:

A single text item

FROMBASETEN returns a text item that results from converting *positive\_integer\_to\_convert* to a text item that can be interpreted as a number using positional notation of the base specified by *base\_to\_convert\_to*.

If *base\_to\_convert\_to* is less than 2 or greater than 36, FROMBASETEN evaluates to "none". Resulting text item characters A-Z are interpreted as digits having decimal values from 10-35, respectively. The characters returned are uppercase.

## Example

- FROMBASETEN (18, 2)

Returns the value 10010

- FROMBASETEN (123, 8)

Returns the value 173

### Related function

- TOBASETEN

## FROMDATETIME

The FROMDATETIME function converts a date/time item to a text string of a specified format. For example, you can use FROMDATETIME to convert a date-time item into a string for parsing or concatenation. You can also use this function to access the individual parts of a date or time, such as the month, day, year, and so forth.

### Syntax:

```
FROMDATETIME (single-character-date-time-item
[ , single-text-expression ] )
```

### Meaning:

```
FROMDATETIME (date_time [ , date_time_format_string ] )
```

### Returns:

A single character text item

FROMDATETIME returns the date/time specified by *date\_time* as a text item with the format specified by *date\_time\_format\_string*. If *date\_time\_format\_string* is not specified, *date\_time* will be returned in the same format as the *date\_time*.

The *date\_time\_format\_string* must conform to the date/time format strings as described in the "Format strings" topic.

### Examples

- HeaderLine="Today is" + FROMDATETIME (CURRENTDATE ()), "{MON DD, CCYY}")

If the current system date is March 3, 1999, this rule evaluates to Today is Mar 03, 1999.

- FROMDATETIME ( TransactionTimeStamp Column::TransHistory )

If the value of **TransactionTimeStamp Column** is 10:14 A.M. on February 3, 2000 and its format is defined as "{CCYYMMDDHH24MM}", this rule evaluates to 200002031014.

- EXTRACT (Expense:Report, FROMDATETIME (Date:Expense:Report, "{MM}") = "03")

In this example, the FROMDATETIME function is used in the condition expression of the EXTRACT function to identify all expenses in the month of March.

## Related functions

- CURRENTDATE
- CURRENTDATETIME
- CURRENTTIME
- DATETOTEXT
- TODATETIME

## FROMNUMBER

The FROMNUMBER function converts a number to a text string of a specified format.

Use FROMNUMBER when you need to convert a number item into a string for parsing or concatenation.

### Syntax:

```
FROMNUMBER (single-character-number-item
  [, single-text-expression])
```

### Meaning:

```
FROMNUMBER (number_to_convert [, number_format_string])
```

### Returns:

A single character text item

FROMNUMBER returns the number specified by *number\_to\_convert* as a text item with the format specified by *number\_format\_string*. If *number\_format\_string* is not specified, *number\_to\_convert* will be returned in the same format as the *number\_to\_convert*.

The *number\_format\_string* must conform to the number format strings as described in the "Format strings" topic.

## Examples

- Greeting = "You are caller number " + FROMNUMBER (SeqNo:::History, "{#;###}") + "!"

If the value of **SeqNo** is 2348192, this rule evaluates to "You are caller number 2,348,192!"

- "\$" + FROMNUMBER(CurrentRate Field:::Schedule, "{#;###'.2##2}")

If the value of **CurrentRate Field** is 15.875, "\$15.88".

## Related functions

- DATETONUMBER
- NUMBERTODATE
- NUMBERTOTEXT
- TONUMBER



## HEXTEXTTOSTREAM

HEXTEXTTOSTREAM is the reverse of STREAMTOHEXTEXT. You can use the HEXTEXTTOSTREAM function to assign a binary text value to a character text item represented by hexadecimal pairs.

HEXTEXTTOSTREAM returns a binary text stream whose value is the evaluation of input character text represented by hexadecimal pairs.

### Syntax:

HEXTEXTTOSTREAM (single-text-expression)

### Meaning:

HEXTEXTTOSTREAM (series\_of\_hex\_pairs)

### Returns:

A single byte stream item

This function returns a binary text stream item whose value is the evaluation of input character text in *series\_of\_hex\_pairs*, ignoring <WSP> characters between the hexadecimal pairs. White space characters include space, horizontal tab, carriage return, and line feed characters.

### Input formats

The following table shows an example of input in its character text representation as viewed through the character editor, and in its ASCII code representation (binary text stream) as viewed through the hex editor. Each pair of binary text in the hex view represents one character in the character view of the character text.

Input ("41 42 43 44")	Editor View	Value
Character text (hex pairs)	Character	"41 42 43 44"
ASCII code representation (binary text stream)	Hex	0x3431203432203433203434

### Examples

- HEXTEXTTOSTREAM ("41 42 43 44")

Returns the evaluated value of the input (ASCII) character text string "41 42 43 44" as the output (ASCII) character text string "ABCD" as viewed in the character editor. (The hex view of the input is 0x3431203432203433203434. The hex view of the output is 0x41424344.)

- HEXTEXTTOSTREAM ("0D 0A 00")

Returns the evaluated value of the input (ASCII) character text string "0D 0A 00" as the output (ASCII) character text string "<CR><LF><NULL>" as viewed in the character editor. (The hex view of the input is 0x3044203041203030. The hex view of the output is 0x0D0A00.)

See Design Studio Introduction documentation for a list of special symbols.

## Related functions

- SYMBOL
- STREAMTOHEXTEXT

## INT

You can use the INT function when you need only the integer portion of a number.

### Syntax:

INT (single-number-expression)

### Meaning:

INT (number\_to\_convert)

### Returns:

A single integer

INT returns the integer portion of a number. The result is the integer part of *number\_to\_convert*. Any fractional part after the decimal point is dropped.

## Examples

- INT (1.45)

Returns 1

- INT (3.6)

Returns 3

- INT (Purchase:Amt - Discount:Amt)

Subtracts **Discount:Amt** from **Purchase:Amt** and returns the result as a whole number.

## Related functions

- MOD
- ROUND
- TRUNCATE

## NUMBERTODATE

You can use the NUMBERTODATE function to perform a calculation on a date.

NUMBERTODATE converts an integer to a date, where the integer is the number of days from December 31, 1864, to the specified date. Only positive, non-zero values can be specified as valid parameters.

**Syntax:**

NUMBERTODATE (single-integer-expression)

**Meaning:**

NUMBERTODATE (integer\_to\_convert)

**Returns:**

A single date

After converting an integer to a date, if the result is being assigned to a date object, the resulting date is in the presentation for that output. If the resulting date is not being assigned to an object, it has a CCYYMMDD presentation.

When the year exceeds four digits, the output will display hash characters (#) for the CCYY values because the field is defined to be only four digits in length.

**Examples**

- NUMBERTODATE (StartDate)

This example converts the **StartDate** value from an integer to a date that is in CCYYMMDD format.

**Related functions**

- ADDDAYS
- DATETONUMBER
- FROMDATETIME
- TODATETIME

**NUMBERTOTEXT**

The NUMBERTOTEXT function converts a character number to a text item that looks like the original object.

You can use NUMBERTOTEXT when you need an object that is defined as a number converted to an object defined as text. This is useful when you need to concatenate text, however, the FROMNUMBER function provides greater flexibility in specifying the format of the resulting text item.

**Syntax:**

NUMBERTOTEXT (single-number-expression)

**Meaning:**

NUMBERTOTEXT (number\_to\_convert)

**Returns:**

A single text item

The resulting text looks like the input argument. The result is truncated, if necessary.

## Examples

- NUMBERTOTEXT (ROUND (1000 - 24.75, 3))

This example converts the result of the calculation (rounded to 3 decimal places) to text, resulting in 975.250.

- NUMBERTOTEXT (PurchaseNumber)

This example converts **PurchaseNumber** from a number to text.

## Related functions

- FROMNUMBER
- TEXTTONUMBER
- TODATETIME
- TONUMBER

## PACK

The PACK function converts an integer to a text item that can be interpreted as a packed decimal number.

The sign values for packed data are as follows:

- C for positive (+)
- D for negative (-)
- F for unsigned, which is read as positive

### Syntax:

PACK (single-integer-expression)

### Meaning:

PACK (integer\_to\_conver)

### Returns:

A single text item

In a packed decimal number, each half-byte is a digit, except for the last half-byte of the rightmost byte, which contains a sign.

## Examples

- PACK (314)

Returns "1L" and results in the hex value 31 4C (which, in ASCII, looks like 1L)

- PACK ((Unit Price \* Quantity) \* 100)

In this example, the packed number has two implied decimal places. Because PACK does not accept decimal places, including implied ones, the nested arithmetic expression, **Unit Price \* Quantity**, is multiplied by 100 before rounding.

Define items as having a packed decimal number presentation. Then, when mapping to or from these items, the conversion to and from packed decimal is automatically performed as needed.

### Related function

- UNPACK

## PACKAGE

The PACKAGE function converts a group or item object to a text item, including its initiator, terminator, and any delimiters it contains.

#### Syntax:

PACKAGE (single-object-expression)

#### Meaning:

PACKAGE (object\_to\_convert)

#### Returns:

A single text item

The PACKAGE function converts *object\_to\_convert*, which must be a type reference to a text item, including the type reference's initiator, terminator, and all delimiters. PACKAGE differs from TEXT in that it includes the initiator and terminator of the specified type reference.

### Examples

- PACKAGE (Record:Card)

Returns: #1339X10A,491.38,Green,42x54@

For this example, the group **Record** has an initiator of "#", a terminator of "@" and a delimiter of ",". The data looks like this: "#1339X10A,491.38,Green,42x54@".

### Related functions

- DATETOTEXT
- NUMBERTOTEXT
- SERIESTOTEXT
- TIMETOTEXT

- TEXT

PACKAGE differs from TEXT because it includes the initiator and terminator of the input object.

## QUOTEDTOTEXT

The QUOTEDTOTEXT decoding function converts a single (text or binary) data object in Quoted-Printable format to a single data object that represents the original text or binary object.

**Syntax:**

QUOTEDTOTEXT (single-object-expression)

**Meaning:**

QUOTEDTOTEXT (quoted\_printable\_object\_to\_convert)

**Returns:**

A single-text-item

You can use this function to decode data that was previously encoded to RFC 1767 Quoted-Printable encoding. Any hexadecimal representations are decoded to the appropriate ASCII character and any soft breaks are removed.

If an error occurs during conversion, no data is returned.

### Example

```
QUOTEDTOTEXT("Unencoded data=0CEncoded")
```

Output: Unencoded data<FF>Encoded

## SERIESTOTEXT

You can use the SERIESTOTEXT function to project your input data as a series and to interpret it as a text item for output.

SERIESTOTEXT converts a contiguous or non-contiguous series to a text item.

**Syntax:**

SERIESTOTEXT (series-object-expression)

**Meaning:**

SERIESTOTEXT (series\_to\_convert)

**Returns:**

A single text item

SERIESTOTEXT returns a text item containing the concatenation of the series of the input argument, including nested delimiters but excluding initiators and terminators.

## Examples

In this example, you have the following data that represents bowler information for a bowling league:

Andrews, Jessica:980206:JBC:145:138:177:159

Little, Randy:980116:BBK:175:168

Wayne, Richard:980102:JBC:185:204:179:164:212

Each record consists of the bowler's name, the date of their last game played, a team code and one or more bowling scores. **Record** is defined as a group that is infix-delimited by a colon.

Using the rule:

= SERIESTOTEXT (Score Field:Bowler:Input)

the following results are produced, which is the concatenation of all of the scores for all of the bowlers, even though the scores are not all contiguous within the data:

145138177159175168185204179164212

However, if the rule was changed, for instance, to concatenate the list of scores to the bowler's name:

= BowlerName Field:Bowler:Input + " ->" + SERIESTOTEXT (Score Field:Bowler:Input)

the following output would be produced:

Andrews, Jessica -> 145138177159

Little, Randy -> 175168

Wayne, Richard -> 185204179164212

In this example, you have an input number that is of variable size, followed by a name. There is no syntax that separates the number from the name. You can define the number as a group with **Byte(s)** as a component and provide a component rule for **Byte(s)**, such as:

ISNUMBER (\$)

## Related functions

- PACKAGE
- TEXT

## STREAMTOHEXTEXT

Use STREAMTOHEXTEXT to assign a character text value to a binary text stream item.

The STREAMTOHEXTEXT function returns a character text string represented by hex pairs whose value is the evaluation of an input binary text stream.

This function is the reverse of the HEXTEXTTOSTREAM function.

**Syntax:**

STREAMTOHEXTEXT (single-byte-stream-item)

**Meaning:**

STREAMTOHEXTEXT (single-byte-stream-item)

**Returns:**

A series of hex pairs

STREAMTOHEXTEXT returns a string of hex pairs whose value is the evaluation of input binary text in *series\_of\_hex\_pairs*.

**Input formats**

The following table shows an example of input in its ASCII code representation (binary text stream) as viewed through the hex editor, and in its character text representation as viewed through the character editor. Each four-character grouping of binary text in the hex view represents one character in the character view.

Input (0x41424344)	Editor View	Value
ASCII code representation (binary text stream)	Hex	0x41424344
	Character	"ABCD"

**Examples**

- STREAMTOHEXTEXT (0x41424344)

Returns the evaluated value of the input (ASCII) binary text stream **0x41424344** as the output (ASCII) binary text stream 0x3431343234333434 as viewed in the hex editor. (The character view of the input is "ABCD". The character view of the output is "41424344".)

- STREAMTOHEXTEXT (0x0D0A00)

Returns the evaluated value of the input binary text stream **0x0D0A00** as the output 0x304430413030 as viewed in the hex editor. (The character view of the input is "<CR><LF><NULL>". The character view of the output is "0D0A00".)

**Related functions**

- HEXTEXTTOSTREAM
- SYMBOL



## SYMBOL

You can use SYMBOL to include a symbol in your output.

The SYMBOL function returns a one-byte character that is the ASCII character equivalent of a specified decimal value.

**Syntax:**

SYMBOL ( single-integer-expression )

**Meaning:**

SYMBOL ( decimal\_value )

**Returns:**

A single one-byte text item

The value of the resulting item is the ASCII character equivalent of *decimal\_value*. Valid input values are 0 to 255. Values outside of this range return "none".

A listing of decimal values (0-127) and their ASCII character equivalents is included in the Design Studio Introduction documentation.

### Examples

- SYMBOL ( 13 )

Produces a carriage return

- SYMBOL ( 13 ) + SYMBOL ( 10 )

Produces a carriage return/linefeed

You can accomplish the same result using the angle-brackets around the decimal value. For example, the second example above would be equivalent to <CR><LF> or <<0D>><<0A>>

### Related functions

- HEXTEXTTOSTREAM
- STREAMTOHEXTEXT

## TEXTTOBASE64

The TEXTTOBASE64 encoding function converts a text item to Base64 format.

**Syntax:**

TEXTTOBASE64 (single-object-expression)

**Meaning:**

TEXTTOBASE64 (text\_object\_to\_convert)

**Returns:**

A single-text-item

The TEXTTOBASE64 function receives a text object and uses the MIME implementation to convert the text to Base64 format. The MIME implementation adds the `<CR><LF>` character sequence after every 76 bytes and also at the end of the output string.

TEXTTOBASE64 returns a single data object that represents the Base64 encoding of the original text object. If an error occurs during conversion, no data is returned.

**Example**

TEXTTOBASE64("Some Example Data")

Output: U29tZSBFeGFtcGxIIERhdGE=

**TEXTTOBCD**

The TEXTTOBCD function converts a text item from decimal digits to BCD (Binary Coded Decimal) format.

**Syntax:**

TEXTTOBCD ( single-integer-text-expression )

**Meaning:**

TEXTTOBCD ( text\_to\_be\_converted )

**Returns:**

A single BCD-formatted text item

TEXTTOBCD converts *text\_to\_be\_converted* (which consists of decimal digits) to BCD format. In this format, each byte contains two decimal digits represented as binary numbers. If there is an odd number of decimal digits in the input, the high-order half-byte of the leftmost output byte will contain the decimal value 15 (hex "F").

If anything other than a decimal digit is encountered in the input, TEXTTOBCD returns "none".

**Examples**

- TEXTTOBCD ( "1234" )

Returns the hexadecimal value x'1234'

- TEXTTOBCD ( "123A" )

Returns "none"

- TEXTTOBCD ( "123" )

Returns the hexadecimal value x'F123'

In this example, the values shown as input ("123") are meant to represent character items in the native character set to the machine on which the map is running. On a personal computer, "123" would contain the ASCII characters for the digits that have the hexadecimal values "31", "32", and "33". The output, described as "the hexadecimal value `F123'", consists of the two binary bytes "F1" and "23".

On an IBM mainframe the input string would contain EBCDIC characters for the digits that have the hexadecimal values "F1", "F2", "F3", but the output would be the same as the personal computer output.

## Related functions

- BCDTOHEX
- BCDTOINT
- BCDTOTEXT

## TEXTTODATE

The TEXTTODATE function converts text item from CCYYMMDD or YYMMDD format to a date.

You can use the TEXTTODATE to convert a text item to a date, however, the TODATETIME function provides greater flexibility for specifying the resulting date-time format .

### Syntax:

```
TEXTTODATE ( single-text-expression )
```

### Meaning:

```
TEXTTODATE ( text_to_convert_to_date )
```

### Returns:

A single date

The *text\_to\_convert\_to\_date* must be in either CCYYMMDD or YYMMDD format. If the *text\_to\_convert\_to\_date* is in error (for example, it is not a valid date), the result is "none".

If *text\_to\_convert\_to\_date* is in YYMMDD format and is being assigned to a date format that includes a century, the century is determined based on the century run option setting using the **CCLookup** parameter or the current century.

## Examples

- `TEXTTODATE ( "990114" )`

Returns a single date item with the value of January 14 in the year 99

- `TEXTTODATE ( OrderDate )`

Returns **OrderDate** as a single date item.

## Related functions

- DATETOTEXT
- TEXTTOTIME
- NUMBERTO-  
TEXT
- TIMETOTEXT
- TEXTTONUM-  
BER
- TO-  
DATETIME

## TEXTTONUMBER

Use TEXTTONUMBER to convert text to a number.

The TONUMBER function provides greater flexibility for specifying the format of the text item that is to be converted to a number.

### Syntax:

TEXTTONUMBER ( single-text-expression )

### Meaning:

TEXTTONUMBER ( text\_to\_convert\_to\_number )

### Returns:

A single character number

The *text\_to\_convert\_to\_number* must be in integer or ANSI-formatted (floating point) presentation. The resulting number looks like the input argument, however, nonsignificant zeroes to the right of the decimal separator will be truncated. If the input argument is in error (for example, it is not a recognizable as a valid number), the result is "none".

When specified as in ANSI-formatted presentation, the text string must meet the following requirements:

- The decimal point can be a period, a comma, or "none".
- The leading sign can be a plus sign, a minus sign, or "none".
- No thousands separator is allowed.

## Examples

- TEXTTONUMBER (OrderQty)

Returns **OrderQty** as a character number item

## Related functions

- DATETOTEXT
- FROMNUMBER
- NUMBERTOTEXT
- TEXTTODATE
- TEXTTOTIME
- TIMETOTEXT

## TEXTTOQUOTED

The TEXTTOQUOTED encoding function converts a text or binary item to Quoted-Printable format.

### Syntax:

```
TEXTTOQUOTED (single_object_expression)
```

### Meaning:

```
TEXTTOQUOTED (object_to_convert)
```

### Returns:

A single text or binary item

You can use this function to encode data that largely consists of octets that correspond to printable characters in the US-ASCII character set. It encodes the data in such a way that the resulting octets are unlikely to be modified by mail transport.

Quoted-Printable lines cannot be longer than 76 characters. Data with lines greater than 76 characters are broken up and indicated with soft breaks of "=<CR><LF>".

If an error occurs during conversion, no data is returned.

### Example

This example converts the <FF> (form feed) character, into a hexadecimal representation, and indicates the end of data with a soft break.

```
TEXTTOQUOTED("Unencoded data<FF>Encoded")
```

Output: Unencoded data=0CEncoded=

The data is converted using the Quoted-Printable encoding as per RFC 1767.

## TEXTTOTIME

Use TEXTTOTIME when you want to convert an object defined as text that is in HHMM or HHMMSS presentation, to an item defined as time. For greater flexibility, use the TODATETIME function for specifying the format of the text item that is to be converted to a date/time.

**Syntax:**

TEXTTOTIME ( single-text-expression )

**Meaning:**

TEXTTOTIME ( text\_to\_convert\_to\_time )

**Returns:**

A single time

The *text\_to\_convert\_to\_time* must be in HHMM or HHMMSS presentation. HH is a two-digit hour in a 24-hour format. If the result is being assigned to a time object, the resulting time looks like the output object. Otherwise, the resulting time looks like the input argument. If the input argument is in error (for example, it is not a valid time), the result is "none".

**Examples**

- TEXTTOTIME ( CallTime )

Returns **CallTime** as a time item

**Related functions**

- DATETOTEXT
- TEXTTONUMBER
- FROMDATETIME
- TIMETOTEXT
- NUMBERTO-TEXT
- TODATETIME
- TEXTTODATE

**TIMETOTEXT**

You can use the TIMETOTEXT function to perform text concatenation. For greater flexibility, use the FROMDATETIME function for specifying the format of the resulting text item.

TIMETOTEX converts a time object or expression to a text item.

**Syntax:**

TIMETOTEXT ( single-time-expression )

**Meaning:**

TIMETOTEXT ( time\_to\_convert\_to\_text )

**Returns:**

A single text item

If *time\_to\_convert\_to\_text* is a time object name, this returns the time as a text item formatted according to the presentation of the input date object.

If *time\_to\_convert\_to\_text* is a time expression produced by a function, this returns the time as a text item formatted according to the presentation of the output argument of that function.

**Examples**

- TIMETOTEXT ( LeadTime )

In this example, **LeadTime** is converted from a time to text. If **LeadTime** has an HH:MM presentation, the resulting text item will be of that presentation.

- TIMETOTEXT ( CURRENTDATETIME ( "{HH:MM:SS}" ) )

Here, CURRENTDATETIME evaluates and returns a time in HH:MM:SS format. Then, TIMETOTEXT evaluates and returns a text string that is that time in HH:MM:SS format.

**Related functions**

- DATETOTEXT
- FROMDATETIME
- NUMBERTOTEXT
- TEXTTODATE
- TEXTTONUMBER
- TEXTTOTIME
- TODATETIME

**TOBASETEN**

The TOBASETEN function converts a text item that can be interpreted as a number, using positional notation of the base specified, to a base 10 number.

**Syntax:**

TOBASETEN ( single-text-expression, single-integer-expression )

**Meaning:**

TOBASETEN ( text\_to\_convert, base\_to\_convert\_from )

**Returns:**

A single integer

TOBASETEN returns a number that results from converting *text\_to\_convert* that can be interpreted as a number, using positional notation of the base specified by *base\_to\_convert\_from*, to its base 10 representation. Text item characters A-Z are interpreted as digits having decimal values from 10-35, respectively.

If *base\_to\_convert\_from* is less than 2 or greater than 36, TOBASETEN evaluates to "none". If *text\_to\_convert* contains a character that is not alphanumeric or is not in the range specified by *base\_to\_convert\_from*, TOBASETEN returns "none".

## Examples

- TOBASETEN ( "A", 16 )

Returns the value 10

- TOBASETEN ( "10", 36 )

Returns the value 36

- TOBASETEN ( "A0", 15 )

Returns the value 150

- TOBASETEN ( "A0", 5 )

Returns the value "none"

## Related functions

- FROMBASETEN

## TODATETIME

The TODATETIME function converts a text string of a specified format to a date-time item.

### Syntax:

```
TODATETIME ( single-character-text-expression
            [, single-text-expression ] )
```

### Meaning:

```
TODATETIME ( text_to_convert [, date_time_format_string ] )
```

### Returns:

A single character date item

TODATETIME returns the date-time that corresponds to the value specified by *text\_to\_convert*, which is in the format specified by *date\_time\_format\_string*. If *date\_time\_format\_string* is not specified, it will be assumed that *text\_to\_convert* is in [CCYYMMDDHH24MMSS] format.

The *date\_time\_format\_string* must conform to the date-time format strings as described in "Format strings".



## Examples

- `TODATETIME ( "05/14/1999@10:14pm" , "{MM/DD/CCYY}@{HH12:MMAM/PM}" )`

In this example, a text string containing a date and time is converted to a date-time item.

- `RptDate = TODATETIME ( RIGHT ( GETRESOURCE() , 8 ) , "CCYYMMDD" )`

Assume that you receive a file that contains historical data. The name of the file identifies the date of the historical data. For example, a filename of **19960424** indicates that the data was produced on April 24, 1996. To map this date to **RptDate**, the `TODATETIME` function could be used with the `RIGHT` and `GETRESOURCE` functions.

## Related functions

- `CURRENTDATE`
- `CURRENTDATETIME`
- `CURRENTTIME`
- `TEXTTODATE`
- `TEXTTOTIME`

## TONUMBER

The `TONUMBER` function converts a text string of a specified format to a number.

### Syntax:

```
TONUMBER ( single-character-text-expression
          [ , single-text-expression ] )
```

### Meaning:

```
TONUMBER ( text_to_convert [ , number_format_string ] )
```

### Returns:

A single character number item

`TONUMBER` returns the number that corresponds to the value specified by *text\_to\_convert*, which is in the format specified by *number\_format\_string*. If *number\_format\_string* is not specified, it will be assumed that *text\_to\_convert* is in ANSI decimal format (for example, "{L-####['.##]}").

The *number\_format\_string* must conform to the number format strings as described in the "Format strings" topic.

## Examples

- `TONUMBER(text_to_convert, "{L+'$'#,###}")`

**L+\$** indicates the leading dollar sign is positive. That leading sign and the comma separators are removed when the text is converted to a number.

Input String: \$123,000,000

Output: 123000000

- `TONUMBER(text_to_convert, "{####T-}")`

Four number signs are required for each whole number, regardless of the actual number of digits in the number.

Input string:	Output:	Note:
12345-	-12345	The output becomes a negative number.
67890	67890	No change occurs.
345-	-345	The output becomes a negative number.

- `TONUMBER(text_to_convert, "{####T+'K'-}")`

If an invalid character, such as an X, is encountered, nothing is returned.

If a **K** is encountered, it is treated as a positive indicator.

Input string:	Output:	Note:
11212-	-11212	The output becomes a negative number.
67890X		The X is an invalid character. No number is returned.
54354	54354	No change occurs.
34567K	34567	The K is recognized as a positive sign. The character is removed and the number is returned as a positive.
345-	-345	The output becomes a negative number.

- `TONUMBER(text_to_convert, "{L-'('#',####T-}")`

The parentheses indicating a negative number are removed and replaced with a negative sign.

Comma separators are removed when the text is converted to a number.

Input string:	Output:	Note:
(12,345)	-12345	The output becomes a negative number. The comma separator is removed.
67,890	67890	The comma separator is removed.
(345)	-345	The output becomes a negative number.

- `TONUMBER(text_to_convert, "{#[']####['.###5]T+'K'-}")`

The optional comma separators are removed, but the decimal points and decimal values are retained.

<b>Input string:</b>	<b>Output:</b>	<b>Note:</b>
54,345.098	54354.098	The comma separator is removed.
67890.0X		The X is an invalid character. No number is returned.
11213-	-11213	The output becomes a negative number.
34567K	34567	The K is recognized as a positive sign. The character is removed and the number is returned as a positive.
345.1-	-345.1	The output becomes a negative number.

### Related functions

- DATETONUMBER
- FROMNUMBER
- NUMBERTODATE
- NUMBERTOTEXT

## UNPACK

You can use the UNPACK function to do arithmetic with a packed decimal number or to move a packed decimal value into a numeric item.

UNPACK converts text that can be interpreted as a packed decimal number to a signed integer item.

The sign values for packed data are as follows:

- C for positive (+)
- D for negative (-)
- F for unsigned, which is read as positive

#### Syntax:

```
UNPACK ( single-fixed-size-text-expression )
```

#### Meaning:

```
UNPACK ( text_to_unpack )
```

#### Returns:

A single signed integer

UNPACK returns a signed integer representing the value *text\_to\_unpack*, which is a packed decimal number. If the *text\_to\_unpack* cannot be interpreted as a valid packed decimal, UNPACK evaluates to "none".

In a packed decimal number, each half-byte is a digit, except for the last half-byte of the rightmost byte, which contains a sign.

## Examples

- `UNPACK ( "1L" )` returns 314

The ASCII string "1L" in hex is 31 4C, which, when interpreted as a packed number, results in (positive) 314. This example returns the value "+314".

The hexadecimal representation of the value "1L" is x`14C', where C in the rightmost half-byte represents a positive sign.

- `UNPACK ( UnitPrice ) / 100 * QuantityOrdered`

**UnitPrice** is unpacked and divided by 100 (to convert it from an integer to a number with two decimal places) and then multiplied by the *QuantityOrdered*.

You can define items as having a packed decimal number presentation. Then, when mapping to or from these items, the conversion to and from packed decimal is automatically performed as needed.

## Related functions

- `PACK`

## UNZONE

You can use the UNZONE function to convert a text item that represents a zoned (signed) number to an integer.

UNZONE converts a text item that can be interpreted as a number with a super-imposed sign in the rightmost byte (called zoned or signed data) to a signed integer item.

### Syntax:

`UNZONE ( single-text-expression )`

### Meaning:

`UNZONE ( text_to_unzone )`

### Returns:

A single integer

UNZONE returns a signed integer representing the value *text\_to\_unzone* that is an integer in zoned (signed) format. If the *text\_to\_unzone* cannot be interpreted as a valid zoned number, UNZONE evaluates to "none".

Zoned integers have a series of digits except for the rightmost byte. The rightmost byte is a digit with a super-imposed sign. See "Positive zoned values" for a list of rightmost byte values.

## Examples

- UNZONE ("123D")

Returns 1234

- UNZONE ("1234")

Returns 1234

- UNZONE ( TaxRate ) / 1000 \* Income

**TaxRate** is converted from zoned format to a signed decimal and divided by 1000 (to convert it to a number with three decimal places), and then multiplied by **Income**.

You can define items as having a zoned character number presentation. Then, when mapping to or from these items, the conversion to and from zoned decimal is automatically performed as needed.

## Related functions

- ZONE

## ZONE

Use ZONE to convert a number to a zoned (signed) number.

The ZONE function converts a signed integer item to a text item that can be interpreted as a number with a superimposed sign in the rightmost byte (called zoned or signed).

You can define items as having a zoned character number presentation. Then, when mapping to or from these items, the conversion to and from zoned decimal is performed automatically, as needed.

### Syntax:

ZONE ( single-integer-expression , single-integer-expression )

### Meaning:

ZONE ( integer\_to\_convert , sign\_indicator )

### Returns:

A single text item

ZONE returns a text string that represents a zoned (signed) number representing *integer\_to\_convert*.

Zoned integers have a sequence of digits except for the rightmost byte. The rightmost byte is a digit with a superimposed sign. The *sign\_indicator* specifies whether a super-imposed sign is required for positive integers, where 0 specifies no sign is required and any other value specifies that a sign is required for positive integers.

The following tables show positive and negative values of the numbers 1230 to 1239, with a sign indicator of 0 (no sign is required for positive values) or a sign of 1 (include a sign for both positive and negative values) in the rightmost byte.

**Table 12. Positive Zoned Values**

Integer Value	sign_indicator = 0	sign_indicator = 1
1230	1230	123{
1231	1231	123A
1232	1232	123B
1233	1233	123C
1234	1234	123D
1235	1235	123E
1236	1236	123F
1237	1237	123G
1238	1238	123H
1239	1239	123I

**Table 13. Negative Zoned Values**

Integer Value	sign_indicator = 0	sign_indicator = 1
-1230	123}	123}
-1231	123J	123J
-1232	123K	123K
-1233	123L	123L
-1234	123M	123M
-1235	123N	123N
-1236	123O	123O
-1237	123P	123P
-1238	123Q	123Q
-1239	123R	123R

## Examples

- ZONE ( 1234 , 0 )

Returns 1234

- ZONE ( 1234 , 1 )

Returns 123D

- ZONE ( -1234 , 1 )

Returns 123M

- ZONE ( INT ( UnitPrice \* 100 ) , 1 )

**UnitPrice** is multiplied by 100 to move the first two decimal places to the left of the decimal sign. The result of this calculation is converted to an integer using the INT function. Finally, the result of the INT is converted to zoned format, using a sign for positive values.

### Related function

- UNZONE

## Date/time functions

### ADDDAYS

The ADDDAYS function adds a specified number of days to a given date.

You cannot use ADDDAYS on a date/time object that does not specify the day of the month because an error will occur.

#### Syntax:

ADDDAYS (single-date-expression, single-integer-expression)

#### Meaning:

ADDDAYS (any\_date, number\_of\_days\_to\_add)

#### Returns:

A single date

The ADDDAYS function returns the date, which results from adding *number\_of\_days\_to\_add* to *any\_date*. If *any\_date* has a presentation that does not contain a century, the century is determined based on the Century > CCLookup map setting (when the Century > Switch = ON), or the current century (when the Century > Switch = OFF).

When the year exceeds four digits, the output will display hash characters (#) for the CCYY values because the field is defined to be only four digits in length.

## Examples

- You can use the **ADDDAYS** function to increment a date by a fixed number of days, such as to calculate a **DueDate** that is always 30 days after the **InvoiceDate**.

To produce a date in the output, such as a ship date, you might need to add a variable number of days to a date in the input. For example: **ADDDAYS** (PODate, LeadTime).

- **ADDDAYS** (InvoiceDate, 10)

Returns the date, which results from adding 10 days to the value of **InvoiceDate**.

- **ADDDAYS** (InvoiceDate, DaysTilDue)

Returns the date that results from adding the value of **DaysTilDue** to the **InvoiceDate** value.

- **ADDDAYS** (TODATETIME ("000101"), -1)

Returns 991231

- **ADDDAYS** (TODATETIME ("20000101"), 1)

Returns 20000102

- **ADDDAYS** (TODATETIME ("10/20/1996", "{MM/DD/CCYY}"), 5)

Returns 10/25/1996

In the examples containing **TODATETIME**, the text literal is first converted to a date, and then the specified number of days is added to that date.

## Related functions

- **ADDHOURS**
- **DATETONUMBER**
- **NUMBERTODATE**
- **TEXTTODATE**
- **TODATETIME**

## ADDHOURS

The **ADDHOURS** function returns a time value that is the result of adding a specified number of hours to a given time.

### Syntax:

**ADDHOURS** (single-datetime-item-expression, hours-expressed-as-signed-integer-expression)

### Meaning:

**ADDHOURS** (any\_datetime, number\_of\_hours\_to\_add)



**Returns:**

A single datetime item

The ADDHOURS function returns a single datetime item that is advanced from the original value of the single-datetime-item-expression by the specified number of hours-expressed-as-signed-integer-expression.

**Examples**

- `ADDHOURS(TODATETIME("Dec 31, 1999 23:59:00"), 2)`

Returns: Jan 1, 2000 01:59:00

- `ADDHOURS(TODATETIME("Dec 31, 1999 23:59:00"), -25)`

Returns: Dec 30, 1999 22:59:00

- `ADDHOURS(TODATETIME("23:59:00"), 2)`

Returns: 01:59:00

If either argument is invalid, the result is "none". If the second argument is "none", the result is the first argument. If the first argument contains only a time portion, date changes are ignored. If the first argument contains only a date portion, the time portion 00:00:00 will be used in the calculation. If both arguments are "none", the result is "none".

**Related functions**

- ADDDAYS
- DATETONUMBER
- NUMBERTODATE
- TEXTTODATE
- TODATETIME

**ADMINUTES**

The ADMINUTES function returns a time value that is the result of adding a specified number of minutes to a given time.

**Syntax:**

`ADMINUTES (single-datetime-item-expression, minutes-expressed-as-signed-integer-expression)`

**Meaning:**

`ADMINUTES (any_datetime, number_of_minutes_to_add)`

**Returns:**

A single datetime item

The ADMINUTES function returns a single datetime item, advanced from the original value of the single-datetime-item-expression by the specified number of minutes-expressed-as-signed-integer-expression.

## Examples

- `ADMINUTES(TODATETIME("Dec 31, 1999 23:59:00"), 2)`

Returns: Jan 1, 2000 00:01:00

- `ADMINUTES(TODATETIME("Dec 31, 1999 23:59:00"), -25)`

Returns: Dec 31, 1999 23:34:00

- `ADMINUTES(TODATETIME("23:59:00"), 2)`

Returns: 00:01:00

If either argument is invalid, the result is "none". If the second argument is "none", the result is the first argument. If the first argument contains only a time portion, date changes are ignored. If the first argument contains only a date portion, the time portion 00:00:00 will be used in the calculation. If both arguments are "none", the result is "none".

## Related Functions

- DATETONUMBER
- NUMBERTODATE
- TEXTTODATE
- TODATETIME

## CURRENTDATE

You can use CURRENTDATE when you need the current date as a transaction processing date, an order received date, or other date that reflects when the data was mapped.

When you need to parse the system date, use CURRENTDATE with the TEXT or FROMDATETIME functions. The function returns the current system date.

### Syntax:

`CURRENTDATE ( )`

### Meaning:

`CURRENTDATE ( )`

### Returns:

A single date

CURRENTDATE has no arguments but it does require parentheses.

If being assigned to a date/time output item, the current date is returned in the format specified by that output item. Otherwise, the system date is returned in an MM/DD/YY presentation.

## Examples

- `StartDate = CURRENTDATE ( )`

In this example, **StartDate** is assigned the value of the current **date**. In this example, because `CURRENTDATE` is assigned to an output, it is automatically converted to the presentation of **StartDate**.

If `CURRENTDATE` evaluates to 06/24/37 and **StartDate** has a YYMMDD presentation, the result is 370624.

- `FROMDATETIME ( CURRENTDATE ( ) , "DD.MON.CCY" )`

In this example, the current date is returned in DD.MON.CCY format. If today's date is January 5, 1999, the date returned would be 05.JAN.1999.

## Related functions

- `CURRENTDATETIME`
- `CURRENTTIME`
- `FROMDATETIME`
- `DATETOTEXT`

## CURRENTDATETIME

`CURRENTDATETIME` returns the current system date and time. You can use this function when you need to map the current system date and time to an item that includes both a date and time portion.

### Syntax:

```
CURRENTDATETIME ( [ single-text-expression ] )
```

### Meaning:

```
CURRENTDATETIME ( [ date_time_format_string ] )
```

### Returns:

A single date-time

`CURRENTDATETIME` has no arguments but it does require parentheses.

The `CURRENTDATETIME` function returns the system date and time in the format specified by *date\_time\_format\_string* or with a CCYYMMDDHHMMSS presentation if no *date\_time\_format\_string* is provided.

The *date\_time\_format\_string* must conform to the date/time format strings as described in "Format strings".



**Note:** The output format for the native schema XML date/time field is always:

```
{CCYY-MM-DD}T{HH24:MM:SS.3-3+/-ZZ:ZZ}
```



To specify a different output format, use Xerces XML instead.

## Examples

- `StartDateTime = CURRENTDATETIME ( )`

In this example, **StartDateTime** is assigned the value of the current date and time. Because `CURRENTDATETIME` is assigned to an output, it is automatically converted to the presentation of **StartDateTime**.

If `CURRENTDATETIME` evaluates to 3:04pm on 6/24/1999 and **StartDateTime** has a `YYMMDDHH12MM` presentation, the result is 9906240304.

- `CURRENTDATETIME ( "{MM.DD.CCY HH24:MM}" )`

In this example, the current date is returned in `MM.DD.CCY HH24:MM` format. If it is currently 4:12 pm on January 5, 1999, the date returned would be 01.05.1999 16:12.

## Related Functions

- `CURRENTDATE`
- `CURRENTTIME`
- `FROMDATETIME`
- `DATETOTEXT`

## CURRENTTIME

The `CURRENTTIME` function returns the current system time.

You can use `CURRENTTIME` when you need the system time as a transaction processing time, an order-received time, or a time that reflects when the data was mapped.

### Syntax:

```
CURRENTTIME ( )
```

### Meaning:

```
CURRENTTIME ( )
```

### Returns:

A single time

The `CURRENTTIME` function returns the system time. If assigned to a date-time output item, the current time is returned in the format specified by that output item. Otherwise, the system time is returned in `HH:MM:SS` presentation.



**Note:** CURRENTTIME has no arguments but it does require parentheses.

## Examples

- End Time = CURRENTTIME ( )

In this example, **End Time** is assigned the current time. Because CURRENTTIME is assigned to an output, it is automatically converted to the presentation of **End Time**.

If CURRENTTIME evaluates to 10:15:02 and **End Time** has an HH12:MM presentation, the result is 1015.

- FROMDATETIME ( CURRENTTIME ( ) , "{HH24MMSS}" )

In this example, the current time is returned in HH24MMSS format. If the current time is 10:15:02 pm, the result is 221502.

## Related functions

- CURRENTDATETIME
- FROMDATETIME
- DATETOTEXT

## DATETONUMBER

Use DATETONUMBER to perform arithmetic on dates.

The DATETONUMBER function returns an integer that results from counting the number of days from December 31, 1864, to the specified date.

### Syntax:

DATETONUMBER (single-date-expression)

### Meaning:

DATETONUMBER (date\_to\_convert)

### Returns:

A single integer

The DATETONUMBER function converts a date to an integer. The resulting integer represents the number of days since December 31, 1864, where using DATETONUMBER with a date of January 1, 1865 returns the integer value 1.

If the input argument is in error, the function returns the value "none".

If the date format specified by the input argument does not include century, the century is determined based on the **Century** map setting using the **CCLookup** parameter or the current century.

## Examples

- `DaysBetween = DATETONUMBER ( StopDate ) - DATETONUMBER ( StartDate )`

This expression could be used in a map rule to produce a value for **DaysBetween**. It converts the **StopDate** and the **StartDate** to integers, then subtracts the resulting two integers, and returns the result as **DaysBetween**.

## Related functions

- ADDBAYS
- NUMBERTODATE

## DATETOTEXT

The DATETOTEXT function converts a date object or expression to a text item.

### Syntax:

`DATETOTEXT (single-date-expression)`

### Meaning:

`DATETOTEXT (date_to_convert)`

### Returns:

A single text item

If *date\_to\_convert* is a date object name, this returns the date as a text item formatted according to the presentation of the date object.

If *date\_to\_convert* is a date expression produced by a function, this returns the date as a text item formatted according to the presentation of the output argument of that function.

## Examples

- `DATETOTEXT ( ShipDate )`

In this example, **ShipDate** is converted from a date to text. If **ShipDate** has a CCYYMMDD presentation, the resulting text item will have that presentation, as well.

- `DATETOTEXT ( CURRENTDATETIME ( "{MM/DD/CCYY}" ) )`

In this example, CURRENTDATETIME evaluates and returns a date in MM/DD/CCYY format. Then DATETOTEXT evaluates and returns a text string that is that date in MM/DD/CCYY format.

For example, use DATETOTEXT, to do text concatenation. The FROMDATETIME function provides greater flexibility in specifying the format of the resulting text item.

## Related Functions

- FROMDATETIME
- NUMBERTOTEXT
- TEXT
- TEXTTODATE
- TEXTTONUMBER
- TEXTTOTIME
- TIMETOTEXT
- TODATETIME

## FROMDATETIME

The FROMDATETIME function converts a date/time item to a text string of a specified format. For example, you can use FROMDATETIME to convert a date-time item into a string for parsing or concatenation. You can also use this function to access the individual parts of a date or time, such as the month, day, year, and so forth.

### Syntax:

```
FROMDATETIME (single-character-date-time-item
[ , single-text-expression ] )
```

### Meaning:

```
FROMDATETIME (date_time [ , date_time_format_string ] )
```

### Returns:

A single character text item

FROMDATETIME returns the date/time specified by *date\_time* as a text item with the format specified by *date\_time\_format\_string*. If *date\_time\_format\_string* is not specified, *date\_time* will be returned in the same format as the *date\_time*.

The *date\_time\_format\_string* must conform to the date/time format strings as described in the "Format strings" topic.

## Examples

- HeaderLine="Today is" + FROMDATETIME (CURRENTDATE ()), "{MON DD, CCYY}")

If the current system date is March 3, 1999, this rule evaluates to Today is Mar 03, 1999.

- FROMDATETIME ( TransactionTimeStamp Column::TransHistory )

If the value of **TransactionTimeStamp Column** is 10:14 A.M. on February 3, 2000 and its format is defined as "{CCYYMMDDHH24MM}", this rule evaluates to 200002031014.

- EXTRACT (Expense:Report, FROMDATETIME (Date:Expense:Report, "{MM}") = "03")

In this example, the FROMDATETIME function is used in the condition expression of the EXTRACT function to identify all expenses in the month of March.

## Related functions

- CURRENTDATE
- CURRENTDATETIME
- CURRENTTIME
- DATETOTEXT
- TODATETIME

## MAX

The MAX function returns the maximum value from a series of number, date, time, or text values.

### Syntax:

MAX (series-item-expression)

### Meaning:

MAX (series\_of\_which\_to\_find\_max)

### Returns:

A single number

The result is the maximum value in the input argument series: number, text, or date/time.

## Examples

- MAX ( UnitPrice:Input )

If the values for **UnitPrice** are {20, 10, 100}, MAX returns 100.

- MAX(EXTRACT( DueDate:Book:Library, CheckedOut:Book:Library = "Y"))

Returns the maximum (latest) **DueDate** for a book that is checked out from the library.

## Related functions

- MIN

## MIN

Use MIN when you need the minimum value from a series of number, date, time, or text values.

The MIN function returns the minimum value from a series.

### Syntax:

MIN (series-item-expression)

### Meaning:

MIN (series\_of\_which\_to\_find\_min)



**Returns:**

A single number

The result is the minimum value of the input series: number, text, or date/time.

**Examples**

- MIN (UnitPrice:Input)

If the values for **UnitPrice** are {20,10,100}, MIN returns 10.

- MIN (StartTime::Schedule)

Returns the minimum (earliest) **StartTime** in **Schedule**.

**Related functions**

- MAX

**NUMBERTODATE**

You can use the NUMBERTODATE function to perform a calculation on a date.

NUMBERTODATE converts an integer to a date, where the integer is the number of days from December 31, 1864, to the specified date. Only positive, non-zero values can be specified as valid parameters.

**Syntax:**

NUMBERTODATE (single-integer-expression)

**Meaning:**

NUMBERTODATE (integer\_to\_convert)

**Returns:**

A single date

After converting an integer to a date, if the result is being assigned to a date object, the resulting date is in the presentation for that output. If the resulting date is not being assigned to an object, it has a CCYYMMDD presentation.

When the year exceeds four digits, the output will display hash characters (#) for the CCYY values because the field is defined to be only four digits in length.

**Examples**

- NUMBERTODATE (StartDate)

This example converts the **StartDate** value from an integer to a date that is in CCYYMMDD format.

## Related functions

- ADDBAYS
- DATETONUMBER
- FROMDATETIME
- TODATETIME

## TEXTTODATE

The TEXTTODATE function converts text item from CCYYMMDD or YYMMDD format to a date.

You can use the TEXTTODATE to convert a text item to a date, however, the TODATETIME function provides greater flexibility for specifying the resulting date-time format .

### Syntax:

TEXTTODATE ( single-text-expression )

### Meaning:

TEXTTODATE ( text\_to\_convert\_to\_date )

### Returns:

A single date

The *text\_to\_convert\_to\_date* must be in either CCYYMMDD or YYMMDD format. If the *text\_to\_convert\_to\_date* is in error (for example, it is not a valid date), the result is "none".

If *text\_to\_convert\_to\_date* is in YYMMDD format and is being assigned to a date format that includes a century, the century is determined based on the century run option setting using the **CCLookup** parameter or the current century.

## Examples

- `TEXTTODATE ( "990114" )`

Returns a single date item with the value of January 14 in the year 99

- `TEXTTODATE ( OrderDate )`

Returns **OrderDate** as a single date item.

## Related functions

- DATETOTEXT
- TEXTTOTIME
- NUMBERTO-  
TEXT
- TIMETOTEXT

- TEXTTONUMBER
- TO-DATETIME

## TEXTTOTIME

Use TEXTTOTIME when you want to convert an object defined as text that is in HHMM or HHMMSS presentation, to an item defined as time. For greater flexibility, use the TODATETIME function for specifying the format of the text item that is to be converted to a date/time.

### Syntax:

TEXTTOTIME ( single-text-expression )

### Meaning:

TEXTTOTIME ( text\_to\_convert\_to\_time )

### Returns:

A single time

The *text\_to\_convert\_to\_time* must be in HHMM or HHMMSS presentation. HH is a two-digit hour in a 24-hour format. If the result is being assigned to a time object, the resulting time looks like the output object. Otherwise, the resulting time looks like the input argument. If the input argument is in error (for example, it is not a valid time), the result is "none".

## Examples

- TEXTTOTIME ( CallTime )

Returns **CallTime** as a time item

## Related functions

- DATETOTEXT
- TEXTTONUMBER
- FROMDATETIME
- TIMETOTEXT
- NUMBERTO-TEXT
- TODATETIME

- TEXTTODATE

## TIMETOTEXT

You can use the TIMETOTEXT function to perform text concatenation. For greater flexibility, use the FROMDATETIME function for specifying the format of the resulting text item.

TIMETOTEX converts a time object or expression to a text item.

### Syntax:

TIMETOTEXT ( single-time-expression )

### Meaning:

TIMETOTEXT ( time\_to\_convert\_to\_text )

### Returns:

A single text item

If *time\_to\_convert\_to\_text* is a time object name, this returns the time as a text item formatted according to the presentation of the input date object.

If *time\_to\_convert\_to\_text* is a time expression produced by a function, this returns the time as a text item formatted according to the presentation of the output argument of that function.

## Examples

- TIMETOTEXT ( LeadTime )

In this example, **LeadTime** is converted from a time to text. If **LeadTime** has an HH:MM presentation, the resulting text item will be of that presentation.

- TIMETOTEXT ( CURRENTDATETIME ( "{HH:MM:SS}" ) )

Here, CURRENTDATETIME evaluates and returns a time in HH:MM:SS format. Then, TIMETOTEXT evaluates and returns a text string that is that time in HH:MM:SS format.

## Related functions

- DATETOTEXT
- FROMDATETIME
- NUMBERTOTEXT
- TEXTTODATE
- TEXTTONUMBER
- TEXTTOTIME
- TODATETIME

## TODATETIME

The TODATETIME function converts a text string of a specified format to a date-time item.

### Syntax:

```
TODATETIME ( single-character-text-expression
            [ , single-text-expression ] )
```

### Meaning:

```
TODATETIME ( text_to_convert [ , date_time_format_string ] )
```

### Returns:

A single character date item

TODATETIME returns the date-time that corresponds to the value specified by *text\_to\_convert*, which is in the format specified by *date\_time\_format\_string*. If *date\_time\_format\_string* is not specified, it will be assumed that *text\_to\_convert* is in [CCYYMMDDHH24MMSS] format.

The *date\_time\_format\_string* must conform to the date-time format strings as described in "Format strings".

## Examples

- TODATETIME ( "05/14/1999@10:14pm" , "{MM/DD/CCYY}@{HH12:MMAM/PM}" )

In this example, a text string containing a date and time is converted to a date-time item.

- RptDate = TODATETIME ( RIGHT ( GETRESOURCE() , 8 ) , "CCYYMMDD" )

Assume that you receive a file that contains historical data. The name of the file identifies the date of the historical data. For example, a filename of **19960424** indicates that the data was produced on April 24, 1996. To map this date to **RptDate**, the TODATETIME function could be used with the RIGHT and GETRESOURCE functions.

## Related functions

- CURRENTDATE
- CURRENTDATETIME
- CURRENTTIME
- TEXTTODATE
- TEXTTOTIME

## Error handling functions

### CONTAINSERRORS

The CONTAINSERRORS function tests a valid object to see whether it contains any objects in error.

**Syntax:**

CONTAINSEERRORS (single-object-expression)

**Meaning:**

CONTAINSEERRORS (object\_to\_test)

**Returns:**

True or false

The CONTAINSEERRORS function returns "true" if any object contained in *object\_to\_test* is in error; it returns "false" if the *object\_to\_test* is completely valid.

The input object, itself, is a *valid* input object. This function does *not* evaluate for invalid objects. Therefore, if you have map rule:

CONTAINSEERRORS ( Invoice:InputFile )

and **Invoice**[1] is valid, **Invoice**[2] is invalid, and **Invoice**[3] is valid, then CONTAINSEERRORS evaluates only twice-once for each *valid* instance of **Invoice**.

**Examples**

- Msg (s) = IF ( CONTAINSEERRORS ( Msg:MailBag ) & Type:Msg:MailBag = "PRIORITY" , Msg:MailBag , "none" )

In this example, if **Msg:MailBag** contains any object in error and **Type:Msg:MailBag** has a value of "PRIORITY", **Msg:MailBag** is mapped; otherwise, "none" is returned for this occurrence of **Msg**. This map rule returns all valid messages (**Msg**) that contain errors with a **Type** of "PRIORITY".

**Related functions**

- ISERROR
- REFORMAT

**FAIL**

You can use the FAIL function to abort a map based on map or application specific logic.

**Syntax:**

FAIL (single-text-expression)

**Meaning:**

FAIL (message\_to\_return)

**Returns:**

"None"

The FAIL function returns "none" to the output to which the function is assigned, aborts the map, and returns *message\_to\_return* as the map completion error message included in the execution audit. The map return code will be "30", indicating that the map failed through the FAIL function.

## Examples

- `AcctID = EITHER (LOOKUP (CustomerID::Xref , MyKey::Xref = ID::Input ) , FAIL ( "Unknown Customer (" + ID::Input + "). Processing Terminated." ) )`

In this example, the FAIL function is being used in conjunction with the EITHER function to conditionally fail the map if a record in the customer cross-reference file does not exist for a given **CustomerID**.

For example, the **ID** in **Input** is ABC123. If the LOOKUP succeeds, the **CustomerID** result is assigned to **AcctID** and the map continues.

If the LOOKUP fails, **AcctID** is assigned a value of "none", the map aborts, and the message Unknown Customer (ABC123). Processing Terminated.is written to the execution audit log.

- `Message = VALID (RUN ("Map1Msg.mmc", "-AE -OMMSMQ1B `QN .\aqueue -CID 2001" ) , FAIL ("Failure on RUN (" + TEXT (LASTERRORCODE ( ) ) + "):" + LASTERRORMSG ( ) ) )`

In this example, the FAIL function is being used in conjunction with the VALID, LASTERRORCODE, and LASTERRORMSG functions to fail (abort) the map if the map executed by the RUN function (**Map1Msg.mmc**) fails. In this example, the map fails and returns the error code and error message reported by the RUN function using the LASTERRORCODE and LASTERRORMSG functions.

If **Map1Msg** fails because one or more of its inputs was invalid, **Message** is assigned a value of "none". The map aborts and the following message is reported in the execution audit log:

"Failure on RUN (8): One or more inputs was invalid."

## Related functions

- LASTERRORCODE
- LASTERRORMSG
- QUIT
- VALID

## GETXMLERRORMSG

Use this function on an output card to return an XML validation error message when a map runs against invalid XML input.

Create the input card with either a native XML schema or an XML-based type tree, and specify the Restart attribute. On the output card, use the GETXMLERRORMSG function, passing the number of the input card as the parameter. When the map runs and the input contains invalid XML data, the output card returns a standard XML error message. For example:

```
Error (-1), "XMLParser: Input XML data is invalid."
SAXParseException, Error [line: 162 column: 36]
Datatype error: Type:SchemaDateTimeException,
Message:Incomplete Date ! '2006-09-' .
```

See the Type Designer documentation or the Map Designer documentation for details about the Restart attribute.

### Syntax

GETXMLERRORMSG (single-number-expression)

### Meaning

GETXMLERRORMSG (input\_card\_number)

### Returns

A text string

### Related functions

- VALIDATE

## ISERROR

The ISERROR function tests an object to see if it is in error

You can use ISERROR to output your data in exactly the same order as it occurs in your input, both the valid data and the data in error. You can also use ISERROR to produce error messages for bad data in the same file in which you map your good data.

### Syntax:

ISERROR (single-object-name)

### Meaning:

ISERROR (object\_to\_test)

### Returns:

"True" or "false"

ISERROR returns "true" when *object\_to\_test* is in error and returns "false" when *object\_to\_test* is completely valid.

### Examples

- InfoRec (s) = IF ( ISERROR ( Record:SomeFile ), "Bad -> " + REJECT ( Record:SomeFile ), "Ok -> " + TEXT ( Record:SomeFile ) )

In this example, ISERROR is used to produce a report for *all Record* objects in **SomeFile**. If the record is in error, the **InfoRec** will have the text Bad -> followed by the data from the input **Record**. If the record is valid, the **InfoRec** will have the text Ok -> followed by the data from the input **Record**, such as:



```
Ok --> SZ-68839,486 Upgrade Microprocessor,186.86,100,W200
Bad --> MK-19309,,369.43,417,W100
Ok --> KL-20349,PCMCIA Network Adaptor,174.82,29,N300
Ok --> WP-37679,AC Adaptor,39.48,245,E100
Bad --> IL-39890,8MB Memory PCMCIA,390.48,0,S100
```

## Related functions

- CONTAINSERRORS

## ONERROR

Use ONERROR in component rules to add user-defined error messages to the data section of the audit log.

The ONERROR function adds a user-defined error message to the data section of the audit log. This function is relevant only in Type Designer Component rules.

### Syntax:

ONERROR (single-condition-expression , single-text-expression)

### Meaning:

ONERROR (condition\_to\_evaluate , message\_to\_display)

### Returns:

"True" or "false"

If *condition\_to\_evaluate* evaluates to "true", the function returns "true".

If *condition\_to\_evaluate* evaluates to "false", the function returns "false".

If data audit is enabled and the object to which the component rule applies will result in a failed component rule message (status E09), *message\_to\_display* is written to the data audit section of the audit log with an entry type of U, representing user-defined.

If the failed component rule message (E09) appears in the data section of the audit log, one or more user-defined error messages can also be included in the data section of the audit log. If a component rule has one ONERROR function, one user-defined error message or none is included in the audit log. If a component rule has two ONERROR functions, two user-defined error messages at most are included in the audit log, and so on.

## Examples

- Here is a component rule without ONERROR:

Claim Date Field > Accident Date Field &

Claim Date Field < ADDDDAYS (Accident Date Field, 365)

If the component rule fails, the data section of the audit log shows the following information:

```
<DataLog>
<input card="1">
  <object ... status="E07">InsuranceClaim</object>
  <object ... status="E09">Claim Date Field</object>
    <Text>980725</Text>
  <object ... status="E07">InsuranceClaim</object>
  <object ... status="E09">Claim Date Field</object>
    <Text>990526</Text>
</input>
</DataLog>
```

Status code E09 for the **Claim Date Field** indicates that the data for that object failed its component rule, but does not provide any further detail.

- Using ONERROR, one or more error messages can be added to the data section of the audit log to provide more information as to how or why the component rule failed. For example,

```
ONERROR ( Claim Date Field > Accident Date Field , "Claim date before accident." ) &
```

```
ONERROR ( Claim Date Field < ADDDDAYS ( Accident Date Field , 365 ) , "Claim is more than one year old." )
```

If the component rule fails and ONERROR is used, the data section of the audit log can show the following messages:

```
<DataLog>
<input card="1">
  <object ... status="E07">InsuranceClaim</object>
  <object ... status="E09">Claim Date Field</object>
    <Text>980725</Text>
    <User>Claim date before accident.</User>
  <object ... status="E07">InsuranceClaim</object>
  <object ... status="E09">Claim Date Field</object>
    <Text>990526</Text>
    <User>Claim is more than one year old.</User>
</input>
</DataLog>
```

## REJECT

The REJECT function returns the content of an object in error as a text item. Use REJECT in conjunction with the restart attribute. You can use the REJECT function in map rules only, not in component rules.

See the schema designer and map designer documentation for information about the restart attribute.

### Syntax:

```
REJECT (series-object-expression)
```

### Meaning:

```
REJECT (series_to_look_for_bad_objects)
```

**Returns:**

A series text item

REJECT evaluates to a series of text items consisting of all the input series members in error.

**Examples**

- REJECT (Record:File)

This example extracts all **Records** that are in error.

- IF (COUNT (REJECT (Msg IN Batch))) = 0, "OK", "ERROR")

In this example, the total number of invalid (rejected) Msg objects is counted. If the total number of invalid Msg objects is equal to zero, it indicates that there were no invalid Msg objects counted and a message of OK results. Otherwise, a message of ERROR results.

**Related functions**

- CONTAINSERRORS
- ISERROR
- VALID

**QUIT**

You can use the QUIT function to stop a map at a specific point during map execution. The QUIT function logs a successful return code and a custom message in the map audit log.

**Syntax:**

QUIT (single-text-expression)

**Meaning:**

QUIT (message\_to\_return)

**Returns:**

"None"

The QUIT function returns "none" to the output, stops the map, and logs map return code 41 and the message `QUIT function terminated map successfully: message_to_return` in the map audit log.

**Related functions**

- FAIL
- LASTERRORCODE
- LASTERRORMSG
- VALID

## VALID

You can use the VALID function to perform conditional processing based on whether an external interface function executes successfully.

VALID returns the result of the first argument if it is valid; otherwise, returns the second argument.

### Syntax:

```
VALID ( single-text-expressions , single-general-expression )
```

### Meaning:

```
VALID ( function_that_can_fail , return_value_if_function_fails )
```

### Returns:

A single text expression

VALID returns the result of the evaluation of *function\_that\_can\_fail* if it is valid. If the function fails, VALID returns *return\_value\_if\_function\_fails*.

The following functions can fail:

- DBLOOKUP
- GET
- DBQUERY
- PUT
- DDEQUERY
- RUN
- EXIT

## Examples

```
SomeObject = VALID ( RUN ( "mymap.mmc" , "-OFl mydata.txt" ) , FAIL ( "My RUN failed!" ) )
```

If the RUN function returns an error return code, the VALID functions returns `none`, the map aborts, and the message `My RUN failed!` is reported under Execution Summary in the execution audit log.

## Related functions

- DBLOOKUP
- DBQUERY
- DDEQUERY

## External interface functions

### DBLOOKUP

The DBLOOKUP function executes an SQL statement against a database. The SQL statement can be any permitted by your database management system or ODBC driver.

When the DBLOOKUP function is used in a map, the default **OnSuccess** action is adapter specific. The default **OnFailure** action is to rollback any changes made during map processing. The default **Scope** will be integral unless the map is defined to run in bursts (which is the case when one or more inputs have the **FetchAs** property set to **Burst**).

There are two ways to specify arguments for DBLOOKUP.

You can use DBLOOKUP to execute an SQL statement when you want to execute a SELECT statement to retrieve a specific column value in a large table in a database using the value of another input, rather than defining the entire table as an input card and using the LOOKUP, SEARCHDOWN, or SEARCHUP functions.

You can use DBLOOKUP to execute an SQL statement when you want to execute a SELECT statement to retrieve a specific column value from a table or database that might vary based on a parameter file. Using Meaning 2 of the DBLOOKUP function allows these parameters to be dynamically specified at run time.

#### Syntax:

```
DBLOOKUP ( single-text-expression , single-text-expression , [ single-text-literal ] )
```

#### Meaning:

1. DBLOOKUP ( SQL\_statement , mdq\_filename , database\_name )
2. DBLOOKUP ( SQL\_statement , parameters )

#### Returns:

A single text item

The DBLOOKUP function returns the results of the query in the same format as a query specified for a map input card, except that it does not include the last carriage return/linefeed. Because this information is removed, it is easier to make use of a single value extracted from a database.

### Arguments for meaning 1

```
DBLOOKUP ( SQL_statement , mdq_filename , database_name )
```

#### • SQL\_statement

The first argument is an SQL statement as a text string. This can be any valid SQL statement permitted by your database management system and supported by your database-specific driver. In addition to a fixed SQL

statement, this argument can be a concatenation of text literals and data objects, enabling the concatenation of data values into your SQL statement.

- **mdq\_filename**

The second argument is the name of a database query file (**.mdq**) produced by the Database Interface Designer. It contains the definition of the database that the SQL statement is to be executed against. If the **.mdq** file is in a directory other than the directory of the map, the path must be specified.



**Note:** The **.mdq** file is accessed at map build time and is not needed at run time.

- **database\_name**

The third argument is the name of a database in the database query file (**.mdq**) as defined in the Database Interface Designer.

If used in this way, both the **.mdq** filename and database name must be literals.

## Arguments for meaning 2

DBLOOKUP ( SQL\_statement , parameters )

- **SQL\_statement**

The first argument is an SQL statement as a text string. This can be any valid SQL statement permitted by your database management system and supported by your database-specific driver. In addition to a fixed SQL statement, this argument can be a concatenation of text literals and data objects, enabling the concatenation of data values into your SQL statement.

- **parameters**

The second argument is a set of parameters, either:

- -MDQ *mdqfilename* -DBNAME *dbname*
- or-
- -DBTYPE *database\_type* [*database specific parameters*]

The keyword -MDQ is followed by the name of the database query file (**.mdq**) produced by the Database Interface Designer. This **.mdq** file contains the definition of the database. If the **.mdq** file is in a directory other than the directory of the map, the path must be specified. The **.mdq** filename is followed by the keyword -DBNAME and the database name as specified in the Database Interface Designer.

Using this syntax, the **.mdq** file is accessed at run time and must be present.

The keyword `-DBTYPE` is followed by a keyword specifying the database type (for example, ODBC or ORACLE) followed, optionally, by database-specific parameters.

This syntax does not use an `.mdq` file, because the database-specific parameters provide the information required to connect to the database. See the Resource Adapters documentation for detailed information about the database-specific parameters that can be specified.

When used with Meaning 2, `DBLOOKUP` must conform to these rules:

- All keywords (for example, `-DBTYPE`) can be upper or lowercase, but not mixed.
- A space is required between the keyword and its value (for example, `-DBTYPE ODBC`).
- The order of the keywords is not important.

All database-specific parameters are optional.

## Examples

Assume that you have a table named "PARTS" that contains the following data:

	<b>PART_NUMBER</b>	<b>PART_NAME</b>
1		1/4" x 3" Bolt
2		1/4" x 4" Bolt

Assume that this database has been defined using the Database Interface Designer in a file named `mytest.mdq` and that the name of the database, as specified in the `.mdq` file, is `PartsDB`.

- `DBLOOKUP ("SELECT PART_NAME from PARTS where PART_NUMBER =1", "mytest.mdq", "PartsDB")`

Returns: 1/4" x 3" Bolt

Using Meaning 2, you can specify the `DBLOOKUP` this way:

- `DBLOOKUP("SELECT PART_NAME from PARTS where PART_NUMBER =1", "-MDQ mytest.mdq -DBNAME PartsDB")`

where both the `.mdq` file name and database name is specified.

Using Meaning 2, you can also specify the database type and the appropriate database-specific parameters:

- `DBLOOKUP("SELECT PART_NAME from PARTS where PART_NUMBER =1", "-DBTYPE ORACLE -CONNECT MyDB -USER janes ")`

## Related functions

- DBQUERY
- EXTRACT
- FAIL
- LASTERRORCODE
- LASTERRORMSG
- LOOKUP
- SEARCHDOWN
- SEARCHUP
- VALID

For more examples using the DBLOOKUP function, see the Database Interface Designer documentation.

## DBQUERY

The DBQUERY function executes an SQL statement against a database. The SQL statement can be any permitted by your database management system or ODBC driver.

When the DBQUERY function is used in a map, the default **OnSuccess** action is adapter specific. The default **OnFailure** action is to rollback any changes made during map processing. The default **Scope** will be integral unless the map is defined to run in bursts (which is the case when one or more inputs have the **FetchAs** property set to **Burst**).

There are two ways to specify the arguments for DBQUERY. You can use DBQUERY [*Meaning 1*] to execute an SQL statement when you want to look up information in a database using a parameterized query that is based on another value in your data. If your SQL statement is a SELECT statement, the DBQUERY function might be used in conjunction with the RUN function to issue dynamic SELECT statements whose results can be used as input to another map.

You can also use the DBQUERY function [*Meaning 2*] to execute an SQL statement when the database, table, or other database parameters might vary; perhaps being supplied by a parameter file.

### Syntax:

```
DBQUERY (single-text-expression , single-text-expression ,
        [ single-text-literal ] )
```

### Meaning:

1. DBQUERY (SQL\_statement , mdq\_filename , database\_name)
2. DBQUERY ( SQL\_statement , parameters )

### Returns:

A single text item



If your SQL statement is a SELECT statement, the results of the query in the same format as a query specified as a map input card, including row delimiters and terminators, and so on.

If your SQL statement is anything other than a SELECT statement, "none".

## Arguments for meaning 1

DBQUERY (SQL\_statement , mdq\_filename , database\_name)

- **SQL\_statement**

The first argument is an SQL statement as a text string. This can be any valid SQL statement that is permitted by your database management system and supported by your database-specific driver. In addition to a fixed SQL statement, this argument can be a concatenation of text literals and data objects, enabling the concatenation of data values into your SQL statement.

- **mdq\_filename**

The second argument is the name of a database query file (**.mdq**) produced by the Database Interface Designer. It contains the definition of the database that the SQL statement is to be executed against. If the **.mdq** file is in a directory other than the directory of the map, the path must be specified.



**Note:** The **.mdq** file is accessed at map build time and is not needed at run time.

- **database\_name**

The third argument is the name of a database in the database query file (**.mdq**) as defined in the Database Interface Designer.

If used in this way, both the **.mdq** filename and database name must be literals.

## Arguments for meaning 2

DBQUERY ( SQL\_statement , parameters )

- The first argument is an SQL statement as a text string. This can be any valid SQL statement that is permitted by your database management system and supported by your database-specific driver. In addition to a fixed SQL statement, this argument can be a concatenation of text literals and data objects, enabling the concatenation of data values into your SQL statement.
- The second argument is a set of parameters, either:
  - -MDQ mdqfilename -DBNAME dbname
  - or-
  - -DBTYPE database\_type [database specific parameters]

The keyword -MDQ is followed by the name of the database query file (**.mdq**) produced by the Database Interface Designer. This **.mdq** file contains the definition of the database. If the **.mdq** file

is in a directory other than the directory of the map, the path must be specified. The **.mdq** filename is followed by the keyword **-DBNAME** and the database name as specified in the Database Interface Designer.



**Note:** Using this syntax, the **.mdq** file is accessed at run time and must be present.

The keyword **-DBTYPE** is followed by a keyword specifying the database type (for example, ODBC or ORACLE) followed, optionally, by database-specific parameters.



**Note:** This syntax does not use an **.mdq** file, because the database-specific parameters provide the information required to connect to the database. See the appropriate database adapter documentation for detailed information about database-specific parameters.

When used with Meaning 2, **DBQUERY** must conform to these rules:

- All keywords (for example, **-DBTYPE**) can be upper or lower case, but not mixed.
- A space is required between the keyword and its value (for example, **-DBTYPE ODBC**).
- The order of the keywords is not important.

All database-specific parameters are optional.

## Examples

Assume that you have a table named "PARTS" that contains the following data:

PART_NUMBER	PART_NAME
1	1/4" x 3" Bolt
2	1/4" x 4" Bolt

Also assume that this database has been defined using the Database Interface Designer in a file named **mytest.mdq** and that the name of the database, as specified in the **.mdq** file, is **PartsDB**.

```
DBQUERY ( "SELECT * from PARTS" , "mytest.mdq" , "PartsDB" )
```

Returns 1|1/4" x 3" Bolt<cr><lf>2|1/4" x 4" Bolt<cr><lf>

where <cr><lf> is a carriage return followed by a line feed.

Using Meaning 2, you can also specify the **DBQUERY** this way:

```
DBQUERY ( "SELECT * from PARTS" , "-MDQ mytest.mdq -DBNAME PartsDB" )
```

where both the **.mdq** file name and database name are specified.

Or, specify it this way, using Meaning 2 by specifying the database type and the appropriate database-specific parameters:

```
DBQUERY ( "SELECT * from PARTS" , "-DBTYPE ORACLE -CONNECT MyDB -USER janes" )
```

Assume that you have an input file containing one order record. To map that order to another proprietary format, you also have a parts table with pricing information for every part for every customer, a very large table. Rather than using the entire parts table as the input to your map, you might use the RUN function with a DBQUERY to dynamically select only those rows from the parts table corresponding to the customer in the order file, as follows:

```
RUN ( "MapOrder.MMC" ,
      "IE2" + DBQUERY ( "SELECT * FROM Parts WHERE CustID = "
      + CustomerNo:OrderRecord:OrderFile + " ORDER BY PartNo" ,
      "PartsDB.MDQ", "PartsDatabase" ) )
```

## Related functions

- DBLOOKUP
- EXTRACT
- FAIL
- LASTERRORCODE
- LASTERRORMSG
- LOOKUP
- SEARCHUP
- SEARCHDOWN
- VALID

## DDEQUERY

The DDEQUERY function allows you to interface to other Windows applications such as Trading Partner PC, Excel, and so forth, provided that certain criteria are met. For example, if you receive an Excel spreadsheet file, you must have the appropriate version of the Excel application installed (that is compatible with the file received) and the application must be open.

### Syntax:

```
DDEQUERY (single-text-expression , single-text-expression , single-text-expression)
```

### Meaning:

```
DDEQUERY (application_name , topic , text)
```

### Returns:

A single text item from an application

## Examples

- DDEQUERY ( "excel" , "[MKTPRICE.XLS]Sheet1" , "R8C1:R14C3" )

In this example, DDEQUERY is used to get data from an Excel spreadsheet. The third argument, **R8C1:R14C3**, specifies the location of the data in the spreadsheet. (In Excel, the 8<sup>th</sup> row, 1<sup>st</sup> column to the 14<sup>th</sup> row, 3<sup>rd</sup> column is A8:C14.) The content of this spreadsheet range is returned as a single text item.

This example assumes that the application, the map, and the spreadsheet all reside in the same directory. If they are not in the same directory you must add the path. For example:

```
DDEQUERY ( "excel" , "c:\spreadsheet[MKTPRICE.XLS]Sheet1" , "R8C1:R14C3" )
```

- DDEQUERY ("tpc","PartnerX","BGyourEDIode")

In this example, DDEQUERY is used as a request to Trading Partner PC.

## Related functions

- EXIT
- FAIL
- GET
- LASTERRORCODE
- LASTERRORMSG
- PUT
- RUN
- VALID

## EXIT

The EXIT function allows you to interface with a function in an external library or application.

You can use EXIT when you need information from an existing function in a library or a program or when you need to use a general function that is not available.

Depending on the execution operating system, there are two different methods for the EXIT function: 1) the library method and 2) the program method. The program method is not supported on Windows operating systems.

### Syntax:

```
EXIT (single-text-expression, single-text-expression,  
single-text-expression)
```

### Meaning:

1. EXIT (library\_name, function\_name, input\_to\_the\_function)
2. EXIT ( program\_name , command\_line\_arg1 , command\_line\_arg2 )

### Returns:

A single text item

## Meaning 1 - library method

At run time, the *function\_name* function will execute in the library specified by *library\_name* passing *input\_to\_the\_function* as a text string. The result of *function\_name* is returned as a text item by means of **lpep- > lpdataFromApp**.

Set **lpep- > nReturn** equal to 0 if the function is to succeed or set it equal to 1 to fail.

For detailed information about the requirements of the library function that is executed by the EXIT, see [Implementing a library EXIT function on page 835](#).

AIX operating system

On the AIX operating system, the shared library that contains *library\_function* must also contain an *entry\_point* function. The *entry\_point* function must be prototyped in the same manner as *library\_function*. The server will invoke the *entry\_point* function before invoking *library\_function*. The server passes the name of the *library\_function* to the *entry\_point* function by way of the *szFile* EXITPARAM structure member. The *entry\_point* function must then determine the address of the *library\_function* and pass this address back to the server by way of the *lpv* EXITPARAM structure member. The server will then use the contents of *lpv* as an address to invoke the *library\_function*.

The following example depicts a shared library called **mcshex.so** which contains the *entry\_point* function and the *library\_function*.

```
#include <stdio.h>
#include <string.h>
#include "runmerc.h"
void AIXEntry(LPEXITPARAM);
void bin2hex(LPEXITPARAM);
unsigned int i;
char *syms[] = { "bin2hex", "AIXEntry" };
void *adr[] = { (void *)bin2hex, (void *)AIXEntry};
void AIXEntry(LPEXITPARAM lpInputStruct)
{
    printf("AIXEntry called\n");
    for (i = 0; i < (sizeof(syms)) / (sizeof(char *)); i++)
        if ( !(strcmp(lpInputStruct->szFile, syms[i])) ) {
            lpInputStruct->lpv = adr[i];
            break;
        }
}
void bin2hex(LPEXITPARAM lpInputStruct)
{
    printf("bin2hex called\n");
    printf("argument to library function: %s\n",lpInputStruct->lpszCmdLine);
    lpInputStruct->nReturn = 0;
}
```

To build this shared library, run the following commands:

```
cc -c share1.c
```

```
cc -o mcshex.so share1.o -bE:shrsb.exp -bM:SRE -eAIXEntry
```

## Meaning 2 - program method

The program method of the EXIT function is *not* supported on Windows operating systems.

At execution time, the program specified by *program\_name* executes and passes the concatenation of *command\_line\_arg1* + " " + *command\_line\_arg2* as a text string.

Whatever is returned by *program\_name* to the standard output device is returned as text.

## Examples

- EXIT ( *program\_name* , *command\_line\_arg1* , *command\_line\_arg2* )

Returns a text string from the function or application that is executed. If the EXIT function is not available for a particular operating system, EXIT returns `none`.

- EXIT ( "mydll.dll" , "myfunction" , "12" )

This Windows library example passes the value 12 to **myfunction**, a function in **mydll.dll**. The value of the item returned depends on what **myfunction** does with the 12 passed to it.

- IF (EXIT ("mylib.sl" , "ckCust" , CustID Column:Row:DB) = "OK" , MapKnownCust ( Row:DB ) , MapUnknownCust ( Row:DB ) )

Similarly, this UNIX library example passes the value of **CustID** Field to **ckCust**, a function in a UNIX shared library called **mylib.sl**. If the value returned by **ckCust** is OK, a functional map is called to map **Row** for a known customer. Otherwise, another functional map is executed to map **Row** for an unknown customer.

- EXIT ( "pwd" , " " , " " )

This UNIX program example executes the UNIX print working directory (`pwd`) command to determine the current directory. The name of the current working directory is then returned as a text string. Notice that although the `pwd` command does not require additional command line arguments, *command\_line\_arg1* and *command\_line\_arg2* must be included as a space enclosed in double quotation marks (" ").

- TEXTTONUMBER ( ( EXIT ( "GetIncome" , Applicant:Form , "\*Mortgage" ) ) )

This program example passes the value of Applicant concatenated to the text literal `*Mortgage` to an application called `GetIncome`. The result is converted to a number.

## Related functions

- DDEQUERY
- FAIL
- GET
- LASTERRORCODE
- LASTERRORMSG
- PUT

- RUN
- VALID

## Implementing a library EXIT function

You can develop a function in a library to be executed from an EXIT function.

### EXIT function's library interface

Library functions are used within an EXIT function using information contained in the EXITPARAM structure. This method provides great flexibility for data passed to and from a map. For example, the map can pass binary data containing nulls, and there is no limitation on the length of the returned data. The method also allows functions to report additional information by providing a return code and error message.

### Using the EXITPARAM Structure

#### Function prototype

The function to be executed must be a function in a library with the following prototype:

```
void MyFunc(LPEXITPARAM lpep);
```

#### Definition of the EXITPARAM structure

The definition of the EXITPARAM structure is as follows:

```
struct tagExitParamStruct
{
    DWORD        dwSize;
    DWORD        dwToLen;
    DWORD        dwFromLen;
    DWORD        dwMapInstance;
    void FAR *   lpv;
    LPSTR        lpszCmdLine;
    BYTE HUGE *  lpDataToApp;
    BYTE HUGE *  lpDataFromApp;
    UINT         uRetryCount;
    UINT         uRetryInterval;
    BOOL         bRollback;
    BOOL         bCleanup;
    int          nReturn;
    char         szErrMsg[100];
    char         szFile[260];
    void FAR *   lpMapHandle;
    void FAR *   lpInternal;
    void FAR *   lpCmdStruct;
    void FAR *   lpAdaptParms;
    void FAR *   lpContext;
    void FAR *   lpWildcard;
    void FAR *   lpfnMS;
    void FAR *   lpMS;
    DWORD        dwWildcardSize;
    LPSTR        lpszMapDirectory;
    WORD         wCardNum;
```

```

WORD        wCleanupAction;
WORD        wScope;
UINT        uUnitSize;
BOOL        bBurst;
BOOL        bFromRule;
BOOL        bSource;
DWORD       dwRecords;
};
typedef struct tagExitParamStruct EXITPARAM;
typedef struct tagExitParamStruct FAR * LPEXITPARAM;

```

The engine environment sets up the `EXITPARAM` structure and the function should fill in the result. The engine will allocate and free the memory associated with `lpDataToApp`. The function must allocate the memory for `lpDataFromApp`. For all Windows operating systems, use the Windows macro `GlobalAllocPtr` defined in `windowsx.h`. For all other operating systems, use the C-runtime `malloc` function. The engine will free this memory.

**Table 14. Components of EXITPARAM as used with the EXIT function.**

Component	Used as	Usage
dwSize	Input	Size (in bytes) of EXITPARAM to assure correct compatibility
dwToLen	Input	Length (size in bytes) of lpDataToApp
dwFromLen	Output	Length (size in bytes) of lpDataFromApp
dwMapInstance		Not used
lpv		Not used
LpszCmdLine		Not used
lpDataToApp	Input	Data sent to the function. This is the third parameter of the EXIT function
lpDataFromApp	Output	Data sent back to the server from the function
uRetryCount		Not used
uRetryInterval		Not used
bRollback		Not Used
bCleanup		Not used
nReturn	Output	Return code based on the outcome of the function
szErrMsg	Output	String message based on nReturn
szFile		Not used
lpInternal		Not used
lpCmdStruct		Not used
lpAdaptParms		Not used



**Table 14. Components of EXITPARAM as used with the EXIT function. (continued)**

Component	Used as	Usage
lpContext		Not used
lpWildcard		Not used
dwWildcardSize		Not used
lpszMapDirectory		Not used
wCardNum		Not used
wCleanupAction		Not used
wScope		Not used
uUnitSize		Not used
bBurst		Not used
lpfnMS		Not used.
lpMS		Not used
dwRecords		Not used

There is no interaction with the **nReturn** or **szErrMsg** fields. However, this might change in a future release.

## GET

You can use the GET function to retrieve data using one of the source adapters, such as a messaging system, a database, a file, and so forth, within the course of your map.

When the GET function is used in a map, the default **OnSuccess** action is adapter specific. The default **OnFailure** action is to rollback any changes made during map processing. The default **Scope** will be integral unless the map is defined to run in bursts (which is the case when one or more inputs have the **FetchAs** property set to **Burst**).

GET can also be used for adapters that support request and reply, such as Socket.

### Syntax:

```
GET (single-text-expression , single-text-expression
    [, single-text-expression ])
```

### Meaning:

```
GET (adapter_alias, adapter_commands
    [, data_request_to_send_to_adapter ])
```

### Returns:

A single character text item

GET returns the data that is returned by the source adapter. The adapter identified by *adapter\_alias* is called using the specified *adapter\_commands* and passing *data\_request\_to\_send\_to\_adapter* as data to the adapter. See the Resource Adapters documentation.

You cannot use the GET function to retrieve a file that was built by an earlier output card within the same map. Instead, map the output of the earlier card to the later card.

To identify whether the function was successful, you can generally use the VALID function with the GET function. However, for certain adapters such as E-mail and FTP, the success state is not known until after map completion and using the VALID function in such cases might consistently return "success".

## Examples

- Reply (s) = GET ("SOCKET", "-HOST localhost -PORT 8015 -EOF -LSN 10 -T", PACKAGE (Request Object:::Input))

In this example, the GET function would connect to the socket-based server on localhost using port 8015. The Socket Adapter would submit the "Input" data to the server, and then wait up to 10 seconds for a reply to that message.

- Acct# = GET ("DB", "-dbtype ORACLE -connect shasta -user rjc -pw vm70" + "SELECT Acct# FROM CustMaster WHERE CustID = " + SenderID Field:Identification Segment:Msg + """)

In this example, the GET function could be used instead of a DBLOOKUP or DBQUERY function to retrieve data from a database from within in a map rule.

## Related functions

- FAIL
- LASTERRORCODE
- LASTERRORMSG
- PUT
- VALID

## JEXIT

The JEXIT function provides a means to interface with a method in an external Java class. Using this function, you can manipulate simple primitive types, simple objects, and complex objects.

There are two required text parameters for a JEXIT function. The first text parameter is the class name or key, depending on which meaning you use. The second text parameter is the method name. There are two optional text parameters for the data.

The first use of the JEXIT function creates an object that is based on the specified class name. Then, it invokes the named Java method, with any specified parameters, on that object. After the Java objects method is invoked, then JEXIT automatically destroys the Java object. If multiple invocations on the same Java object are required, then the

predefined "<init>" function can be used in the second parameter of Meaning 1. The returned key from the "<init>" function now references a specific Java object and can then be used multiple times as the first parameter in Meaning 2 function calls.

The second use of the JEXIT function invokes the named Java method, with any specified parameters, on a specific Java object. The method is invoked by using the specified key. This method allows Java objects to be manipulated throughout the map's execution cycle. These Java objects are stored in the HCL® OneTest™ Data object pool and are referenced during map execution by using the specified key. The specified key can be returned from either a JEXIT meaning 1 or meaning 2 invocation, particularly when a non-scalar, complex Java object must be returned.

JEXIT is not supported on CICS, Batch, or MVS.

## Prerequisites

To use JEXIT, do the following configuration steps, depending on the HCL® OneTest™ Data product you are using.

For HCL® OneTest™ Data with Command Server or HCL® OneTest™ Data with Launcher, either add the Java classes to your system class path or add the JAR files to the *install\_dir/dtx.ini* file.

Add the JAR files to the `jar` statements under the **[External Jar Files]** section of the *dtx.ini* file. For each JAR file, uncomment a `jar` statement and add the fully qualified path and JAR file name, including the file extension. If a `jar` statement is already set to a value, add the JAR file to the next `jar` statement in the sequence. As an example, if the `jar1` statement is set to a value, and the `jar2` statement is not set to a value, add your JAR file to the `jar2` statement.

For HCL® OneTest™ Data, on Windows operating systems, copy the JAR file to the classes folder in the path where version of application is installed. As an example, copy the JAR file to `%MQSI_WORKPATH%\shared-classes`. On UNIX operating systems, copy the JAR file to the `$MQSI_WORKPATH/shared-classes`.

If you are using Meaning 1, a public default constructor must be declared in the external Java class. If the default constructor is not available or needed, you can use the "<init>" method to instantiate the Java object, but only when there is a class constructor that is defined with one or two string-type parameters.

## Syntax

```
JEXIT ("single-text-expression" , "single-text-expression" , ["single-text-expression"] , ["single-text-expression"])
```

## Returns

A single text item.

## Meaning 1

```
JEXIT (class_name, method_name, [input_to_the_function_or_key], [input_to_the_function_or_key])
```

Using this approach, you provide a Java class and method name, and two optional inputs to the function.

When JEXIT is used in a map rule, at run time this function instantiates a Java virtual machine (JVM). The JVM reads your `.class` or `.jar` file and creates an object that is based on the defined Java class. The method, along with any specified parameters, is then invoked on the object.

When the returned object is a complex object, a unique identifier (key) is created that can be used to reference the object in the object pool.

All JEXIT parameters are text strings. Class name and method name are the two required parameters. The two optional parameters are generally input data. At run time, the strings are converted to UTF-8 and passed to the Java method, and the output from Java method is a UTF-8 text string.

The following simple Java class objects are internally converted to strings and returned to the map:

```
java.lang.Void
java.lang.Short
java.lang.Number
java.lang.Long
java.lang.Integer
java.lang.Float
java.lang.Double
java.lang.Byte
java.lang.Boolean
java.lang.String
java.lang.Character
```

## Meaning 2

```
JEXIT (key, method_name, [input_to_the_function_or_key], [input_to_the_function_or_key])
```

The second use of JEXIT is similar to Meaning 1. The difference is that it assumes that you already have a key that references an object in the object pool. In this scenario, you use JEXIT to invoke a method on an object by using the key as the first required parameter instead of the class name.

Whenever the JEXIT function returns a key, the associated Java object must be deleted before the map completes processing. The object can be deleted by using the predefined JEXIT "<destroy>" method. Failure to invoke the "<destroy>" method results in a JVM memory leak. To prevent the JVM heap from getting fragmented, exhausted, or both, make sure that all object references returned from the JEXIT function are deleted. To delete those object references, use the JEXIT function in the following way:

```
JEXIT("key", "<destroy>")
```

## Troubleshooting

To retrieve error messages for JEXIT, you can use the VALID and FAIL functions in your map rule. For example:

```
=VALID(JEXIT("key", "method_name", "data" ), FAIL(LASTERROMSG( ) ) )
```

## JEXIT Examples

The JEXIT examples are based on the following defined Java classes.

```
class :
package com.ibm.websphere.dtx.test;
```

```

public class TestJExit {
    Person person = new Person();
    public TestJExit() {
    }
    public String toUpper(String test) {
        return test.toUpperCase();
    }
    public String Concat(String test1, String test2) {
        return test1 + test2;
    }
    public Integer toNumber(String test1) {
        return new Integer(test1);
    }
    public Double toDouble(String test1) {
        return new Double(test1);
    }
    public void toNothing() {
    }
    public int returnint() {
        return 9;
    }
    public byte returnbyte() {
        return 8;
    }
    public Person getPerson() {
        return person;
    }
}

```

```

class :
package com.ibm.websphere.dtx.test;
public class Person{
    private String ext = new String("DTX");
    private String name = new String("Websphere");
    public Person() {
    }
    public String getName() {
        return name;
    }
    public String getExt() {
        return ext;
    }
}

```

### Simple primitive types and simple objects

- JEXIT ("com.ibm.websphere.dtx.test.TestJExit", "toUpper", "HCL")

Returns HCL

- JEXIT ("com.ibm.websphere.dtx.test.TestJExit", "Concat", "HCL Transformation", " Extender")

Returns HCL Transformation Extender

- JEXIT ("com.ibm.websphere.dtx.test.TestJExit", "toDouble", "12")

Returns 12.0

- JEXIT ("com.ibm.websphere.dtx.test.TestJExit" , "toNumber" , "12")

Returns 12

- JEXIT ("com.ibm.websphere.dtx.test.TestJExit" , "returnint")

Returns 9

- JEXIT ("com.ibm.websphere.dtx.test.TestJExit" , "returnbyte")

Returns 8

## Complex objects

When the function is returning a complex object, the returning object is added to the object pool and the key is returned to the map rule.

- JEXIT ("com.ibm.websphere.dtx.test.TestJExit " , "getPerson")

Returns a key (as a string)

- JEXIT ("key" , "getName")

Returns Websphere

- JEXIT ("key" , "<destroy>")

Deletes the `Person` object from the JVM memory. The "<destroy>" method must be invoked before the map completes processing.

## PUT

The PUT function passes data to a target adapter.

Use PUT to route data within your map using one of the target adapters such as to a messaging system, a database, a file, and so forth.

When the PUT function is used in a map, the default **OnSuccess** action is adapter specific. The default **OnFailure** action is to rollback any changes made during map processing. The default **Scope** will be integral unless the map is defined to run in bursts (which is the case when one or more inputs have the **FetchAs** property set to **Burst**).

### Syntax:

PUT (single-text-expression , single-text-expression , single-text-expression)

### Meaning:

PUT (adapter\_alias , adapter\_commands , data\_to\_send\_to\_adapter)

### Returns:

"None"

The adapter identified by *adapter\_alias* is called using the specified *adapter\_commands* and passing *data\_to\_send\_to\_adapter* as data to the adapter.

The PUT function is not processed if the third argument is zero-sized data or NONE.

To identify whether the function was successful, you can generally use the VALID function with the PUT function. However, for certain adapters such as E-mail and FTP, the success state is not known until after map completion and using the VALID function in such cases might consistently return "success".

## Examples

This example illustrates a mapping situation in which a set of messages is being produced in the output (output card #1). However, the ultimate target for these messages is a message queue and the messages need to be placed on the queue one at a time.

To accomplish this, the first output card builds the set of messages and its **Target** is defined as **Sink**. So, this set of messages is constructed in memory, but is discarded after the map completes. A second output card, uses the following **PUT** function to put each output message (from output card# 1), individually, on the output queue.

```
Message(s) = PUT ("MQS", "-QMN myqueuemgr -QN chips_queue -T", PACKAGE
(PaymentMessage:CHIPS_Payment_Message))
```

- The first argument, MQS, identifies that the data is to be routed using the IBM WebSphere MQ (server) adapter.
- The second argument, "-QMN myqueuemgr -QN chips\_queue -T", provides the adapter commands needed by the adapter to put the message on the queue.
- The third argument, PACKAGE, passes the data that is to be sent as the body of the message.

## Related functions

- FAIL
- GET
- LASTERRORCODE
- LASTERRORMSG
- VALID

## RUN

The RUN function allows you to execute another compiled map from a component or map rule.

You can use RUN to dynamically name source or destination files, or both, or to dynamically pass data to a map. You can also use the RUN function to split the output data into separate files based on some value in the input.

### Syntax:

```
RUN (single-text-expression [ , single-text-expression ])
```

**Meaning:**

RUN (map\_to\_run [, command\_option\_list ])

**Returns:**

A single text item

The first argument, *map\_to\_run*, is an expression identifying the name of the compiled map (**.mmc**) to be run.

The *command\_option\_list* argument is an optional argument that you can use to specify execution commands applicable to the map to be run. *Command\_option\_list* is a text item containing a series of execution commands separated by a space. Any execution command can be used as part of the *command\_option\_list* argument. For example, you can send data to another map by using the echo command option (**-IE{x}**).

See the Execution Commands documentation for a list of command options.

The result of the RUN function depends on the command options in *command\_option\_list*.

**Echo command option**

- If you use the Echo command option for an output card, the data from that card will be passed back as a text item to the object in the map from which it was run.
- If you use the Echo command option for more than one output card, the data from all echoed cards will be concatenated together and passed back as a text-item to the object in the map from which it was run.
- If you do not use the Echo command option, the return code indicating the status of the map that was run will be passed back to the object in the map from which it was run.

**Examples**

- RUN ("MyMap", "-IE1s502 " + Invoice:File + " -OF1 *install\_dir*\\" + CustomerID:Invoice)

This example runs the **MyMap** map, sending 502-byte fixed-size **Invoice** data as the data source for input card **1**, overriding the filename of output card **1** based on **Customer** data and returns the map return code as the result.

- RUN ("GetDbOpt", " ")

This example runs the **GetDbOpt** map (with no command options specified) and returns the map return code as the result.

- RUN ("DoOneSet", "-A -GR" + ECHOIN( 1 , Set:InFile ) ) + " -OF1 OUT\_SET." + NUMBERTOTEXT ( INDEX ( \$ ) ) + " -OE2" )

This example runs the **DoOneSet** map. Command options include the following choices:

- "-A -GR"

The **DoOneSet** map will produce no **Audit Log** and restrictions will be ignored.

- ECHOIN( 1 , Set:InFile )



The ECHOIN function creates the -IE command option for echoing the data represented by Set:InFile to input **1** of the RUN map.

- "-OF1 OUT\_SET." + NUMBERTOTEXT ( INDEX ( \$ ) )

The output file for card **1** of the **DoOneSet** map will be called "OUT\_SET." plus a sequence number based on the index of the Set in **InFile**. For example, the first output set will be OUT\_SET.1, and so forth.

- "-OE2"

Using the output echo command option, the data built for output card **2** of the **DoOneSet** map will be returned as the result of the RUN function.

An alternative to using the ECHOIN function shown above is using the long version. For example, replace ECHOIN( 1 , Set:InFile ) with:

```
"-IE1S" + NUMBERTOTEXT ( SIZE ( Set:InFile ) + " " + TEXT ( Set:InFile )
```

Using the Echo input command option (**-IE**) with the sizing method (**Sn**) or the ECHOIN function, one **Set** object of **InFile** is passed to input card **1** for the **DoOneSet** map.

## Related functions

- DDEQUERY

## RENAMEFILE

The RENAMEFILE function renames the source file to that of the destination file. If the operation succeeds, the returned value is 0; if it fails, then an operating system-specific error number is returned.

### Syntax:

```
RENAMEFILE (single-text-expression, single-text-expression, single-text-expression, single-text-expression)
```

### Meaning:

```
RENAMEFILE (source, destination, number_of_retries (max 100), pause_between_retries_in_ms (max 100000))
```

### Returns:

A single number

---

Related reference

[COPYFILE on page 847](#)

## HARDLINKFILE

A hard link points or references to a specific space on the hard drive. Multiple files can be hard linked to the same space in the hard drive. If some data is changed on one of the hard linked files, all the other files reflect that change. The HARDLINKFILE function returns the hard link file's command status. On successful execution of this function: 0 is returned; on failure: an operating system-specific error code is returned.

**Syntax:**

HARDLINKFILE(*single-text-expression, single-text-expression, single-text-expression, single-text-expression*)

**Meaning:**

HARDLINKFILE [oldpath, newpath, number\_of\_retries (max 100), pause\_between\_retries\_in\_ms (max 100000)]

**Returns:**

A single number

---

Related reference

[SOFTLINKFILE on page 846](#)

## SOFTLINKFILE

A soft link points to a specific file, which in turn, references to a particular location on the hard drive. The soft link, also known as symbolic link is a second file that exists independently of its target. The SOFTLINKFILE function returns the soft (symbolic) link file's command status. On successful execution of this function: 0 is returned; on failure: an operating system-specific error code is returned.

**Syntax:**

SOFTLINKFILE(*single-text-expression, single-text-expression, single-text-expression, single-text-expression*)

**Meaning:**

SOFTLINKFILE [(oldpath, newpath, number\_of\_retries (max 100), pause\_between\_retries\_in\_ms (max 100000)]

**Returns:**

A single number

---

Related reference

[HARDLINKFILE on page 846](#)

## COPYFILE

The COPYFILE function copies the source file to the destination file. It returns the copy file command status. On successful execution of this function: 0 is returned; on failure: an operating system-specific error code is returned.

**Syntax:**

`COPYFILE(single-text-expression, single-text-expression, single-text-expression, single-text-expression)`

**Meaning:**

`COPYFILE [source, destination, number_of_retries (max 100), pause_between_retries_in_ms (max 100000)]`

**Returns:**

A single number

---

Related reference

[RENAMEFILE on page 845](#)

## SLEEP

The SLEEP function suspends the execution of the current map until the time-out interval elapses. The time-out or sleep interval is represented in milliseconds. On successful execution of this function: 0 is returned; on failure: -1 is returned.

**Syntax:**

`SLEEP(single-number-expression)`

**Meaning:**

SLEEP (the sleep time is in milliseconds)

**Returns:**

A single number

## Inspection functions

### ABSENT

The ABSENT function tests for the absence of an object.

**Syntax:**

`ABSENT (single-object-expression)`

**Meaning:**

`ABSENT (object_to_test)`

**Returns:**

True or false

ABSENT returns "true" if the *object\_to\_test* evaluates to "none". If the *object\_to\_test* does not evaluate to "none", the function returns "false".

**Examples**

- You can use this function to map an object only if another object is absent. For example, you might want to map **BillTo** information to the **ShipTo** fields if the **ShipToName** is absent.
- ABSENT (AreaCode:Phone)

This example evaluates to "true" when **AreaCode:Phone** evaluates to "none" or evaluates to "false" when **AreaCode:Phone** does not evaluate to "none".

**Related Functions**

- PRESENT

## CONTAINSERRORS

The CONTAINSERRORS function tests a valid object to see whether it contains any objects in error.

**Syntax:**

CONTAINSERRORS (single-object-expression)

**Meaning:**

CONTAINSERRORS (*object\_to\_test*)

**Returns:**

True or false

The CONTAINSERRORS function returns "true" if any object contained in *object\_to\_test* is in error; it returns "false" if the *object\_to\_test* is completely valid.

The input object, itself, is a *valid* input object. This function does *not* evaluate for invalid objects. Therefore, if you have map rule:

CONTAINSERRORS ( Invoice:InputFile )

and **Invoice**[1] is valid, **Invoice**[2] is invalid, and **Invoice**[3] is valid, then CONTAINSERRORS evaluates only twice-once for each *valid* instance of **Invoice**.

## Examples

- `Msg(s) = IF ( CONTAINSERRORS ( Msg:MailBag ) & Type:Msg:MailBag = "PRIORITY" , Msg:MailBag , "none" )`

In this example, if **Msg:MailBag** contains any object in error and **Type:Msg:MailBag** has a value of "PRIORITY", **Msg:MailBag** is mapped; otherwise, "none" is returned for this occurrence of **Msg**. This map rule returns all valid messages (**Msg**) that contain errors with a **Type** of "PRIORITY".

## Related functions

- ISERROR
- REFORMAT

## DELETEFILE

The DELETEFILE function returns the status of a file deletion command as a single number. DELETEFILE returns 0 for successful file deletion or the appropriate error code if file deletion was not successful.

### Syntax:

```
resourceLib->DELETEFILE (single-text-expression, single-text-expression, single-text-expression)
```

### Meaning:

```
DELETEFILE (filename, number_of_retries, pause_between_retries)
```

### Returns:

A single number

### *number\_of\_retries*

The number of times the DELETEFILE function is to check for the deleted file. The maximum is 100 retries.

### *pause\_between\_retries*

The amount of time, in milliseconds, that the function waits between checking for the deleted file. The maximum is 100,000.

## Example

```
(resourceLib->DELETEFILE(GETDIRECTORY () + "myfile1.txt","10","1000"))
```

## ISALPHA

You can use ISALPHA when you need to know whether a text string is all alphabetic characters.

### Syntax:

```
ISALPHA (single-text-expression)
```

**Meaning:**

ISALPHA (text\_to\_test)

**Returns:**

This function evaluates to a Boolean "true" or "false" and should only be used as a conditional expression within a logical function.

The ISALPHA function tests a text object to see if it contains all alphabetic characters.

If *text\_to\_test* contains only alphabetic characters (for example, A-Z and a-z), ISALPHA evaluates to "true".

If *text\_to\_test* contains other than just alphabetic characters, ISALPHA returns "false".

**Examples**

- IF(ISALPHA ("AnywhereUSA"))

Returns "true"

- IF(ISALPHA ("Anywhere USA"))

Returns "false"

- IF(ISALPHA ("Mr. Brown"))

Returns "false"

**Related functions**

- ISLOWER
- LEAVEAL-PHANUM
- ISNUMBER
- LEAVENUM
- ISUPPER
- LEAVEPRINT
- LEAVEAL-PHA

**ISERROR**

The ISERROR function tests an object to see if it is in error

You can use ISERROR to output your data in exactly the same order as it occurs in your input, both the valid data and the data in error. You can also use ISERROR to produce error messages for bad data in the same file in which you map your good data.

**Syntax:**

```
ISERROR (single-object-name)
```

**Meaning:**

```
ISERROR (object_to_test)
```

**Returns:**

```
"True" or "false"
```

ISERROR returns "true" when *object\_to\_test* is in error and returns "false" when *object\_to\_test* is completely valid.

**Examples**

- InfoRec (s) = IF ( ISERROR ( Record:SomeFile ), "Bad --> " + REJECT ( Record:SomeFile ), "Ok --> " + TEXT ( Record:SomeFile ) )

In this example, ISERROR is used to produce a report for *all Record* objects in **SomeFile**. If the record is in error, the **InfoRec** will have the text Bad --> followed by the data from the input **Record**. If the record is valid, the **InfoRec** will have the text Ok --> followed by the data from the input **Record**, such as:

```
Ok --> SZ-68839,486 Upgrade Microprocessor,186.86,100,W200
Bad --> MK-19309,,369.43,417,W100
Ok --> KL-20349,PCMCIA Network Adaptor,174.82,29,N300
Ok --> WP-37679,AC Adaptor,39.48,245,E100
Bad --> IL-39890,8MB Memory PCMCIA,390.48,0,S100
```

**Related functions**

- CONTAINSERRORS

**ISLOWER**

The ISLOWER function tests a text object to see if it contains all lowercase alphabetic characters.

**Syntax:**

```
ISLOWER (series-text-expression)
```

**Meaning:**

```
ISLOWER (text_to_test)
```

**Returns:**

This function evaluates to a Boolean "true" or "false" and should only be used as a conditional expression within a logical function.

If *text\_to\_test* contains only lowercase alphabetic characters (for example, a-z), ISLOWER evaluates to "true".

If *text\_to\_test* contains other than lowercase alphabetic characters, ISLOWER returns "false".

## Examples

- IF(ISLOWER ("company"))

Returns "true"

- IF(ISLOWER ("pots and pans"))

Returns "false"

- IF(ISLOWER ("Andrew"))

Returns "false"

## Related functions

- ISALPHA
- LEAVEAL-PHANUM
- ISNUMBER
- LEAVENUM
- ISUPPER
- LEAVEPRINT
- LEAVEAL-PHA

## ISNUMBER

The ISNUMBER function tests a text object to determine whether it contains all numeric characters.

### Syntax:

ISNUMBER (single-text-expression)

### Meaning:

ISNUMBER (text\_to\_test)

### Returns:

This function evaluates to a Boolean "true" or "false" and should only be used as a conditional expression within a logical function.

If *text\_to\_test* contains only digits (for example, 0-9), ISNUMBER evaluates to "true".



If *text\_to\_test* contains other than digits, ISNUMBER evaluates to "false".

## Examples

- `IF(ISNUMBER("1"))`

Returns "true"

## Related functions

- ISALPHA
- LEAVEAL-PHANUM
- ISLOWER
- LEAVENUM
- ISUPPER
- LEAVEPRINT
- LEAVEAL-PHA

## ISUPPER

The ISUPPER function tests a text object to determine whether it contains all uppercase alphabetic characters.

### Syntax:

ISUPPER (series-text-expression)

### Meaning:

ISUPPER (text\_to\_test)

### Returns:

This function evaluates to a Boolean "true" or "false" and should only be used as a conditional expression within a logical function.

If *text\_to\_test* contains only uppercase alphabetic characters (for example, A-Z), ISUPPER evaluates to "true".

If *text\_to\_test* contains other than uppercase alphabetic characters, ISUPPER returns "false".

## Examples

- IF(ISUPPER ( "BOMBAY" ))

Returns "true"

- IF(ISUPPER ( "CD-ROM" ))

Returns "false"

- IF(ISUPPER ( "Map Designer" ))

Returns "false"

## Related functions

- ISALPHA
- LEAVEAL-PHANUM
- ISLOWER
- LEAVENUM
- ISNUMBER
- LEAVEPRINT
- LEAVEAL-PHA

## MEMBER

Use MEMBER when you need to know whether an object occurs within a series.

The MEMBER function searches a series, looking for a single specified object in the series. If any object in the series matches the specified object, MEMBER returns "true". If there is no match, MEMBER returns "false".

### Syntax:

MEMBER (single-object-expression , series-object-expression)

MEMBER (single-object-expression , { literal, literal ... })

### Meaning:

MEMBER (object\_to\_look\_for , series\_of\_objects\_to\_look\_at)

### Returns:

"True" or "false"

MEMBER returns "true" if *object\_to\_look\_for* matches one of the values in *series\_of\_objects\_to\_look\_at*.

It returns "false" if *object\_to\_look\_for* does *not* match at least one of the values in *series\_of\_objects\_to\_look\_at*.

The two arguments, *object\_to\_look\_for* and *series\_of\_objects\_to\_look\_at*, must be objects of the same item interpretation or the same group type. For example, if *object\_to\_look\_for* is a date/time item, *series\_of\_objects\_to\_look\_at* must be a series of date/time items.

## Examples

- MEMBER (EntityIDCode:Name, {"BT", "ST"})

This example tests whether **EntityIDCode** has one of a particular set of literal values.

- MEMBER (Store# , EntityIDCode:Name)

This example tests whether **Store#** has the same value as any **EntityIDCode:Name**.

## Related functions

- EXTRACT
- LOOKUP

## NOT

Use the NOT function to test a condition and have it return the inverse of its "true" or "false" result. For example, you want the function to return "true" if the condition results in "false".

### Syntax:

NOT (single-condition-expression)

### Meaning:

NOT (condition\_to\_evaluate)

### Returns:

"True" or "false"

NOT returns "true" if the condition evaluates to "false" and returns "false" if the condition evaluates to "true".

## Examples

- NOT (Qty::InputFile = 0)

This example returns "false" if the **Qty** equals 0 (the condition is true) and returns "true" if **Qty** does not equal 0 (the condition is false).

- IF (NOT (PRESENT (StartDate)), "Unknown", "none")

This example returns "unknown" if **StartDate** is not present and returns "none" if **StartDate** is present.

Another way to test that an object is not present is to use the ABSENT function.

## OFFSET

Use the OFFSET function when you need to know the position of a particular data object within its card object.

OFFSET returns an integer representing the offset of the specified object within the data.

### Syntax:

OFFSET (single-object-expression)

### Meaning:

OFFSET (object\_whose\_offset\_is\_needed)

### Returns:

A single integer

OFFSET returns the offset, in bytes, of the specified object within its card object, beginning at offset 0. For an object that has an initiator, the offset will apply to the first byte of the data. For an object that is right-justified with pad characters, OFFSET will return the offset of the first byte of data.

The OFFSET function works the same for output objects as it does for input objects.

## Examples

- OFFSET (Application:LoanData)

In this example, if the first character of the first occurrence of the object **Application** occurs 210 bytes from offset 0 within **LoanData**, the OFFSET function returns 210.

## Related function

- SIZE

## OR

Use the OR function to test whether one of a series of conditions is true.

OR evaluates a series of conditions and returns "true" if at least one evaluates to "true"; otherwise returns "false".

### Syntax:

OR (series-condition-expression)

### Meaning:

OR (conditions\_to\_evaluate)

**Returns:**

"True" or "false"

OR evaluates to "true" if *any* member of the argument evaluates to "true" and evaluates to "false" if *all* members of the argument evaluate to "false".

**Examples**

- Order(s)=IF (OR (Store:Table = Store#:Order:Input), Order:Input)

This example produces an **Order** if the **Store#** of an **Order** in Input matches any **Store** in **Table**.

**Related function**

- ALL

**PARTITION**

The PARTITION function checks to see if an occurrence of an object belongs to a certain partition. If the object is that partition, "true" is returned. Otherwise, "false" is returned.

**Syntax:**

PARTITION (single-object-expression, single-simple-object-name)

**Meaning:**

PARTITION (partitioned\_object, simple\_name\_of\_partition\_to\_check\_for)

**Returns:**

"True" or "false"

PARTITION returns "true" if the data object of *partitioned\_object* belongs to the partition represented by *simple\_name\_of\_partition\_to\_check\_for*. Otherwise, PARTITION returns "false".

**Examples**

- Assume that **Transaction** has been partitioned into three partitioned subtypes: **Invoice**, **Order**, and **Remittance**, as shown.

The following rule could be used to detect whether a given **Transaction** is an **Invoice**:

PARTITION (Transaction::Batch, Invoice)

If the **Transaction** is an **Invoice**, PARTITION returns "true". If the **Transaction** is not an **Invoice** (for example, it is an **Order** or a **Remittance**), PARTITION returns "false" for that **Transaction**.

## Related function

- GETPARTITIONNAME

## PRESENT

The PRESENT function tests for the presence of an object.

PRESENT is commonly used with the IF function in a map rule to provide conditional logic. For example, if the object is present, do this; otherwise, do something else.

Similarly, PRESENT is commonly used with the WHEN function in component rules to provide conditional validation logic.

### Syntax:

PRESENT (single-object-expression)

### Meaning:

PRESENT (object\_to\_look\_for)

### Returns:

"True" or "false"

PRESENT returns "true" if the input argument does *not* evaluate to "none"; the object is present. It returns "false" if the input argument evaluates to "none"; the object is not present.

## Examples

- PRESENT (Trailer:File)

This example returns "true" if **Trailer** is present and returns "false" if **Trailer** is absent.

- IF (PRESENT(MiddleInitial::Input), MiddleInitial::Input, "\*\*\*" )

This example in a map rule maps **MiddleInitial** if it is present. If **MiddleInitial** is not present, three asterisks are mapped.

- WHEN (PRESENT (AreaCode Field), PRESENT (PhoneNo Field))

In this example, PRESENT is being used in conjunction with the WHEN function in a component rule to determine whether a particular object is valid.

## Related functions

- ABSENT
- IF
- WHEN

## SIZE

The SIZE function returns an integer representing the size of a specified object, exclusive of any pad characters.

**Syntax:**

SIZE (single-object-expression )

**Meaning:**

SIZE (object\_whose\_size\_is\_needed )

**Returns:**

A single integer

The SIZE function returns the size, in bytes, of *object\_whose\_size\_is\_needed*. The byte size returned does not include any pad characters that might be in the object, but does include separators and signs.

The size of a group is the size, beginning with the first character of the first component and ending with the last character of the last component, of the group. If the group has delimiters, the infix delimiters are included in the size.

### Examples

- SIZE ( Transaction )

Returns 8000 if the size of **Transaction** (without pad characters) is 8000 bytes

### Using SIZE with NORMXML

If the SIZE function is used with the NORMXML function to determine the size of the specified object after NORMXML has removed the XML formatting from the input XML fragment, the SIZE operation calculates the size of the data in Unicode, since the NORMXML function converts the input data to Unicode before it removes the XML formatting. The returned size does not match the size that would have been calculated if the data remained in its original character set, unless the original character set was Unicode.

## TESTOFF

The TESTOFF function tests a specified bit in a binary number item to see whether it is off.

**Syntax:**

TESTOFF ( single-binary-number-expression , single-integer-expression )

**Meaning:**

TESTOFF ( binary\_number\_to\_test , bit\_to\_test )

**Returns:**

"True" or "false"

The value of *bit\_to\_test* specifies which bit of *binary\_number\_to\_test* should be tested for the value 0. If *bit\_to\_test* has the value 1, it refers to the leftmost bit of *binary\_number\_to\_test*.

The TESTOFF function returns "true" if the specified bit is off and returns "false" if the specified bit is on.

If *bit\_to\_test* is less than one or greater than the number of bits of *binary\_number\_to\_test*, TESTOFF returns "false".

## Examples

- TESTOFF (A, 16)

Assume **A** is the two-byte binary value of "1", which is all zeros except for bit **16**. The binary representation of the value in **A** is 0001.

This example returns "false".

- TESTOFF (A, 20)

Returns "false" because bit **20** does not exist.

## TESTON

The TESTON function tests a specified bit in a binary number to see if it is on.

### Syntax:

TESTON ( single-binary-number-expression, single-integer-expression )

### Meaning:

TESTON ( binary\_number\_to\_test ,bit\_to\_test )

### Returns:

"True" or "false"

The value of *bit\_to\_test* specifies the bit of *binary\_number\_to\_test* to test for the value 1. If *bit\_to\_test* has the value 1, it refers to the leftmost bit of *binary\_number\_to\_test*.

The function returns "true" if the specified bit is on; it has the value 1. It returns "false" if the specified bit is off; it has the value 0.

If *bit\_to\_test* is less than one or greater than the number of bits of *binary\_number\_to\_test*, TESTON returns "false".

## Examples

- TESTON ( A , 16 )

Assume **A** is the two-byte binary value of "1", which is all zeros except for bit **16**. The binary representation of the value in **A** is 0001.

This example returns "true".

- TESTON ( A , 20 )

Returns "false" because bit **20** does not exist in a two-byte value.



## VALID

You can use the VALID function to perform conditional processing based on whether an external interface function executes successfully.

VALID returns the result of the first argument if it is valid; otherwise, returns the second argument.

### Syntax:

```
VALID ( single-text-expressions , single-general-expression )
```

### Meaning:

```
VALID ( function_that_can_fail , return_value_if_function_fails )
```

### Returns:

A single text expression

VALID returns the result of the evaluation of *function\_that\_can\_fail* if it is valid. If the function fails, VALID returns *return\_value\_if\_function\_fails*.

The following functions can fail:

- DBLOOKUP
- GET
- DBQUERY
- PUT
- DDEQUERY
- RUN
- EXIT

## Examples

```
SomeObject = VALID ( RUN ( "mymap.mmc" , "-OFl mydata.txt" ) , FAIL ( "My RUN failed!" ) )
```

If the RUN function returns an error return code, the VALID functions returns `none`, the map aborts, and the message `My RUN failed!` is reported under Execution Summary in the execution audit log.

## Related functions

- DBLOOKUP
- DBQUERY
- DDEQUERY

## Logical functions

### ALL

The ALL function evaluates a series of conditions and returns "true" if they all evaluate to "true"; otherwise returns "false".

**Syntax:**

ALL (series-condition-expression)

**Meaning:**

ALL (conditions\_to\_evaluate)

**Returns:**

True or false

The ALL function evaluates to "true" if *all* members of the input argument evaluate to "true"; it evaluates to "false" if *any* member of the input argument is "false".

### Examples

- You can use ALL when you want to test whether all conditions of a series are true.
- PO (s)=IF (ALL( PO#:Line:Order = PO#:Line:Order[1]), Order, "none")

If each and every **PO#** matches the **PO#** of the first **Order**, ALL evaluates to "true". Otherwise, ALL evaluates to "false".

### Related functions

- NOT
- OR

### BINDABS

Use this function in a component rule for the implementation of binding and iterative evaluation. The implementation of binding ensures that when there are multiple references to a component in a rule, all references are to the same instance of that component. The implementation of iterative evaluation ensures that the rule is evaluated using all possible combinations of type references.

Do not use this function unless you have a clear understanding of the behavior.

**Syntax:**

BINDABS (Boolean\_expression)

**Meaning:**

BINDABS (component rule)

**Returns:**

A Boolean "true" or "false"



**Note:** It is not necessary to use the BINDABS function for iterative behavior with the following functions because these functions already iterate: COUNT, COUNTABS, INDEX, MEMBER, MAX, MIN, SUM, and SERIESTOTEXT.

**How it works**

When the evaluation starts and the BINDABS function is encountered at the beginning of the component rule, binding is enforced and the rule is iterated across all valid combinations of objects specified in the rule.

BINDABS must be positioned first in the component rule and the entire component rule must be enclosed. Any other use of this function will have no effect.

When using BINDABS, the evaluation order is the same as mapping. Rules that contain both a normal instance and a last instance for the first instance of the type are not evaluated.

**Example**

```

BINDABS (Qty:LineItem:Order * Price:LineItem:Order = ExtendedPrice:LineItem:Order)

```

As a result of binding, the Qty, Price, and ExtendedPrice elements in this example are all in the same LineItem of the same Order. As a result of iteration, all line items of all orders are checked.

**EITHER**

The EITHER function returns the result of the first argument that does not evaluate to "none".

**Syntax:**

```

EITHER (single-general-expression { , single-general-expression } )

```

**Meaning:**

```

EITHER ( try_this { , if_none_try_this } )

```

**Returns:**

A single object

The EITHER return methods work in the following ways:

**Returns...**

When...

***try\_this***

*try\_this* does not evaluate to "none"

***if\_none\_try\_this***

*try\_this* evaluates to "none"

**the next *if\_none\_try\_this***

the first *if\_none\_try\_this* evaluates to "none"



**Note:** The Design Studio compiler does not validate the second argument or subsequent arguments against the type tree.

**Examples**

- EITHER ( OrderDate Field , CURRENTDATE ( ) )

If **OrderDate Field** does not evaluate to "none", it is returned. If **OrderDate Field** evaluates to "none", the current system date (using CURRENTDATE) is returned.

- EITHER ( LOOKUP ( PriorityCd:Msg , CustID:Msg = "93X" ) , 8 )

This example returns the result of the LOOKUP if that result does not evaluate to "none"; otherwise, it returns the second argument of 8.

- EITHER ( IF ( SomeCode = "C" , CustomerID:Input ) , IF ( SomeCode = "S" , SupplierID:Input ) , IF ( SomeCode = "O" , Reference#:Input ) , "####" )

If **SomeCode** is **C** and **CustomerID:Input** has a value, EITHER returns the value of **CustomerID**. Otherwise, if **SomeCode** is **S** and **CustomerID:Input** has a value, EITHER returns the value of **SupplierID**. Otherwise, if **SomeCode** is **O** and **Reference#:Input** has a value, EITHER returns the value of **Reference#**. If none of those conditions result in a value, EITHER returns "####".

- You can use EITHER when you want a default value when an expression evaluates to "none" and the expression might cause common arguments to produce an unintended result. For example, use:

```
EITHER (LOOKUP (PriorityCd:Msg , CustID:Msg = "93X") , 8)
```

-instead of-

```
IF (PRESENT (LOOKUP (PriorityCd:Msg , CustID:Msg = "93X" ) ) , LOOKUP ( PriorityCd:Msg , CustID:Msg = "93X" ) , 8 )
```

**IF**

You can use the IF function for conditional logic. For example, to return one of two objects depending on the evaluation of a condition.

IF evaluates a conditional expression, returning one value if true, another if false.

**Syntax:**

IF (single-condition-expression , single-general-expression  
 [ , single-general-expression ])

**Meaning:**

IF (test\_this , result\_if\_true [ , result\_if\_false ])

**Returns:**

A single item or single group

If *test\_this* evaluates to "true", the result is *result\_if\_true*.

Otherwise, if *test\_this* evaluates to "false", the result is *result\_if\_false*. If no *result\_if\_false* is specified, this evaluates to "none".

The *result\_if\_true* and *result\_if\_false* arguments must correspond to the same item interpretation or the same group type or either can be "none". For example, if *result\_if\_true* evaluates to a number, *result\_if\_false* must also evaluate to a number. If *result\_if\_true* evaluates to a **LinItem** group, *result\_if\_false* must also evaluate to a **LinItem** group.

**Examples**

- IF (Quantity:LinItem:PO > 500, "PRIORITY", "REGULAR")

This example tests the **Quantity** value. If that **Quantity** is > 500, the IF function evaluates to the value PRIORITY. If that **Quantity** is <= 500, the IF function evaluates to the value, REGULAR.

- IF (Status:Order = "Special", "SPCL")

This example tests the **Status** value. If the **Status** is Special, the IF function evaluates to the value, SPCL. Otherwise, it evaluates to "none". Notice that this rule is equivalent to

IF (Status:Order = "Special", "SPCL", "none")

**ISALPHA**

You can use ISALPHA when you need to know whether a text string is all alphabetic characters.

**Syntax:**

ISALPHA (single-text-expression)

**Meaning:**

ISALPHA (text\_to\_test)

**Returns:**

This function evaluates to a Boolean "true" or "false" and should only be used as a conditional expression within a logical function.

The ISALPHA function tests a text object to see if it contains all alphabetic characters.

If *text\_to\_test* contains only alphabetic characters (for example, A-Z and a-z), ISALPHA evaluates to "true".

If *text\_to\_test* contains other than just alphabetic characters, ISALPHA returns "false".

## Examples

- IF(ISALPHA ("AnywhereUSA"))

Returns "true"

- IF(ISALPHA ("Anywhere USA"))

Returns "false"

- IF(ISALPHA ("Mr. Brown"))

Returns "false"

## Related functions

- ISLOWER
- LEAVEAL-PHANUM
- ISNUMBER
- LEAVENUM
- ISUPPER
- LEAVEPRINT
- LEAVEAL-PHA

## ISLOWER

The ISLOWER function tests a text object to see if it contains all lowercase alphabetic characters.

### Syntax:

ISLOWER (series-text-expression)

### Meaning:

ISLOWER (text\_to\_test)

### Returns:

This function evaluates to a Boolean "true" or "false" and should only be used as a conditional expression within a logical function.

If *text\_to\_test* contains only lowercase alphabetic characters (for example, a-z), ISLOWER evaluates to "true".

If *text\_to\_test* contains other than lowercase alphabetic characters, ISLOWER returns "false".

## Examples

- IF(ISLOWER ("company"))

Returns "true"

- IF(ISLOWER ("pots and pans"))

Returns "false"

- IF(ISLOWER ("Andrew"))

Returns "false"

## Related functions

- ISALPHA
- LEAVEAL-PHANUM
- ISNUMBER
- LEAVENUM
- ISUPPER
- LEAVEPRINT
- LEAVEAL-PHA

## ISNUMBER

The ISNUMBER function tests a text object to determine whether it contains all numeric characters.

### Syntax:

ISNUMBER (single-text-expression)

### Meaning:

ISNUMBER (text\_to\_test)

### Returns:

This function evaluates to a Boolean "true" or "false" and should only be used as a conditional expression within a logical function.

If *text\_to\_test* contains only digits (for example, 0-9), ISNUMBER evaluates to "true".

If *text\_to\_test* contains other than digits, ISNUMBER evaluates to "false".

## Examples

- `IF ( ISNUMBER ( " 1 " )`

Returns "true"

## Related functions

- ISALPHA
- LEAVEAL-PHANUM
- ISLOWER
- LEAVENUM
- ISUPPER
- LEAVEPRINT
- LEAVEAL-PHA

## ISUPPER

The ISUPPER function tests a text object to determine whether it contains all uppercase alphabetic characters.

### Syntax:

ISUPPER (series-text-expression)

### Meaning:

ISUPPER (text\_to\_test)

### Returns:

This function evaluates to a Boolean "true" or "false" and should only be used as a conditional expression within a logical function.

If *text\_to\_test* contains only uppercase alphabetic characters (for example, A-Z), ISUPPER evaluates to "true".

If *text\_to\_test* contains other than uppercase alphabetic characters, ISUPPER returns "false".



## Examples

- IF(ISUPPER ( "BOMBAY" ))

Returns "true"

- IF(ISUPPER ( "CD-ROM" ))

Returns "false"

- IF(ISUPPER ( "Map Designer" ))

Returns "false"

## Related functions

- ISALPHA
- LEAVEAL-PHANUM
- ISLOWER
- LEAVENUM
- ISNUMBER
- LEAVEPRINT
- LEAVEAL-PHA

## NOT

Use the NOT function to test a condition and have it return the inverse of its "true" or "false" result. For example, you want the function to return "true" if the condition results in "false".

### Syntax:

NOT (single-condition-expression)

### Meaning:

NOT (condition\_to\_evaluate)

### Returns:

"True" or "false"

NOT returns "true" if the condition evaluates to "false" and returns "false" if the condition evaluates to "true".

## Examples

- NOT (Qty::InputFile = 0)

This example returns "false" if the **Qty** equals 0 (the condition is true) and returns "true" if **Qty** does not equal 0 (the condition is false).

- IF (NOT (PRESENT (StartDate)), "Unknown", "none")

This example returns "unknown" if **StartDate** is not present and returns "none" if **StartDate** is present.

Another way to test that an object is not present is to use the ABSENT function.

## OR

Use the OR function to test whether one of a series of conditions is true.

OR evaluates a series of conditions and returns "true" if at least one evaluates to "true"; otherwise returns "false".

### Syntax:

OR (series-condition-expression)

### Meaning:

OR (conditions\_to\_evaluate)

### Returns:

"True" or "false"

OR evaluates to "true" if *any* member of the argument evaluates to "true" and evaluates to "false" if *all* members of the argument evaluate to "false".

## Examples

- Order(s)=IF (OR (Store:Table = Store#:Order:Input), Order:Input)

This example produces an **Order** if the **Store#** of an **Order** in Input matches any **Store** in **Table**.

## Related function

- ALL

## WHEN

You can use the WHEN function in a component rule to test for the validity of one object based on another object.

WHEN evaluates a condition. Then, based on that evaluation, evaluates another condition and returns "true" or "false".

### Syntax:

WHEN ( single-condition-expression , single-condition-expression [ , single-condition-expression ] )

**Meaning:**

WHEN ( condition1 , condition2 [ , condition3 ] )

**Returns:**

"True" or "false"

The following statements summarize how the WHEN function works:

- Returns "true" if *condition1* and *condition2* both evaluate to "true".
- Returns "true" if *condition1* evaluates to "false" and there is no *condition3* or if *condition1* evaluates to "false" and *condition3* evaluates to "true".
- Returns "false" if *condition1* evaluates to "true" and *condition2* evaluates to "false".
- Returns "false" if both *condition1* and *condition3* evaluate to "false".

Evaluation of the three arguments:

Condition 1	Condition 2	Condition 3	WHEN returns:
True	True		True
False			True
False		True	True
True	False		False
False		False	False

**Examples**

- WHEN (ABSENT(CatalogueField), ABSENT (QuantityField))

Returns "true" when **CatalogueField** and **QuantityField** are both absent

- WHEN (PRESENT(ShipDate), PRESENT(InStock), PRESENT(BackOrderDate ))

Returns "false" when **ShipDate** and **BackOrderDate** are both absent

**Lookup and reference functions****CHOOSE**

The CHOOSE function returns the object within a series whose position in the series corresponds to a specified number.

**Syntax:**

CHOOSE (series-object-name , single-integer-expression)

**Meaning:**

CHOOSE (from\_these\_objects , pick\_the\_nth\_one)

**Returns:**

A single object whose index within the from\_these\_objects series matches the number specified by pick\_the\_nth\_one. If that member of the series does not exist, CHOOSE returns "none".

You can use CHOOSE to use a variable value to specify the index for a particular object from a series. The CHOOSE function does not operate on a series object that was returned by another function, such as EXTRACT.

**Examples**

- CHOOSE ( Row:DBSelect , 2 )

Returns the second **Row**

- CHOOSE ( Set:Claim , INDEX ( Row:Header ) )

Returns the **Set** within the **Claim** that corresponds to the index of the **Row** of **Header**.

**Related function**

- LOOKUP

**DBLOOKUP**

The DBLOOKUP function executes an SQL statement against a database. The SQL statement can be any permitted by your database management system or ODBC driver.

When the DBLOOKUP function is used in a map, the default **OnSuccess** action is adapter specific. The default **OnFailure** action is to rollback any changes made during map processing. The default **Scope** will be integral unless the map is defined to run in bursts (which is the case when one or more inputs have the **FetchAs** property set to **Burst**).

There are two ways to specify arguments for DBLOOKUP.

You can use DBLOOKUP to execute an SQL statement when you want to execute a SELECT statement to retrieve a specific column value in a large table in a database using the value of another input, rather than defining the entire table as an input card and using the LOOKUP, SEARCHDOWN, or SEARCHUP functions.

You can use DBLOOKUP to execute an SQL statement when you want to execute a SELECT statement to retrieve a specific column value from a table or database that might vary based on a parameter file. Using Meaning 2 of the DBLOOKUP function allows these parameters to be dynamically specified at run time.

**Syntax:**

DBLOOKUP ( single-text-expression , single-text-expression , [ single-text-literal ] )

**Meaning:**

1. DBLOOKUP ( SQL\_statement , mdq\_filename , database\_name )
2. DBLOOKUP ( SQL\_statement , parameters )

**Returns:**

A single text item

The DBLOOKUP function returns the results of the query in the same format as a query specified for a map input card, except that it does not include the last carriage return/linefeed. Because this information is removed, it is easier to make use of a single value extracted from a database.

**Arguments for meaning 1**

DBLOOKUP ( SQL\_statement , mdq\_filename , database\_name )

- **SQL\_statement**

The first argument is an SQL statement as a text string. This can be any valid SQL statement permitted by your database management system and supported by your database-specific driver. In addition to a fixed SQL statement, this argument can be a concatenation of text literals and data objects, enabling the concatenation of data values into your SQL statement.

- **mdq\_filename**

The second argument is the name of a database query file (**.mdq**) produced by the Database Interface Designer. It contains the definition of the database that the SQL statement is to be executed against. If the **.mdq** file is in a directory other than the directory of the map, the path must be specified.



**Note:** The **.mdq** file is accessed at map build time and is not needed at run time.

- **database\_name**

The third argument is the name of a database in the database query file (**.mdq**) as defined in the Database Interface Designer.

If used in this way, both the **.mdq** filename and database name must be literals.

**Arguments for meaning 2**

DBLOOKUP ( SQL\_statement , parameters )

- **SQL\_statement**

The first argument is an SQL statement as a text string. This can be any valid SQL statement permitted by your database management system and supported by your database-specific driver. In addition to a fixed SQL

statement, this argument can be a concatenation of text literals and data objects, enabling the concatenation of data values into your SQL statement.

- **parameters**

The second argument is a set of parameters, either:

- `-MDQ mdqfilename -DBNAME dbname`
- or-
- `-DBTYPE database_type [database specific parameters]`

The keyword `-MDQ` is followed by the name of the database query file (**.mdq**) produced by the Database Interface Designer. This **.mdq** file contains the definition of the database. If the **.mdq** file is in a directory other than the directory of the map, the path must be specified. The **.mdq** filename is followed by the keyword `-DBNAME` and the database name as specified in the Database Interface Designer.

Using this syntax, the **.mdq** file is accessed at run time and must be present.

The keyword `-DBTYPE` is followed by a keyword specifying the database type (for example, ODBC or ORACLE) followed, optionally, by database-specific parameters.

This syntax does not use an **.mdq** file, because the database-specific parameters provide the information required to connect to the database. See the Resource Adapters documentation for detailed information about the database-specific parameters that can be specified.

When used with Meaning 2, DBLOOKUP must conform to these rules:

- All keywords (for example, `-DBTYPE`) can be upper or lowercase, but not mixed.
- A space is required between the keyword and its value (for example, `-DBTYPE ODBC`).
- The order of the keywords is not important.

All database-specific parameters are optional.

## Examples

Assume that you have a table named "PARTS" that contains the following data:

	<b>PART_NUMBER</b>	<b>PART_NAME</b>
1		1/4" x 3" Bolt
2		1/4" x 4" Bolt

Assume that this database has been defined using the Database Interface Designer in a file named **mytest.mdq** and that the name of the database, as specified in the **.mdq** file, is **PartsDB**.

- DBLOOKUP ( "SELECT PART\_NAME from PARTS where PART\_NUMBER =1", "mytest.mdq", "PartsDB")

Returns: ¼" x 3" Bolt

Using Meaning 2, you can specify the DBLOOKUP this way:

- DBLOOKUP( "SELECT PART\_NAME from PARTS where PART\_NUMBER =1", "-MDQ mytest.mdq -DBNAME PartsDB" )

where both the **.mdq** file name and database name is specified.

Using Meaning 2, you can also specify the database type and the appropriate database-specific parameters:

- DBLOOKUP( "SELECT PART\_NAME from PARTS where PART\_NUMBER =1", "-DBTYPE ORACLE -CONNECT MyDB -USER janes " )

## Related functions

- DBQUERY
- EXTRACT
- FAIL
- LASTERRORCODE
- LASTERRORMSG
- LOOKUP
- SEARCHDOWN
- SEARCHUP
- VALID

For more examples using the DBLOOKUP function, see the Database Interface Designer documentation.

## DBQUERY

The DBQUERY function executes an SQL statement against a database. The SQL statement can be any permitted by your database management system or ODBC driver.

When the DBQUERY function is used in a map, the default **OnSuccess** action is adapter specific. The default **OnFailure** action is to rollback any changes made during map processing. The default **Scope** will be integral unless the map is defined to run in bursts (which is the case when one or more inputs have the **FetchAs** property set to **Burst**).

There are two ways to specify the arguments for DBQUERY. You can use DBQUERY [*Meaning 1*] to execute an SQL statement when you want to look up information in a database using a parameterized query that is based on another value in your data. If your SQL statement is a SELECT statement, the DBQUERY function might be used in conjunction with the RUN function to issue dynamic SELECT statements whose results can be used as input to another map.

You can also use the DBQUERY function [*Meaning 2*] to execute an SQL statement when the database, table, or other database parameters might vary; perhaps being supplied by a parameter file.

**Syntax:**

```
DBQUERY (single-text-expression , single-text-expression ,
[ single-text-literal ] )
```

**Meaning:**

1. DBQUERY (SQL\_statement , mdq\_filename , database\_name)
2. DBQUERY ( SQL\_statement , parameters )

**Returns:**

A single text item

If your SQL statement is a SELECT statement, the results of the query in the same format as a query specified as a map input card, including row delimiters and terminators, and so on.

If your SQL statement is anything other than a SELECT statement, "none".

### Arguments for meaning 1

DBQUERY (SQL\_statement , mdq\_filename , database\_name)

- **SQL\_statement**

The first argument is an SQL statement as a text string. This can be any valid SQL statement that is permitted by your database management system and supported by your database-specific driver. In addition to a fixed SQL statement, this argument can be a concatenation of text literals and data objects, enabling the concatenation of data values into your SQL statement.

- **mdq\_filename**

The second argument is the name of a database query file (**.mdq**) produced by the Database Interface Designer. It contains the definition of the database that the SQL statement is to be executed against. If the **.mdq** file is in a directory other than the directory of the map, the path must be specified.



**Note:** The **.mdq** file is accessed at map build time and is not needed at run time.

- **database\_name**

The third argument is the name of a database in the database query file (**.mdq**) as defined in the Database Interface Designer.

If used in this way, both the **.mdq** filename and database name must be literals.



## Arguments for meaning 2

DBQUERY ( SQL\_statement , parameters )

- The first argument is an SQL statement as a text string. This can be any valid SQL statement that is permitted by your database management system and supported by your database-specific driver. In addition to a fixed SQL statement, this argument can be a concatenation of text literals and data objects, enabling the concatenation of data values into your SQL statement.
- The second argument is a set of parameters, either:
  - -MDQ mdqfilename -DBNAME dbname
  - or-
  - -DBTYPE database\_type [database specific parameters]

The keyword -MDQ is followed by the name of the database query file (**.mdq**) produced by the Database Interface Designer. This **.mdq** file contains the definition of the database. If the **.mdq** file is in a directory other than the directory of the map, the path must be specified. The **.mdq** filename is followed by the keyword -DBNAME and the database name as specified in the Database Interface Designer.



**Note:** Using this syntax, the **.mdq** file is accessed at run time and must be present.

The keyword -DBTYPE is followed by a keyword specifying the database type (for example, ODBC or ORACLE) followed, optionally, by database-specific parameters.



**Note:** This syntax does not use an **.mdq** file, because the database-specific parameters provide the information required to connect to the database. See the appropriate database adapter documentation for detailed information about database-specific parameters.

When used with Meaning 2, DBQUERY must conform to these rules:

- All keywords (for example, -DBTYPE) can be upper or lower case, but not mixed.
- A space is required between the keyword and its value (for example, -DBTYPE ODBC).
- The order of the keywords is not important.

All database-specific parameters are optional.

## Examples

Assume that you have a table named "PARTS" that contains the following data:

PART_NUMBER	PART_NAME
1	1/4" x 3" Bolt

PART_NUMBER	PART_NAME
2	1/4" x 4" Bolt

Also assume that this database has been defined using the Database Interface Designer in a file named **mytest.mdq** and that the name of the database, as specified in the **.mdq** file, is **PartsDB**.

```
DBQUERY ( "SELECT * from PARTS" , "mytest.mdq" , "PartsDB" )
```

Returns 1|¼" x 3" Bolt<cr><lf>2|¼" x 4" Bolt<cr><lf>

where <cr><lf> is a carriage return followed by a line feed.

Using Meaning 2, you can also specify the DBQUERY this way:

```
DBQUERY ( "SELECT * from PARTS" , "-MDQ mytest.mdq -DBNAME PartsDB" )
```

where both the **.mdq** file name and database name are specified.

Or, specify it this way, using Meaning 2 by specifying the database type and the appropriate database-specific parameters:

```
DBQUERY ( "SELECT * from PARTS" , "-DBTYPE ORACLE -CONNECT MyDB -USER janes" )
```

Assume that you have an input file containing one order record. To map that order to another proprietary format, you also have a parts table with pricing information for every part for every customer, a very large table. Rather than using the entire parts table as the input to your map, you might use the RUN function with a DBQUERY to dynamically select only those rows from the parts table corresponding to the customer in the order file, as follows:

```
RUN ( "MapOrder.MMC" ,
      "IE2" + DBQUERY ( "SELECT * FROM Parts WHERE CustID = "
      + CustomerNo:OrderRecord:OrderFile + " ORDER BY PartNo" ,
      "PartsDB.MDQ", "PartsDatabase" ) )
```

## Related functions

- DBLOOKUP
- EXTRACT
- FAIL
- LASTERRORCODE
- LASTERRORMSG
- LOOKUP
- SEARCHUP
- SEARCHDOWN
- VALID

## DDEQUERY

The DDEQUERY function allows you to interface to other Windows applications such as Trading Partner PC, Excel, and so forth, provided that certain criteria are met. For example, if you receive an Excel spreadsheet file, you must have

the appropriate version of the Excel application installed (that is compatible with the file received) and the application must be open.

**Syntax:**

DDEQUERY (single-text-expression , single-text-expression , single-text-expression)

**Meaning:**

DDEQUERY (application\_name , topic , text)

**Returns:**

A single text item from an application

**Examples**

- DDEQUERY ( "excel" , "[MKTPRICE.XLS]Sheet1" , "R8C1:R14C3" )

In this example, DDEQUERY is used to get data from an Excel spreadsheet. The third argument, **R8C1:R14C3**, specifies the location of the data in the spreadsheet. (In Excel, the 8<sup>th</sup> row, 1<sup>st</sup> column to the 14<sup>th</sup> row, 3<sup>rd</sup> column is A8:C14.) The content of this spreadsheet range is returned as a single text item.

This example assumes that the application, the map, and the spreadsheet all reside in the same directory. If they are not in the same directory you must add the path. For example:

DDEQUERY ( "excel" , "c:\spreadsheet[MKTPRICE.XLS]Sheet1" , "R8C1:R14C3" )

- DDEQUERY ("tppc","PartnerX","BGyourEDlode")

In this example, DDEQUERY is used as a request to Trading Partner PC.

**Related functions**

- EXIT
- FAIL
- GET
- LASTERRORCODE
- LASTERRORMSG
- PUT
- RUN
- VALID

**EXTRACT**

Use EXTRACT whenever you need only particular members of a series returned-those that meet a certain condition. An example might be only POs that contain back-ordered items.

The EXTRACT function can only be used in a map rule. It cannot be used in a component rule.

The EXTRACT function returns all members of a series for which a specified condition is true.

**Syntax:**

EXTRACT (series-object-expression, single-condition-expression)

**Meaning:**

EXTRACT (objects\_to\_extract, condition\_to\_evaluate)

**Returns:**

A series object.

The result is each member of *series\_to\_search* for which the condition specified by *condition\_to\_evaluate* evaluates to "true". EXTRACT returns "none", if no member of *series\_to\_search* has a corresponding *condition\_to\_evaluate* that evaluates to "true".

**Examples**

- EXTRACT ( PO:Transaction , Store# = Location:PO:Transaction )

This example returns all **POs**, individually, whose **Location** is a particular **Store#**.

- EXTRACT ( Row:DBSelect , ProcessFlag Column:Row:DBSelect = "Y" )

This example returns all **Rows** that have a **ProcessFlag Column** value of "Y".

**Related Functions**

- CHOOSE
- LOOKUP
- SEARCHDOWN
- SEARCHUP

**GETANDSET**

You can use GETANDSET when you have an input file that keeps track of control information that serves as an input to your map and the control information in the file needs to be updated based on processing that occurs in your map.

The GETANDSET function gets a fixed length value from the input data stream, updates that value in the input data stream, and returns either the original or updated value.

**Syntax:**

GETANDSET (single-fixed-size-item-object-name, single-item-expression, single-integer-expression)

**Meaning:**

GETANDSET (original\_value ,new\_value, integer\_that\_determines\_which\_value\_to\_return)

**Returns:**

A single-fixed-size-item and replaces the original value in the input data stream.

This function does not support decrementing integers past 0 into negative numbers.

GETANDSET updates an input by finding the object represented by *original\_value* and replacing it with the value represented by the *new\_value*. The function returns either the *original\_value* or the *new\_value*, depending on the value of the *integer\_that\_determines\_which\_value\_to\_return*. For example,

- If *integer\_that\_determines\_which\_value\_to\_return* has the value 1, *original\_value* is returned.
- If *integer\_that\_determines\_which\_value\_to\_return* has the value 2, *new\_value* is returned.
- If *integer\_that\_determines\_which\_value\_to\_return* has any other value, *original\_value* is returned.
- If one of the input arguments evaluates to "none", GETANDSET returns "none".

The original value specified for *new\_value* must be a fixed size item. During map execution, the original value is updated to reflect the evaluation of GETANDSET. When the source is a file, that file will be updated after map completion. However, if the source is a database, message, or application, the content of the source is in memory and the source, itself, is not updated.

## Examples

- New Tracking# = GETANDSET (Tracking#:Card, Tracking#:Card + 3, 1)

This rule finds the object **Tracking#:Card** (assume that it has the value 6), adds 3 to it, giving 9, and replaces the 6 with the 9 in the data location of **Tracking#:Card**. Subsequent references to **Tracking#:Card** will find its value to be 9. If **Card** is an input card whose source is a file, the file is rewritten with the new value.

Because the third input argument (*integer\_that\_determines\_which\_value\_to\_return*) is 1, the returned value is the original value of 6.

## GETDIRECTORY

The GETDIRECTORY function returns the full path (directory) for the compiled map file or the source or destination associated with a specified card object.

You can use GETDIRECTORY in a map rule or component rule when you need the full path (directory) of either the compiled map or of a data source or destination.

### Syntax:

```
GETDIRECTORY ([ single-simple-object-name ] )
```

### Meaning:

```
GETDIRECTORY ([ card_or_object_for_which_directory_is_needed ] )
```

### Returns:

A single text item

Without an argument, GETDIRECTORY returns the full path associated with the compiled map.

With an argument, the following occurs:

- From a map rule, the function returns the full path of the source or destination that is associated with the card. In a map rule, the argument must be the name of the card for which to get the directory.
- From a component rule, the function returns the full path of the source or destination that is associated with the active source or destination.

The GETDIRECTORY command without arguments will return the directory of the compiled map, regardless of whether it is used in a map rule or in a component rule.

## Examples

- GETDIRECTORY (OrderFile)

If the card **OrderFile** is associated with the data file, *install\_dir\order.txt*, GETDIRECTORY returns *\install\_dir\*.

- GETDIRECTORY ( )

If the compiled map on the HP-UX is */maps/prod/mymap.mmc*, GETDIRECTORY returns */maps/prod/*.

Suppose you want to run the map, **MyMap**, from a component rule on **Record** in your input to determine whether input customer names are valid. **MyMap** has two inputs and one output. The first input is the customer name to look up. The second input is a lookup file. The output is a text item whose value is "valid" or "error".

The name of the lookup file is constant; it is always **XREF\_TBL.TXT**. However, its location might vary; it will always be in the same directory as the data file used as the source you are trying to validate. For example, if the name of the data file used as the source is **C:\SHR\ABC\INPUT.TXT**, the lookup file name is **C:\SHR\ABC\XREF\_TBL.TXT**. If the data file name is */local/data/somefile*, the lookup file name is */local/data/XREF\_TBL.TXT*, and so forth.

You could use this component rule on **Record** to determine whether the customer name is valid:

```
RUN ( "MyMap.mmc" , "-IE1S10" + CustomerName:$ + " -IF2 " + GETDIRECTORY ( ) + "XREF_TBL.TXT" + "
-OE1" ) = "VALID"
```

## Related functions

- GETFILENAME
- GETRESOURCEName

## GETITXUID

The GETITXUID function returns a universally unique identifier (UUID).

### Syntax:

```
resourcelib->GETITXUID ( )
```

**Meaning:**

GETITXUID ()

**Returns:**

A unique string of 36 single-digit hexadecimal numbers

This function has no arguments but requires parentheses ().

## GETLOCALE

The GETLOCALE function returns the locale setting of the computer.

**Syntax:**

resourcelib->GETLOCALE ()

**Meaning:**

GETLOCALE ()

**Returns:**

A single text item

The GETLOCALE function returns the locale setting of the computer where the map runs. The locale is returned in the format specified by the operating system. Generally, the return format is an ISO Language Code (as defined by ISO-639) in combination with an ISO Country Code (as defined by ISO-3166) when applicable. The language codes are two lowercase letters and the country codes are two uppercase letters. For example, **en\_US** is the code for English (United States).

This function has no arguments but requires parentheses.

**Exemple**

**Table 15. Examples**

	<b>System locale</b>	<b>Returns</b>
French		fr
Korean		ko
Brazilian Portuguese		pt_BR
Taiwanese Chinese		zh-TW

## GETFILENAME

The GETFILENAME function returns the file name for a file adapter source or target of a specified card object. Without an argument, it returns the adapter command associated with the active source or destination.

You can use GETFILENAME in a map rule or component rule when you need the data file name.

**Syntax:**

```
GETFILENAME ( [ single-simple-object-name ] )
```

**Meaning:**

```
GETFILENAME ( [ card_for_which_resource_info_is_needed ] )
```

**Returns:**

A single text item

With an argument, GETFILENAME returns the file name for a file source or target of a specified card object. Without an argument, the function returns the source or destination name associated with the active source or destination.

**Examples**

- GETFILENAME (OrderFile)

If the card **OrderFile** is associated with the data file, *install\_dir\order.txt*, GETFILENAME returns *install\_dir\order.txt*.

Suppose you want to run the map **MyMap** from a component rule on **Record** in your input to determine if input customer names are valid. **MyMap** has two inputs and one output. The first input is the customer name to be looked up. The second input is a lookup file. The output is a text item whose value is "valid" or "error".

The name of the lookup file can vary; it must correspond to the name of the data file used as the source you are trying to validate. For example, if the data file name is **c:\DATA.AAA**, the lookup file name is **c:\LOOKUP.AAA**. If the data file name is **c:\DATA.XYZ**, the lookup file name is **c:\LOOKUP.XYZ**, and so on.

You could use this component rule on **Record** to determine whether the customer name is valid:

```
RUN ( "MyMap.mmc", "-IE1S10" + CustomerName:$ + " -IF2 LOOKUP." + RIGHT ( GETFILENAME ( ), 3 ) + "-OE1" ) = "VALID"
```

**Related functions**

- GETDIRECTORY
- GETRESOURCENAME

**GETPARTITIONNAME**

You can use GETPARTITIONNAME when you need to know the name of the partition to which the data for the given object belongs.

**Syntax:**

```
GETPARTITIONNAME (single-partitioned-object-expression)
```



**Meaning:**

GETPARTITIONNAME (partitioned\_object)

**Returns:**

A single simple object name

GETPARTITIONNAME returns a text item that represents the name of the partition to which the data for *partitioned\_object* belongs.

**Examples**

- GETPARTITIONNAME (Transaction:File)

The type defined by **Transaction** is a partitioned object that has three partitions: **Add**, **Delete**, and **Modify**. In this example, if the input data for **Transaction** belongs to the **Delete** partition, the GETPARTITIONNAME function returns "delete".

**Related functions**

- PARTITION

**GETRESOURCEALIAS**

The GETRESOURCEALIAS function returns the resource alias value specified in a Resource Registry resource name file (.mrn).

**Syntax:**

resourcelib->GETRESOURCEALIAS (single-text-expression, single-text-expression)

**Meaning:**

GETRESOURCEALIAS (single-text-expression, single-text-expression)

**Returns:**

A single text item

The GETRESOURCEALIAS function loads a Resource Registry resource configuration file and retrieves the specified alias value. The values are defined as part of the Global object in the .mrc file.

**Example**

In the below example the resource configuration file is defined as:

```
<?xml version="1.0" encoding="UTF-8"?>
<ResourceCfg>

  <Global>
    <ResourceFile ActiveVirtualServer="test">Company.mrn</ResourceFile>
```

```
</Global>
</ResourceCfg>
```

The resource name file is defined as:

```
<?xml version="1.0" encoding="UTF-8"?>
<MRN>
  <VirtualServerSet>
    <VirtualServer>test</VirtualServer>
  </VirtualServerSet>
  <Resource>
    <Name>company</Name>
    <Value Server="test" encrypt="OFF">ABC</Value>
  </Resource>
</MRN>
```

..when used as a part of this rule:

```
=resourceLib->GETRESOURCEALIAS("company.mrc", "%company%")
```

In this scenario, the return value is "ABC".

When an absolute path is not defined in the location, the default location is the map directory.

## GETRESOURCENAME

The GETRESOURCENAME function returns the adapter source or target command of a specified card object. Without an argument, it returns the adapter command associated with the active source or destination.

You can use GETRESOURCENAME in a component or map rule when you need to know the name of a source or destination.

### Syntax:

```
GETRESOURCENAME ([ single-simple-object-name ])
```

### Meaning:

```
GETRESOURCENAME ([ card_for_which_resource_info_is_needed ])
```

### Returns:

A single text item

With an argument, the source or destination name that is associated with the card. Without an argument, returns the source or destination name associated with the active source or destination.

## Examples

- GETRESOURCENAME (OrderFile)

If the card **OrderFile** is associated with the data file, *install\_dir\order.txt*, the GETRESOURCENAME function returns *install\_dir\order.txt*.

## Related functions

- GETDIRECTORY
- GETFILENAME

## GETTXINSTALLDIRECTORY

The GETTXINSTALLDIRECTORY function returns the HCL® OneTest™ Data product installation directory.

### Syntax:

```
resourcelib->GETTXINSTALLDIRECTORY ()
```

### Meaning:

```
GETTXINSTALLDIRECTORY ()
```

### Returns:

A single text item

GETTXINSTALLDIRECTORY returns the product installation directory.

This function has no arguments but requires parentheses.

This function cannot be used in a z/OS environment.

## INDEX

You can use the INDEX function when you need to select or test particular objects based on their occurrence, or to add a sequence number to output objects.

INDEX returns an integer that represents the index of an object relative to its nearest contained object, counting only valid objects.

INDEX cannot be used in a component rule.

### Syntax:

```
INDEX (single-object-name)
```

### Meaning:

```
INDEX (object_for_which_to_get_index)
```

### Returns:

A single integer

The *object\_for\_which\_to\_get\_index* variable must be a type name. The result is the index of *object\_for\_which\_to\_get\_index*.

- If *object\_for\_which\_to\_get\_index* is an input, this will be the index within all valid objects.
- If *object\_for\_which\_to\_get\_index* is an output, this will be the index within all objects (valid and invalid).
- Returns 0 if the input argument is "none".

The difference between INDEXABS and INDEX is that INDEXABS counts both valid and invalid instances, whereas INDEX counts only valid instances.

## Examples

- Message (s) = IF (INDEX (Message:Input) > 3, Message:Input, "none")

For example, there are five **Messages** in **Input**. The first three evaluations of this rule return "none". The fourth evaluation returns **Message[4]**. The fifth evaluation returns **Message[5]**.

- Invoice (s) = MyMap (Invoice Segment:Input, INDEX (\$))

For the first evaluation of **MyMap**, INDEX(\$)<sup>1</sup> is 1. For the second evaluation of **MyMap**, INDEX(\$)<sup>2</sup> is 2.

## Related functions

- CHOOSE
- COUNT
- COUNTABS
- INDEXABS

## INDEXABS

You can use INDEXABS when you need the absolute occurrence of a particular object across all occurrences, rather than across only valid occurrences.

The INDEXABS function returns an integer that represents the index of an object relative to its nearest contained object, counting both valid and invalid instances of the object.

### Syntax:

INDEXABS (single-object-name)

### Meaning:

INDEXABS (object\_for\_which\_to\_get\_index)

### Returns:

A single integer

INDEXABS returns an integer that represents the absolute index of *object\_for\_which\_to\_get\_index*. The integer indicates the instance this object that is in the set of *all* instances of the object, including both valid and invalid occurrences. Returns 0 if the input argument is "none".

- If *object\_for\_which\_to\_get\_index* is an input, this will be the index within all members of the series, including valid objects, invalid objects, and existing "none"s.
- If *object\_for\_which\_to\_get\_index* is an output, this will be the index within all existing members of the series, including existing "none"s.

The difference between INDEXABS and INDEX is that INDEXABS counts both valid and invalid instances, as well as existing "none"s, whereas INDEX counts only valid instances.

## Examples

- INDEXABS (Message Record:Order:PO\_File)

For this example, that **Order** contains the following **Messages**:

Message Record[1] Valid

Message Record[2] Error

Message Record[3] Valid

In a map rule, INDEXABS (MessageRecord[3]:Order:PO\_File) would evaluate to 3.

If the INDEX function was used, INDEX ( Message Record[3]:Order:PO ) would evaluate to 2.

## Related function

- INDEX

## LASTERRORCODE

The LASTERRORCODE function returns a text item whose value is the last error code returned by one of a specified set of functions during map execution.

You can use LASTERRORCODE to interrogate or report the error code returned by one of the external interface functions.

### Syntax:

LASTERRORCODE ( )

### Meaning:

LASTERRORCODE ( )

### Returns:

A single text item

LASTERRORCODE has no arguments but it requires parentheses.

LASTERRORCODE returns a text item whose value is the last error code returned by one of a specified set of functions during map execution.

The following functions can fail:

- DBLOOKUP
- DBQUERY
- DDEQUERY
- EXIT
- GET
- PUT
- RUN

## Examples

- Message = VALID ( RUN ( "Map1Msg.mmc" , "-AE -OMMSMQ1B `QN .\aqueue -CID 2001" ), FAIL ( "Failure on RUN (" + TEXT ( LASTERRORCODE ( ) ) + "):" + LASTERRORMSG ( ) ) )

In this example, the LASTERRORCODE and LASTERRORMSG functions are being used in conjunction with the FAIL and VALID functions to fail (abort) the map if the map executed by the RUN function (**Map1Msg.mmc**) fails. In this example, the map fails and returns the error code and error message reported by the RUN function using the LASTERRORCODE and LASTERRORMSG functions.

If **Map1Msg** fails because one or more of its inputs was invalid, **Message** is assigned a value of "none". The map aborts and the following message is reported in the execution audit log:

Failure on RUN (8): One or more inputs was invalid.

## Related functions

- DBLOOKUP
- DBQUERY
- DDEQUERY

## LASTERRORMSG

The LASTERRORMSG function returns a text item whose value is the message corresponding to the last error code returned by one of a specified set of functions during map execution.

### Syntax:

LASTERRORMSG ( )

### Meaning:

LASTERRORMSG ( )

**Returns:**

A single text item

Although LASTERRORMSG has no arguments, it does require parentheses.

LASTERRORMSG returns a text item whose value is the message corresponding to the last error code returned by one of a specified set of functions during map execution.

The following is the list of functions that can fail:

- DBLOOKUP
- DBQUERY
- DDEQUERY
- EXIT
- GET
- PUT
- RUN

**Examples**

```
• Message = VALID ( RUN ( "Map1Msg.mmc" , "-AE -OMMSMQ1B ^-QN .\aqueue -CID 2001'" ), FAIL ( "Failure
on RUN ( " + TEXT (LASTERRORCODE ( ) ) + "):" + LASTERRORMSG ( ) ) )
```

In this example, the LASTERRORCODE and LASTERRORMSG functions are being used in conjunction with the FAIL and VALID functions to fail (abort) the map if the map executed by the RUN function (**Map1Msg.mmc**) fails. In this example, the map fails and returns the error code and error message reported by the RUN function using the LASTERRORCODE and LASTERRORMSG functions.

If **Map1Msg** fails because one or more of its inputs was invalid, Message is assigned a value of "none". The map aborts and the following message is reported in the execution audit log:

Failure on RUN (8): One or more inputs was invalid.

**Related functions**

- DBLOOKUP
- DBQUERY
- DDEQUERY

## LOOKDOWN

The LOOKDOWN function sequentially searches a series beginning at the end of the series, returning the last member of the series that meets a specified condition.

### Syntax:

LOOKDOWN (series-object-expression , single-condition-expression)

### Meaning:

LOOKDOWN (series\_to\_search , condition\_to\_evaluate)

### Returns:

A single object

LOOKDOWN returns the last member of *series\_to\_search* for which *condition\_to\_evaluate* evaluates to "true"; it returns "none" if no member of *series\_to\_search* meets the condition specified by *condition\_to\_evaluate*.

## Examples

- LOOKDOWN (Account#:Customer , Company Name:Customer = "ACME")

This example returns the **Account#** of **Customer** whose **Company Name** is ACME.

- LOOKDOWN (Part#:Row:DBSelect , Model#:Row:DBSelect = ModelCode:Legacy & Serial#:Row:DBSelect > "123")

This example returns the **Part#** of **DBSelect** where the **Model#** in that row matches the **ModelCode** of **Legacy** and the **Serial#** is greater than 123.

## Related functions

- CHOOSE
- EXTRACT
- LOOKUP



**Note:** LOOKDOWN differs from LOOKUP in that LOOKDOWN returns the last member of *series\_to\_search* that meets the *condition\_to\_evaluate*, while LOOKUP returns the first member of *series\_to\_search* that meets the *condition\_to\_evaluate*.

- SEARCHUP
- SEARCHDOWN

The LOOKDOWN function performs a sequential search on an unsorted series, starting from the end of the series. The SEARCHUP and SEARCHDOWN functions perform a binary search on a series that is sorted in ascending or descending code-page byte-order. The SEARCHUP and SEARCHDOWN functions are efficient for



performing multiple searches on a large series. The LOOKDOWN function can be more efficient for performing fewer searches. Note that LOOKDOWN returns a different instance from the one returned by SEARCHUP and SEARCHDOWN if multiple instances in the series match the condition.

## LOOKUP

The LOOKUP function sequentially searches a series, returning the first member of the series that meets a specified condition.

### Syntax:

LOOKUP (series-object-expression , single-condition-expression)

### Meaning:

LOOKUP (series\_to\_search , condition\_to\_evaluate)

### Returns:

A single object

LOOKUP returns the first member of *series\_to\_search* for which *condition\_to\_evaluate* evaluates to "true"; it returns "none" if no member of *series\_to\_search* meets the condition specified by *condition\_to\_evaluate*.

## Examples

- LOOKUP (Account#:Customer , Company Name:Customer = "ACME")

This example returns the **Account#** of **Customer** whose **Company Name** is ACME.

- LOOKUP (Part#:Row:DBSelect , Model#:Row:DBSelect = ModelCode:Legacy & Serial#:Row:DBSelect > "123")

This example returns the **Part#** of **DBSelect** where the **Model#** in that row matches the **ModelCode** of **Legacy** and the **Serial#** is greater than 123.

## Related functions

- CHOOSE
- EXTRACT



**Note:** LOOKUP differs from EXTRACT in that LOOKUP returns the first member of *series\_to\_search* that meets the *condition\_to\_evaluate*, while EXTRACT returns all members (one at a time) of *series\_to\_search* that meet the *condition\_to\_evaluate*.

- LOOKDOWN



**Note:** LOOKUP differs from LOOKDOWN in that LOOKUP returns the first member of *series\_to\_search* that meets the *condition\_to\_evaluate*, while LOOKDOWN returns the last member of *series\_to\_search* that meets the *condition\_to\_evaluate*.

- SEARCHUP
- SEARCHDOWN

The LOOKUP function performs a sequential search on an unsorted series, starting from the beginning of the series. The SEARCHUP and SEARCHDOWN functions perform a binary search on a series that is sorted in ascending or descending code-page byte-order. The SEARCHUP and SEARCHDOWN functions are efficient for performing multiple searches on a large series. The LOOKUP function can be more efficient for performing fewer searches.

## MEMBER

Use MEMBER when you need to know whether an object occurs within a series.

The MEMBER function searches a series, looking for a single specified object in the series. If any object in the series matches the specified object, MEMBER returns "true". If there is no match, MEMBER returns "false".

### Syntax:

MEMBER (single-object-expression , series-object-expression)

MEMBER (single-object-expression , { literal, literal ... })

### Meaning:

MEMBER (object\_to\_look\_for , series\_of\_objects\_to\_look\_at)

### Returns:

"True" or "false"

MEMBER returns "true" if *object\_to\_look\_for* matches one of the values in *series\_of\_objects\_to\_look\_at*.

It returns "false" if *object\_to\_look\_for* does *not* match at least one of the values in *series\_of\_objects\_to\_look\_at*.

The two arguments, *object\_to\_look\_for* and *series\_of\_objects\_to\_look\_at*, must be objects of the same item interpretation or the same group type. For example, if *object\_to\_look\_for* is a date/time item, *series\_of\_objects\_to\_look\_at* must be a series of date/time items.

## Examples

- MEMBER (EntityIDCode:Name, {"BT", "ST"})

This example tests whether **EntityIDCode** has one of a particular set of literal values.

- MEMBER (Store# , EntityIDCode:Name)

This example tests whether **Store#** has the same value as any **EntityIDCode:Name**.

## Related functions

- EXTRACT
- LOOKUP

## SEARCHDOWN

Use the SEARCHDOWN function to look up data within a series of data that is sorted in descending code-page byte-order. SEARCHDOWN performs a binary search on the sorted series and returns a related object that corresponds to the item found.

The byte-order of characters varies by code page. For example, because the ASCII **A** character (0x41) has a higher numeric value than the ASCII **0** character (0x30), the SEARCHDOWN function requires the **0** character to be sorted after the ASCII **A** character in the series. The EBCDIC **A** character (0xC1) has a lower numeric value than the EBCDIC **0** character, so the SEARCHDOWN function requires the EBCDIC **A** character to be sorted be after the EBCDIC **0** character in the series.

The HCL® OneTest™ Data Data Language property values change the collation order of some characters. For example, the German National Language property value is arranged in byte order, with the exception of certain characters: æ is ordered after a, e is ordered before ä, and so forth. There are particular collation rules for other National Language property values. However, there is no collation order for "Western" and "Japanese" property values, and the deprecated Data Language property values are always in byte order.

### Syntax:

```
SEARCHDOWN (series-object-expression , series-item-object-expression , single-item-expression)
```

### Meaning:

```
SEARCHDOWN (corresponding_object_to_return , descending_items_to_search , item_to_match)
```

### Returns:

A single object

SEARCHDOWN performs a binary search on the item series of *descending\_items\_to\_search*. The *descending\_items\_to\_search* must be sorted in descending code page byte-order. The value to search for is specified as the *item\_to\_match*. The object returned (*corresponding\_object\_to\_return*) must be related to *descending\_items\_to\_search* by a common object name.

If no match is found, SEARCHDOWN returns "none".

## Examples

- SEARCHDOWN ( Age Column:Row:DBSelect , SSN Column:Row:DBSelect , SSN\_Value:Message )

If there are ten rows in **DBSelect**, the search starts by comparing the first **SSN Column** of the fifth row with the **SSN\_Value** in **Message**. If the result matches, SEARCHDOWN returns the first **Age Column** of that **Row**. If the value of **SSN Column** is less than the **SSN\_Value** in **Message**, the search continues with the third **Row**. If the value of **SSN Column** is greater than the **SSN\_Value** in **Message**, the search continues with the seventh Row in **DBSelect**. The search continues in this fashion until either a match is found or until one **Row** is selected. If there is more than one **SSN Column** for the selected **Row**, a similar search is initiated for all **SSN Column**'s for the selected **Row** in **DBSelect**.

SEARCHDOWN returns the first **Age Column** for the selected **Row** of **DBSelect**.

## Related functions

- EXTRACT
- LOOKUP
- SEARCHUP

## SEARCHUP

Use the SEARCHUP function to look up data within a series of data that is sorted in ascending code-page byte-order. The SEARCHUP function performs a binary search on the sorted series and returns a related object that corresponds to the item found.

The byte-order of characters varies by code page. For example, because the ASCII **A** character (0x41) has a higher numeric value than the ASCII **0** character (0x30), the SEARCHUP function requires the **0** character to be sorted before the ASCII **A** character in the series. The EBCDIC **A** character (0xC1) has a lower numeric value than the EBCDIC **0** character, so the SEARCHUP function requires the EBCDIC **A** character to be sorted before the EBCDIC **0** character in the series.

The HCL® OneTest™ Data Data Language property values change the collation order of some characters. For example, the German National Language property value is arranged in byte order, with the exception of certain characters: æ is ordered after a, e is ordered before ä, and so forth. There are particular collation rules for other National Language property values. However, there is no collation order for "Western" and "Japanese" property values, and the deprecated Data Language property values are always in byte order.

### Syntax:

```
SEARCHUP (series-object-expression, series-item-object-expression, single-item-expression)
```

### Meaning:

```
SEARCHUP (corresponding_object_to_return, ascending_items_to_search, item_to_match)
```

**Returns:**

A single object

SEARCHUP performs a binary search on the item series of *ascending\_items\_to\_search*. The *ascending\_items\_to\_search* must be sorted in ascending code page byte-order. The value to search for is specified as the *item\_to\_match* and must be of the same type as the *ascending\_item\_to\_search*.

The object returned (*corresponding\_object\_to\_return*) must be related to *ascending\_items\_to\_search* by a common object name. If no match is found, SEARCHUP returns "none".

**Examples**

- SEARCHUP (Age Column:Row:DBSelect, SSN Column:Row:DBSelect, SSN\_Value:Message)

If there are ten rows in **DBSelect**, the search starts by comparing the first **SSN Column** of the fifth **Row** with the **SSN\_Value** in **Message**. If the result matches, SEARCHUP returns the first **Age Column** of that **Row**. If the value of **SSN Column** is greater than the **SSN\_Value** in **Message**, the search continues with the third **Row**. If the value of **SSN Column** is less than the **SSN\_Value** in **Message**, the search continues with the seventh **Row** in **DBSelect**. The search continues in this fashion until either a match is found or until one **Row** is selected. If there is more than one **SSN Column** for the selected **Row**, a similar search is initiated for all **SSN Column**'s for the selected **Row** in **DBSelect**.

SEARCHUP returns the first **Age Column** for the selected **Row** of **DBSelect**.

**Related functions**

- EXTRACT
- LOOKUP
- SEARCHDOWN

**SORTDOWN**

Use the SORTDOWN function to sort objects in a series in descending sequence. The function returns a series containing the values from the input series in descending order.

- When the character set of the data is ASCII, the objects are sorted in ASCII descending order.
- For all other character sets, the objects are sorted in binary descending order.

**Syntax:**

SORTDOWN (series-item-expression)

**Meaning:**

SORTDOWN (item\_series\_to\_sort)

**Returns:**

A series object

Returns the values in *item\_series\_to\_sort* in descending order.

**Examples**

- SORTDOWN ( Abbr:File )

In this example, if **Abbr** has these ASCII values:

ABC, GHI, DEF

SORTDOWN returns these values as a series as:

GHI, DEF, ABC

- The following table displays the results of using the SORTDOWN and SORTUP functions for a typical series of ASCII text items.

Original Input Series	Result using SORTDOWN	Result using SORTUP
Clams Casino	shrimps	1 Shrimp
Grouper	raw oysters	22 Shrimp
groupers	oysters	A
Shrimp	lobster tails	A 1 A shrimp
shrimps	lobster	A1A Shrimp
lobster	groupers	AA
lobster tails	clams	AAAAA
oysters	aaaaa	Clams Casino
raw oysters	aa	Grouper
clams	a	Rock Lobster
SHRIMP	Snapper	SHRIMP
1 Shrimp	Snapper	Shark Fin Soup
22 Shrimp	Shrimp	Shrimp
A 1 A shrimp	Shark Fin Soup	Snapper
A1A Shrimp	SHRIMP	Snapper
AAAAA	Rock Lobster	a
aaaaa	Grouper	aa

Original Input Series	Result using SORTDOWN	Result using SORTUP
aa	Clams Casino	aaaaa
AA	AAAAA	clams
A	AA	groupers
a	A1A Shrimp	lobster
Rock Lobster	A 1 A shrimp	lobster tails
Snapper	A	oysters
Snapper	22 Shrimp	raw oysters
Shark Fin Soup	1 Shrimp	shrimps

### Related functions

- SEARCHDOWN
- SEARCHUP
- SORTUP
- UNIQUE

### SORTDOWNBY

Use SORTDOWNBY to sort the objects of a series based on a component that the objects contain. The SORTDOWNBY function returns the input series, sorted in descending order by the value of the specified component.

- When the character set of the data is ASCII, the objects are sorted in ASCII descending order.
- For all other character sets, the objects are sorted in binary descending order.

The SORTDOWNBY function does not sort the output of the EXTRACT, REJECT, or UNIQUE functions.

#### Syntax:

```
SORTDOWNBY (series-item-expression, single-item-expression)
```

#### Meaning:

```
SORTDOWNBY (group_series_to_sort, group_component_to_sort_by)
```

#### Returns:

A series object

SORTDOWNBY returns a series of instances of a group, sorted in descending order by the value of an item that the group contains.

### Example

In this example, the group **Soldier** contains the items:

- Name
- Rank
- Serial number

The following command returns the instances of Soldier, sorted in descending order by Rank:

```
SORTDOWNBY (Soldier:Army, Rank:Soldier:Army)
```

## Related functions

- SEARCHDOWN
- SEARCHUP
- SORTDOWN
- SORTUP
- SORTUPBY

## SORTUP

Use SORTUP to sort the objects of a series in ascending sequence. The SORTUP function returns a series containing the values from an input series in ascending order.

- When the character set of the data is ASCII, the objects are sorted in ASCII ascending order.
- For all other character sets, the objects are sorted in binary ascending order.

### Syntax:

```
SORTUP ( series-item-expression )
```

### Meaning:

```
SORTUP ( item_series_to_sort )
```

### Returns:

A series object

SORTUP returns the values in *item\_series\_to\_sort* in ascending order.

## Examples

- SORTUP ( Abbr:File )

In this example, if **Abbr** had the ASCII values ABC, GHI, DEF, the SORTUP function would return these values as a series: ABC, DEF, GHI.

## Related functions

- SEARCHDOWN
- SEARCHUP



- SORTDOWN
- UNIQUE

## SORTUPBY

Use SORTUPBY to sort the objects of a series based on a component that the objects contain. The SORTUPBY function returns the input series, sorted in ascending order by the value of the specified component.

- When the character set of the data is ASCII, the objects are sorted in ASCII ascending order.
- For all other character sets, the objects are sorted in binary ascending order.

The SORTUPBY function does not sort the output of the EXTRACT, REJECT, or UNIQUE functions.

### Syntax:

```
SORTUPBY (series-item-expression, single-item-expression)
```

### Meaning:

```
SORTUPBY (group_series_to_sort, group_component_to_sort_by)
```

### Returns:

A series object

SORTUPBY returns a series of instances of a group, sorted in ascending order by the value of an item that the group contains.

## Example

In this example, the group **Soldier** contains the items:

- Name
- Rank
- Serial number

The following command returns the instances of Soldier, sorted in ascending order by Rank:

```
SORTUPBY (Soldier:Army, Rank:Soldier:Army)
```

## Related functions

- SEARCHDOWN
- SEARCHUP
- SORTUP
- SORTDOWN
- SORTDOWNBY

## UNIQUE

The UNIQUE function returns a series containing all "unique" members of a series.

UNIQUE can only be used in a map rule, not in a component rule.

**Syntax:**

UNIQUE ( series-object-expression )

**Meaning:**

UNIQUE ( series\_to\_evaluate )

**Returns:**

A series object

UNIQUE evaluates to a series containing all unique members of *series\_to\_evaluate* and evaluates to "none" if *series\_to\_evaluate* evaluates to "none".

### Examples

- UNIQUE ( PartNumber:Inventory:File )

Returns the unique **PartNumbers** in **Inventory:File**

- COUNT ( UNIQUE ( Customer:Order:File ) )

Returns the number of unique **Customers** in **Order:File**

### Related functions

- SORTDOWN
- SORTUP

## GETENV

The GETENV function returns the value of the environment variable. If no variable exists, an empty string is returned.

**Syntax:**

GETENV(*single-text-expression*)

**Meaning:**

GETENV (the environment variable name)

**Returns:**

A single-text-item

---

Related reference

[GETOSNAME on page 903](#)

## GETOSNAME

The GETOSNAME function returns the name of the operating system.

The returned values can be:

- Windows for any version of Microsoft Windows operating system
- zLinux for Linux operating system on z/OS hardware
- Linux for Linux operating system on Intel hardware
- AIX for AIX operating system on PowerPC hardware
- OS/390 for native z/OS operating system

### Syntax:

```
GETOSNAME()
```

### Meaning:

```
GETOSNAME ()
```

### Returns:

A single-text-item

---

Related reference

[GETENV on page 902](#)

[GETFILENAME on page 883](#)

## Math and statistics functions

Precede all MATHLIB functions with

```
mathlib->
```

## ABS

The ABS function returns the absolute value of a number.

### Syntax:

```
ABS (single-number-expression)
```

### Meaning:

```
ABS (number)
```

**Returns:**

A single number; the absolute value of a number

**Examples**

- ABS (-3)

Returns 3

- ABS (3)

Returns 3

- AvailableCredit has a value of -69.42

ABS (AvailableCredit) returns 69.42

- FlexDollars has a value of 50

ABS ((100 - FlexDollars)/2) returns 25

## ACOSINE

Use the ACOSINE function to calculate the arccosine of a value.

**Syntax:**

```
mathlib->ACOSINE (single-number-expression)
```

**Meaning:**

```
mathlib->ACOSINE (number_to_convert)
```

**Returns:**

A single number item

The result is the arccosine of the converted value.

## ASIN

Use the ASIN function to calculate the arcsine of a value.

**Syntax:**

```
mathlib->ASIN (single-number-expression)
```

**Meaning:**

```
mathlib->ASIN (number_to_convert)
```

**Returns:**

A single number item

The result is the arcsine of the converted value.

## ATAN

The ATAN function calculates the arctangent of a value.

**Syntax:**

```
mathlib->ATAN (single-number-expression)
```

**Meaning:**

```
mathlib->ATAN (number_to_convert)
```

**Returns:**

A single number item

## ATAN2

The ATAN2 function calculates the arctangent of y,x.

**Syntax:**

```
mathlib->ATAN2 (single-number-expression, single-number-expression)
```

**Meaning:**

```
mathlib->ATAN2 (number_to_convert)
```

**Returns:**

A single number item

## COSINE

The COSINE function calculates the cosine of a value.

**Syntax:**

```
mathlib->COSINE (single-number-expression)
```

**Meaning:**

```
mathlib->COSINE (number_to_convert)
```

**Returns:**

A single number item

## COSINEH

The COSINEH function calculates the hyperbolic cosine of a value.

### Syntax:

```
mathlib->COSINEH (single-number-expression)
```

### Meaning:

```
mathlib->COSINEH (number_to_convert)
```

### Returns:

A single number item

## COUNT

You can use the COUNT function to return an integer representing the number of valid input or output objects in a series.

### Syntax:

```
COUNT (series-object-expression)
```

### Meaning:

```
COUNT (valid_objects_to_count)
```

### Returns:

A single integer

The result is the number of *valid\_objects\_to\_count*. If the input argument evaluates to "none", COUNT returns 0.

COUNT does not count existing "none"s unless its group was defined as an explicit format with a **Track** setting of Places.

## Examples

- COUNT ( Claim Record:Patient File )

This example returns the number of valid **Claim Record** objects in **Patient File**.

- COUNT ( Class IN Transcript )

This example returns the number of valid **Class** objects in **Transcript**.

- COUNT ( UNIQUE ( Class IN Transcript ) )

This example returns the number of valid **Unique Class** objects in **Transcript**.

## Related functions

- COUNTABS

## COUNTABS

You can use COUNTABS to count the input or output objects in a series, regardless of the validity of the object.

Unlike COUNT, COUNTABS includes both valid and invalid objects in a series.

### Syntax:

COUNTABS (series-object-expression)

### Meaning:

COUNTABS (objects\_to\_count)

### Returns:

A single-integer

The result is the number of *objects\_to\_count*. If the input argument evaluates to "none", COUNTABS returns 0.

COUNTABS does not count existing "none's" unless its group was defined as an explicit format with a **Track** setting of **Places**.

## Examples

- COUNTABS ( Claim Record:Patient File )

This example returns the number of **Claim Record** objects in **Patient File**.

- COUNTABS ( Class IN Transcript )

This example returns the number of **Class** objects in **Transcript**.

## Related functions

- COUNT

## EXP

The EXP function calculates the exponential of a value.

### Syntax:

`mathlib->EXP` (single-number-expression)

### Meaning:

`mathlib->EXP` (number\_to\_convert)

**Returns:**

A single number item

## FACTORIAL

The FACTORIAL function calculates the factorial of a value.

**Syntax:**

```
mathlib->FACTORIAL (single-number-expression)
```

**Meaning:**

```
mathlib->FACTORIAL (number_to_convert)
```

**Returns:**

A single number item

## FROMBASETEN

You can use FROMBASETEN when you need to convert numbers to a base other than 10.

The FROMBASETEN function converts an integer to a text item that can be interpreted as a number, using positional notation of the base specified.

**Syntax:**

```
FROMBASETEN (single-integer-expression, single-integer-expression)
```

**Meaning:**

```
FROMBASETEN (positive_integer_to_convert, base_to_convert_to)
```

**Returns:**

A single text item

FROMBASETEN returns a text item that results from converting *positive\_integer\_to\_convert* to a text item that can be interpreted as a number using positional notation of the base specified by *base\_to\_convert\_to*.

If *base\_to\_convert\_to* is less than 2 or greater than 36, FROMBASETEN evaluates to "none". Resulting text item characters A-Z are interpreted as digits having decimal values from 10-35, respectively. The characters returned are uppercase.

### Example

- FROMBASETEN (18, 2)

Returns the value 10010



- FROMBASETEN (123, 8)

Returns the value 173

### Related function

- TOBASETEN

## INT

You can use the INT function when you need only the integer portion of a number.

#### Syntax:

INT (single-number-expression)

#### Meaning:

INT (number\_to\_convert)

#### Returns:

A single integer

INT returns the integer portion of a number. The result is the integer part of *number\_to\_convert*. Any fractional part after the decimal point is dropped.

### Examples

- INT (1.45)

Returns 1

- INT (3.6)

Returns 3

- INT (Purchase:Amt - Discount:Amt)

Subtracts **Discount:Amt** from **Purchase:Amt** and returns the result as a whole number.

### Related functions

- MOD
- ROUND
- TRUNCATE

## LOG

The LOG function calculates the logarithms of a value.

**Syntax:**

`mathlib->LOG (single-number-expression)`

**Meaning:**

`mathlib->LOG (number_to_convert)`

**Returns:**

A single number item

## LOG10

The LOG10 function calculates the logarithms for base 10 of a value.

**Syntax:**

`mathlib->LOG10 (single-number-expression)`

**Meaning:**

`mathlib->LOG10 (number_to_convert)`

**Returns:**

A single number item

## MAX

The MAX function returns the maximum value from a series of number, date, time, or text values.

**Syntax:**

`MAX (series-item-expression)`

**Meaning:**

`MAX (series_of_which_to_find_max)`

**Returns:**

A single number

The result is the maximum value in the input argument series: number, text, or date/time.

## Examples

- `MAX ( UnitPrice:Input )`

If the values for **UnitPrice** are {20, 10, 100}, MAX returns 100.

- `MAX(EXTRACT( DueDate:Book:Library, CheckedOut:Book:Library = "Y"))`

Returns the maximum (latest) **DueDate** for a book that is checked out from the library.

## Related functions

- MIN

## MIN

Use MIN when you need the minimum value from a series of number, date, time, or text values.

The MIN function returns the minimum value from a series.

### Syntax:

MIN (series-item-expression)

### Meaning:

MIN (series\_of\_which\_to\_find\_min)

### Returns:

A single number

The result is the minimum value of the input series: number, text, or date/time.

## Examples

- MIN (UnitPrice:Input)

If the values for **UnitPrice** are {20,10,100}, MIN returns 10.

- MIN (StartTime:::Schedule)

Returns the minimum (earliest) **StartTime** in **Schedule**.

## Related functions

- MAX

## MOD

Use MOD when you need the modulus of an integer and a number.

The MOD function returns the modulus that remains after a number is divided by an integer.

### Syntax:

MOD (single-number-expression, single-integer-expression)

### Meaning:

MOD (dividend , divisor)

### Returns:

A single integer

The result is the remainder (modulus) after *dividend* is divided by *divisor*. The result has the same sign as *divisor*.

The *dividend* is first divided by the integer *divisor*, resulting in a quotient. The modulus is calculated by multiplying the integer portion of the quotient by *divisor* and then subtracting that product from *dividend*.

If *divisor* is 0, MOD returns "none".

## Examples

- MOD (3, 2) or MOD (-3, 2)

Returns 1

- MOD (3, -2) or MOD (-3, -2)

Returns -1

- MOD (-3, 2) or MOD (-3, -2)

Returns 1

- MOD (-3, -2) or MOD (-3, -2)

Returns -1

## POWER

The POWER function calculates x raised to the power of y.

### Syntax:

```
mathlib->POWER (single-number-expression, single-number-expression)
```

### Meaning:

```
mathlib->POWER (base_number, exponent_number)
```

### Returns:

A single number item

## RAND

The RAND function returns a pseudorandom number.

### Syntax:

```
RAND ()
```

### Meaning:

```
RAND ()
```

**Returns:**

A single number item

RAND takes no parameters. If the SEED function is called prior to RAND, RAND will return the same value based on the value to SEED.

**Related functions**

- RANDDATA
- SEED

**ROUND**

The ROUND function rounds a number to a specified number of decimal places. If the number of decimal places is not specified, the number is rounded to a whole number. The result is in character number format.

**Syntax:**

ROUND (single-number-expression [ , single-integer-expression ])

**Meaning:**

ROUND (number\_to\_round [ , number\_of\_decimal\_places ])

**Returns:**

A single number

ROUND converts *number\_to\_round* to character format, if necessary, and then produces the value of *number\_to\_round* rounded to the number of decimal places specified by *number\_of\_decimal\_places*. If *number\_of\_decimal\_places* is not specified, *number\_to\_round* is rounded to the nearest whole number.

**Examples**

- ROUND (1.46, 1)

Returns 1.5

- ROUND (1.46)

Returns 1

**Related functions**

- TRUNCATE

**SEED**

The SEED function primes the RAND and RANDDATA functions so they produce repeated results.

**Syntax:**

SEED(*single-number-expression*)

**Meaning:**

SEED(*number\_for\_seed*)

**Returns:**

NONE

**Related functions**

- RAND
- RANDDATA

## SIN

The SIN function calculates the sine of a value.

**Syntax:**

`mathlib->SIN` (*single-number-expression*)

**Meaning:**

`mathlib->SIN` (*number\_to\_convert*)

**Returns:**

The value of a number

## SINH

The SINH function calculates the hyperbolic sine of a value.

**Syntax:**

`mathlib->SINH` (*single-number-expression*)

**Meaning:**

`mathlib->SINH` (*number\_to\_convert*)

**Returns:**

A single number item

## SQRT

The SQRT function returns the square root of a number.

**Syntax:**

SQRT ( single-number-expression )

**Meaning:**

SQRT ( number )

**Returns:**

A single number

SQRT returns the square root of *number*.

**Examples**

- SQRT ( 4 )

Returns 2

**SUM**

The SUM function calculates the sum of a series of numbers.

**Syntax:**

SUM ( series-number-expression )

**Meaning:**

SUM ( series\_to\_sum )

**Returns:**

A single number

SUM returns the sum of all members in *series\_to\_sum*.

**Examples**

- SUM ( Quantity:LineItem )

This example calculates the sum of all the **Quantity** objects of **LineItem**.

**TAN**

The TAN function calculates the tangent of a value.

**Syntax:**

`mathlib->`TAN (single-number-expression)

**Meaning:**

`mathlib->`TAN (number\_to\_convert)

**Returns:**

A single number item

## TANH

The TANH function calculates the hyperbolic tangent of a value.

**Syntax:**

```
mathlib->TANH (single-number-expression)
```

**Meaning:**

```
mathlib->TANH (number_to_convert)
```

**Returns:**

A single number item

## TOBASETEN

The TOBASETEN function converts a text item that can be interpreted as a number, using positional notation of the base specified, to a base 10 number.

**Syntax:**

```
TOBASETEN ( single-text-expression, single-integer-expression )
```

**Meaning:**

```
TOBASETEN ( text_to_convert, base_to_convert_from )
```

**Returns:**

A single integer

TOBASETEN returns a number that results from converting *text\_to\_convert* that can be interpreted as a number, using positional notation of the base specified by *base\_to\_convert\_from*, to its base 10 representation. Text item characters A-Z are interpreted as digits having decimal values from 10-35, respectively.

If *base\_to\_convert\_from* is less than 2 or greater than 36, TOBASETEN evaluates to "none". If *text\_to\_convert* contains a character that is not alphanumeric or is not in the range specified by *base\_to\_convert\_from*, TOBASETEN returns "none".

### Examples

- TOBASETEN ( "A", 16 )

Returns the value 10

- TOBASETEN ( "10", 36 )

Returns the value 36



- TOBASETEN ( "A0", 15 )

Returns the value 150

- TOBASETEN ( "A0", 5 )

Returns the value "none"

### Related functions

- FROMBASETEN

## TRUNCATE

The TRUNCATE function removes decimal places from a number, leaving a specified number of decimal places.

You can use TRUNCATE with a second argument to truncate a number to a specified number of decimal places or without a second argument to reduce a number to an integer by removing all decimal places.

#### Syntax:

TRUNCATE (single-number-expression[ , single-integer-expression ] )

#### Meaning:

TRUNCATE (number\_to\_truncate[ , number\_of\_decimal\_places ] )

#### Returns:

A single number

TRUNCATE first converts *number\_to\_truncate* to character format, if necessary. It then truncates that number by removing decimal places to the right of *number\_of\_decimal\_places*. If *number\_of\_decimal\_places* is not used, the number is truncated to an integer.

### Examples

- TRUNCATE ( 3.9292, 2 )

Returns 3.92

- TRUNCATE ( 3.9292 )

Returns 3

### Related functions

- INT
- ROUND

## Text functions

## BCDTOTEXT

The BCDTOTEXT function converts the digits in a BCD (Binary Coded Decimal) item to a text item containing the digits of the BCD-encoded item as a string of characters.

**Syntax:**

BCDTOTEXT (single-text-expression)

**Meaning:**

BCDTOTEXT (BCD\_item\_to\_convert)

**Returns:**

A single text item

*BCD\_item\_to\_convert* is converted from BCD format to a text string containing the digits of the BCD-encoded value as a string of characters.

Numbers in BCD format have two decimal digits in each byte. Each half-byte, therefore, can contain a binary value from 0000 (which represents the digit 0) through 1001, which represents the digit 9). Based on this definition, the following behavior applies:

- If any half-byte of the BCD number contains the binary value 1101 or 1111, that half-byte is ignored.
- If the BCD item contains the binary value 1010, 1011, 1100, or 1110, the output of the function is "none".

### Examples

- BCDTOTEXT ( Qty:Item )

If **Qty** is x`1234`, the result is 1234.

- BCDTOTEXT ( DiscountAmt )

If **DiscountAmt** is x`0123`, the result is 0123.

- BCDTOTEXT ( TotalDollars )

If **Total** is x`F123`, the result is 123.

### Related functions

- BCDTOHEX
- BCDTOINT
- TEXTTOBCD

## COUNTSTRING

You can use the COUNTSTRING function when you need to know the number of times a specific text string appears within another text string. The function begins to look for the character string from the first position of the first string, and proceeds forward one byte at a time.

### Syntax:

COUNTSTRING (single-text-expression , single-text-expression)

### Meaning:

COUNTSTRING (text\_to\_search , text\_to\_find\_and\_count)

### Returns:

A single-integer

COUNTSTRING returns an integer that represents the number of times that a specified character string appears in another character string.

The result is a number representing the number of times *text\_to\_find\_and\_count* appears within *text\_to\_search*. If either *text\_to\_search* or *text\_to\_find\_and\_count* evaluates to "none", COUNTSTRING returns 0.

## Examples

- COUNTSTRING ( "banana" , "a" )

Returns a value of 3

- COUNTSTRING ( "aaaa" , "aa" )

Returns a value of 3.

## Related functions

- FIND
- LEFT
- MID
- RIGHT

## CPACKAGE

CPACKAGE specifies the character set of the output of the function. From that point onward, the data is treated as if it were in that character set. If the data is not in the specified character set, you get the wrong answer.

The character set is required to be specified in this function. If you choose not to specify a character set, you should use the original version of the PACKAGE function.

### Syntax:

CPACKAGE (single-object-expression , "character-set-of-object-content")

**Meaning:**

CPACKAGE (object\_to\_convert, object\_character\_set)

**Returns:**

A single text item

The second argument, *object\_character\_set*, represents the character set of the resulting object. Character set codes are listed in [Character set codes on page 978](#).

**Examples**

In this example, the group **Record** has an initiator of "#", a terminator of "@" and a delimiter of "," with the following data:

```
"#1339X10A,491.38,Green,42x54@"
```

- CPACKAGE (Record:Card, "ASCII")

Returns: #1339X10A,491.38,Green,42x54@

## CSERIESTOTEXT

CSERIESTOTEXT specifies the character set of the output of the function. From that point onward, the data is treated as if it were in that character set. If the data is not in the specified character set, you get the wrong answer.

The character set is required to be specified in this function. If you choose not to specify a character set, you should use the original version of the SERIESTOTEXT function.

**Syntax:**

CSERIESTOTEXT (series-object-expression , "character-set-of-object-content")

**Meaning:**

CSERIESTOTEXT (series\_to\_convert, object\_character\_set )

**Returns:**

A single text item

The *series\_to\_convert* argument concatenates the series of the input argument, including nested delimiters but excluding initiators and terminators.

The second argument, *object\_character\_set*, represents the character set of the resulting object.

**Examples**

In this example, you have the following data that represents bowler information for a bowling league:

```
Andrews, Jessica:980206:JBC:145:138:177:159
```

Little, Randy:980116:BBK:175:168

Wayne, Richard:980102:JBC:185:204:179:164:212

Each record consists of the bowler's name, the date of their last game played, a team code and one or more bowling scores. **Record** is defined as a group that is infix delimited by a colon.

Using the following rule produces results of the concatenation of all scores for all of the bowlers, even though the scores are not all contiguous within the data.

- = CSERIESTOTEXT (Score Field: Bowler:Input, "ASCII")

Returns: 145138177159175168185204179164212

You can change the rule to concatenate the list of scores to the bowler's name using the following rule:

- = BowlerName Field: Bowler:Input + " ->" + CSERIESTOTEXT (Score Field: Bowler:Input, "ASCII")

Returns:

Andrews, Jessica -> 145138177159

Little, Randy -> 175168

Wayne, Richard -> 185204179164212

In this example, you have an input number that is of variable size, followed by a name. There is no syntax that separates the number from the name. You can define the number as a group with **Byte(s)** as a component and provide a component rule for **Byte(s)**, such as:

ISNUMBER (\$)

## CSIZE

The CSIZE function returns an integer representing the size of a specified object in characters, exclusive of any pad characters.

### Syntax:

CSIZE (single object expression)

### Meaning:

CSIZE (object whose size is needed)

### Returns:

A single integer

### Exemple

### Example

**Example**

This example shows how the CSIZE function differs from the SIZE function.

**Example**

The SIZE function returns the number of bytes used to represent those symbolic characters in a particular code page.

**Example**

The CSIZE function returns the number of symbolic characters.

**Example**

Symbolic text: `Mañana`

**Example**

UTF-8 hexadecimal representation : `0x4D 0x61 0xC3 0xA3 0x61 0x6E 0x61`

**Example**

`SIZE(Mañana)`

**Example**

Returns 7 bytes

**Example**

`CSIZE(Mañana)`

**Example**

Returns 6 characters

## CTEXT

CTEXT specifies the character set of the output of the function. From that point onward, the data is treated as if it were in that character set. If the data is not in the specified character set, you get the wrong answer.

The character set is required to be specified in this function. If you choose not to specify a character set, you should use the original version of the TEXT function.

**Syntax:**

`CTEXT ( single-object-expression , "character-set-of-object-content" )`

**Meaning:**

`CTEXT ( object_to_convert , object_character_set )`

**Returns:**

A single text item

The first argument, *object\_to\_convert*, represents the object that is converted to a text item, excluding the initiator and terminator of the input object.

The second argument, *object\_character\_set*, represents the character set of the resulting object.

### Example

In this example, the group **Record** has an initiator of the pound sign (#), a terminator of the at sign (@), and a delimiter of a comma (,), and uses the following data:

```
#1339X10A,491.38,Green,42x54@
```

- CTEXT ( Record:card, "ASCII")

Returns: 1339X10A,491.38,Green,42x54

The initiator and terminator are not included because only the content of the object is converted to text.

## DATETOTEXT

The DATETOTEXT function converts a date object or expression to a text item.

### Syntax:

```
DATETOTEXT (single-date-expression)
```

### Meaning:

```
DATETOTEXT (date_to_convert)
```

### Returns:

A single text item

If *date\_to\_convert* is a date object name, this returns the date as a text item formatted according to the presentation of the date object.

If *date\_to\_convert* is a date expression produced by a function, this returns the date as a text item formatted according to the presentation of the output argument of that function.

### Examples

- DATETOTEXT ( ShipDate )

In this example, **ShipDate** is converted from a date to text. If **ShipDate** has a CCYYMMDD presentation, the resulting text item will have that presentation, as well.

- DATETOTEXT ( CURRENTDATETIME ( "{MM/DD/CCYY}" ) )

In this example, CURRENTDATETIME evaluates and returns a date in MM/DD/CCYY format. Then DATETOTEXT evaluates and returns a text string that is that date in MM/DD/CCYY format.

For example, use DATETOTEXT, to do text concatenation. The FROMDATETIME function provides greater flexibility in specifying the format of the resulting text item.

## Related Functions

- FROMDATETIME
- NUMBERTOTEXT
- TEXT
- TEXTTODATE
- TEXTTONUMBER
- TEXTTOTIME
- TIMETOTEXT
- TODATETIME

## FILLLEFT

The FILLLEFT function returns a text item of the length specified. In the output, the text item is preceded with the specified pad value.

You can use FILLLEFT when you have a value that needs to be of a fixed size with a variable number of leading characters with a specified value.

### Syntax:

FILLLEFT (single-text-expression , single-text-expression , single-integer-expression )

### Meaning:

FILLLEFT (text\_to\_fill , pad\_character , pad\_to\_length)

### Returns:

A single text item

The FILLLEFT function returns the text string that results from padding out *text\_to\_fill* by preceding it with the *pad\_character* up to *pad\_to\_length* bytes.

If the pad-length argument is less than the number of bytes in the text to fill, no padding will appear.

## Examples

- FILLLEFT ( AcctID:Transaction , "0" , 5 )  
If **AcctID** has the value 14, FILLLEFT returns 00014
- FILLLEFT ( NUMBERTOTEXT ( InvoiceAmt ) , "\*" , 10 )  
If **InvoiceAmt** has the value 24.75, FILLLEFT returns \*\*\*\*\*24.75

## Related functions

- FILLRIGHT
- LEAVEALPHA



- LEAVEALPHANUM
- LEAVENUM
- LEAVEPRINT
- SQUEEZE
- SUBSTITUTE
- TRIMLEFT
- TRIMRIGHT

## FILLRIGHT

The FILLRIGHT function returns a text item of the length specified. In the output, the text item is appended with the specified pad value.

You can use FILLRIGHT when you have a value that needs to be of a fixed size with a variable number of trailing characters of a specified value.

### Syntax:

FILLRIGHT (single-text-expression , single-text-expression , single-integer-expression)

### Meaning:

FILLRIGHT (text\_to\_fill , pad\_character , pad\_to\_length)

### Returns:

A single text item

FILLRIGHT returns the text string that results from padding out *text\_to\_fill* by appending the *pad\_character* up to *pad\_to\_length* bytes.

If the pad-length argument is less than the number of bytes in the text to fill, no padding will appear.

## Examples

- FILLRIGHT (LastName:Contact, " ", 25)

If **LastName** has the value Peterson, FILLRIGHT returns Peterson followed by 17 spaces.

## Related functions

- FILLLEFT
- LEAVEALPHA
- LEAVEALPHANUM
- LEAVENUM
- LEAVEPRINT
- SQUEEZE
- SUBSTITUTE

- TRIMLEFT
- TRIMRIGHT

## FIND

The FIND function looks for one text string within another text string and returns to its starting position, if found.

### Syntax:

```
FIND (single-text-expression , single-text-expression  
[ , single-number-expression ] )
```

### Meaning:

```
FIND (text_to_find, where_to_look[ , position_to_start_the_search ] )
```

### Returns:

A single integer

FIND returns the starting position of the text item specified by *text\_to\_find* within the text item specified by *where\_to\_look*. A third argument (*position\_to\_start\_the\_search*) can be used to specify the location in *where\_to\_look* for the FIND to begin. Bytes in the text are numbered from left to right, with the leftmost byte being position 1.

If *text\_to\_find* is "none", FIND evaluates to "none".

If a third argument is not used or *position\_to\_start\_the\_search* evaluates to a negative number, it is assumed to be 1. If *position\_to\_start\_the\_search* evaluates to a number greater than the size of *where\_to\_look*, FIND evaluates to "none".

If *text\_to\_find* is not found in the *where\_to\_look* string, FIND evaluates to 0.

## Examples

- FIND ("id", "Florida")

Returns the value 5

- FIND ("id", "Florida", 8)

Returns 0 because the 8 (*position\_to\_start\_the\_search*) is greater than the size of *where\_to\_look*

- FIND ("\", "mypath",2)

Returns 0 because the string "\" was not found in argument 2

## Related functions

- LEFT
- MID
- RIGHT

## HEXTEXTTOSTREAM

HEXTEXTTOSTREAM is the reverse of STREAMTOHEXTEXT. You can use the HEXTEXTTOSTREAM function to assign a binary text value to a character text item represented by hexadecimal pairs.

HEXTEXTTOSTREAM returns a binary text stream whose value is the evaluation of input character text represented by hexadecimal pairs.

### Syntax:

HEXTEXTTOSTREAM (single-text-expression)

### Meaning:

HEXTEXTTOSTREAM (series\_of\_hex\_pairs)

### Returns:

A single byte stream item

This function returns a binary text stream item whose value is the evaluation of input character text in *series\_of\_hex\_pairs*, ignoring <WSP> characters between the hexadecimal pairs. White space characters include space, horizontal tab, carriage return, and line feed characters.

### Input formats

The following table shows an example of input in its character text representation as viewed through the character editor, and in its ASCII code representation (binary text stream) as viewed through the hex editor. Each pair of binary text in the hex view represents one character in the character view of the character text.

Input ("41 42 43 44")	Editor View	Value
Character text (hex pairs)	Character	"41 42 43 44"
ASCII code representation (binary text stream)	Hex	0x3431203432203433203434

### Examples

- HEXTEXTTOSTREAM ("41 42 43 44")

Returns the evaluated value of the input (ASCII) character text string "41 42 43 44" as the output (ASCII) character text string "ABCD" as viewed in the character editor. (The hex view of the input is 0x3431203432203433203434. The hex view of the output is 0x41424344.)

- HEXTEXTTOSTREAM ("0D 0A 00")

Returns the evaluated value of the input (ASCII) character text string "0D 0A 00" as the output (ASCII) character text string "<CR><LF><NULL>" as viewed in the character editor. (The hex view of the input is 0x3044203041203030. The hex view of the output is 0x0D0A00.)

See Design Studio Introduction documentation for a list of special symbols.

## Related functions

- SYMBOL
- STREAMTOHEXTEXT

## LEAVEALPHA

The LEAVEALPHA function removes all non-alphabetic characters from a specified text item.

You can use LEAVEALPHA to remove non-alphabetic characters such as symbols or numbers from a text item.

### Syntax:

LEAVEALPHA (single-text-expression)

### Meaning:

LEAVEALPHA (text\_to\_change)

### Returns:

A single text item

LEAVEALPHA returns a string containing only the alphabetic characters (for example, A-Z and a-z) in *text\_to\_change*.

## Examples

- LEAVEALPHA ("A-b-C-1\$3")

Returns: AbC

## Related functions

- ISALPHA
- ISLOWER
- ISNUMBER
- ISUPPER
- LEAVEALPHANUM
- LEAVENUM
- LEAVEPRINT

## LEAVEALPHANUM

The LEAVEALPHANUM function removes all non-alphanumeric characters (such as symbols) from a specified text item.

### Syntax:

LEAVEALPHANUM (single-text-expression)

**Meaning:**

LEAVEALPHANUM (text\_to\_change)

**Returns:**

A single text item

LEAVEALPHANUM returns a string containing only the alphanumeric characters (for example, A-Z, a-z and 0-9) in *text\_to\_change*.

**Examples**

- LEAVEALPHANUM ( "A-b-C-1\$3" )

Returns: AbC13

**Related functions**

- ISALPHA
- LEAVEALPHA
- ISLOWER
- LEAVENUM
- ISNUMBER
- LEAVEPRINT
- ISUPPER

**LEAVENUM**

The LEAVENUM function removes all non-numeric characters from a text item. For example, you can use LEAVENUM when you want to remove all alphabetic characters and symbols from a text string.

**Syntax:**

LEAVENUM (single-text-expression)

**Meaning:**

LEAVENUM (text\_to\_change)

**Returns:**

A single text item

LEAVENUM returns a string containing only the numeric characters (for example, 0-9) in *text\_to\_change*.

## Examples

- `LEAVENUM ("A-b-C-1$3")`

Returns: 13

## Related functions

- ISALPHA
- LEAVEALPHA
- ISLOWER
- LEAVEALPHANUM
- ISNUMBER
- LEAVEPRINT
- ISUPPER

## LEAVEPRINT

The LEAVEPRINT function removes all non-printable characters from a text item.

### Syntax:

`LEAVEPRINT (single-text-expression)`

### Meaning:

`LEAVEPRINT (text_to_change)`

### Returns:

A single text item

LEAVEPRINT returns a string containing only printable characters in *text\_to\_change*.

## Examples

- `LEAVEPRINT ("A-b<SP>C-1$3<CR><LF>")`

Returns: A-b C-1\$3

## Related functions

- ISALPHA
- LEAVEALPHA
- ISLOWER
- LEAVEAL-PHANUM
- ISNUMBER
- LEAVENUM
- ISUPPER

## LEFT

The LEFT function returns a specified number of characters from a text expression beginning with the leftmost byte of a text item.

You can use LEFT when you need a specific part of a text item. For example, a customer number might have several uses and sometimes only the first 10 characters are needed. Therefore, LEFT can be used to return only the leftmost 10 characters.

### Syntax:

LEFT (single-text-expression , single-integer-expression)

### Meaning:

LEFT (text\_to\_extract\_from , number\_of\_characters\_to\_extract)

### Returns:

A single text item

LEFT returns the leftmost *number\_of\_characters\_to\_extract* characters from *text\_to\_extract\_from* starting at the first (the leftmost) character in *text\_to\_extract\_from*.

If *number\_of\_characters\_to\_extract* evaluates to an integer whose value is less than 1, LEFT evaluates to "none". If *number\_of\_characters\_to\_extract* evaluates to an integer whose value is greater than the size of *text\_to\_extract\_from*, LEFT evaluates to the entire value of *text\_to\_extract\_from*.

## Examples

- LEFT ("Abcd", 2)

Returns Ab

- LEFT ("Abcd", 6)

Returns Abcd

- LEFT (LastName + ", " + FirstName, 25)

Returns the leftmost 25 characters of the text string resulting from the concatenation of **LastName** and **FirstName** (separated by a comma and a space).

### Related functions

- RIGHT
- MID
- FIND

## LOWERCASE

The LOWERCASE function converts an alphabetic text item to all lowercase characters.

#### Syntax:

LOWERCASE (single-text-expression)

#### Meaning:

LOWERCASE (text\_to\_convert)

#### Returns:

A single text item

LOWERCASE produces a text item in which each byte from the input has been converted to lowercase. Any numeric or symbol characters in the text item remain unchanged.

### Examples

- LOWERCASE ("A1b2C!")

Returns: a1b2c!

### Related functions

- ISLOWER
- ISUPPER
- UPPERCASE

## MAX

The MAX function returns the maximum value from a series of number, date, time, or text values.



**Syntax:**

MAX (series-item-expression)

**Meaning:**

MAX (series\_of\_which\_to\_find\_max)

**Returns:**

A single number

The result is the maximum value in the input argument series: number, text, or date/time.

**Examples**

- MAX ( UnitPrice:Input )

If the values for **UnitPrice** are {20, 10, 100}, MAX returns 100.

- MAX(EXTRACT( DueDate:Book:Library, CheckedOut:Book:Library = "Y"))

Returns the maximum (latest) **DueDate** for a book that is checked out from the library.

**Related functions**

- MIN

**MID**

You can use the MID function when you need specific characters from a text item. MID returns one or more characters from a text item.

**Syntax:**

MID (single-text-expression, single-number-expression,  
single-number-expression)

**Meaning:**

MID (source\_text, position\_to\_start\_the\_search, number\_of\_characters)

**Returns:**

A single text item

MID extracts one or more characters from *source\_text* where *position\_to\_start\_the\_search* is the position of the first character to extract and *number\_of\_characters* specifies the number of characters to extract. The first character (leftmost) of *source\_text* has a starting position of 1.

If *position\_to\_start\_the\_search* is greater than the length of *source\_text*, MID returns "none". If *position\_to\_start\_the\_search* or *number\_of\_characters* is less than one, MID returns "none". If the rightmost number of characters of *source\_text*, starting at *position\_to\_start\_the\_search*, is less than *number\_of\_characters*, MID returns the rightmost characters starting at the position specified in *position\_to\_start\_the\_search*.

## Examples

- MID ("abc123", 5, 3)

Returns 23

- MID ("abc123", 7, 1)

Returns "none" because argument2 is larger than the number of characters in argument1

- MID ("abc123", -1, 3)

Returns "none" because argument2 is a negative number

- MID ("abc123", 2, -2)

Returns "none" because argument3 is a negative number

## Related functions

- FIND
- LEFT
- RIGHT

## MIN

Use MIN when you need the minimum value from a series of number, date, time, or text values.

The MIN function returns the minimum value from a series.

### Syntax:

MIN (series-item-expression)

### Meaning:

MIN (series\_of\_which\_to\_find\_min)

### Returns:

A single number

The result is the minimum value of the input series: number, text, or date/time.

## Examples

- MIN (UnitPrice:Input)

If the values for **UnitPrice** are {20,10,100}, MIN returns 10.

- MIN (StartTime::Schedule)

Returns the minimum (earliest) **StartTime** in **Schedule**.

## Related functions

- MAX

## NORMXML

Use this function to remove XML formatting from an input XML fragment. The function accepts text items only.

### Syntax:

NORMXML (single-text-expression)

### Meaning:

NORMXML (XML fragment)

### Returns:

A single text item

This function removes any XML whitespaces from between a (parent) start tag and the start tag of the first child element; and from between the end tag of the child element and the (parent) end tag. *XML whitespace* pertains to carriage returns, line feeds, tabs, and spaces.

For example, all carriage returns, line feeds, tabs, and spaces in between parent tags `<A>` and `</A>` are removed, excluding any that are inside of child element tags `<a>` and `</a>`.

In the following example, there is a carriage return after `<A>`, `</a>`, and `</b>`. There are also three spaces before `<a>` and three spaces before `<b>`.

```
<A>
  <a>This is sample text</a>
  <b>More   sample text</b>
</A>
```

The NORMXML function removes the three carriage returns and the six spaces. As a result, there is one line instead of four lines:

```
<A><a>This is sample text</a><b>More   sample text</b></A>
```

## Limitation

When a type has mixed content, the function removes any XML whitespaces from the character data sections. Here is the previous example, but with mixed content added:

```
<A>My first example
  <a>This is sample text</a>
  <b>More   sample text</b>
</A>
```

This is the result of using the NORMXML function:

```
<A>Myfirstexample<a>This is sample text</a><b>More   sample text</b></A>
```

The XML whitespaces in `My first example` are removed because they are not enclosed within child element tags.

## Using SIZE with NORMXML

The NORMXML function converts the input data to Unicode, and then removes the XML formatting from the input XML fragment. The conversion to Unicode is transparent because the normal processing automatically converts the data from Unicode to the character set that was assigned to the output object.

If the SIZE function is used with the NORMXML function to determine the size of the specified object after NORMXML removes the XML formatting from the input XML fragment, the SIZE operation calculates the size of the data as it exists at that time, in the Unicode character set. The returned size does not match the size that would have been calculated if the data remained in its original character set, unless the original character set was Unicode.

## Using CSIZE with NORMXML

If the CSIZE function is used with the NORMXML function, the CSIZE operation returns the size in characters, regardless of the character set.

## NUMBERTOTEXT

The NUMBERTOTEXT function converts a character number to a text item that looks like the original object.

You can use NUMBERTOTEXT when you need an object that is defined as a number converted to an object defined as text. This is useful when you need to concatenate text, however, the FROMNUMBER function provides greater flexibility in specifying the format of the resulting text item.

### Syntax:

NUMBERTOTEXT (single-number-expression)

### Meaning:

NUMBERTOTEXT (number\_to\_convert)

### Returns:

A single text item

The resulting text looks like the input argument. The result is truncated, if necessary.

## Examples

- NUMBERTOTEXT (ROUND (1000 - 24.75, 3))

This example converts the result of the calculation (rounded to 3 decimal places) to text, resulting in 975.250.

- NUMBERTOTEXT (PurchaseNumber)

This example converts **PurchaseNumber** from a number to text.

## Related functions

- FROMNUMBER
- TEXTTONUMBER
- TODATETIME
- TONUMBER

## PACKAGE

The PACKAGE function converts a group or item object to a text item, including its initiator, terminator, and any delimiters it contains.

### Syntax:

PACKAGE (single-object-expression)

### Meaning:

PACKAGE (object\_to\_convert)

### Returns:

A single text item

The PACKAGE function converts *object\_to\_convert*, which must be a type reference to a text item, including the type reference's initiator, terminator, and all delimiters. PACKAGE differs from TEXT in that it includes the initiator and terminator of the specified type reference.

## Examples

- PACKAGE (Record:Card)

Returns: #1339X10A,491.38,Green,42x54@

For this example, the group **Record** has an initiator of "#", a terminator of "@" and a delimiter of ",". The data looks like this: "#1339X10A,491.38,Green,42x54@".

## Related functions

- DATETOTEXT
- NUMBERTOTEXT
- SERIESTOTEXT
- TIMETOTEXT
- TEXT

PACKAGE differs from TEXT because it includes the initiator and terminator of the input object.

## PROPERCASE

Use PROPERCASE to convert the first alphabetic character in each word to uppercase and all remaining characters to lowercase.

**Syntax:**

PROPERCASE (single-text-item-expression)

**Meaning:**

PROPERCASE (text\_item\_to\_convert)

**Returns:**

A single text item

The PROPERCASE function views the text string as containing a series of "words" where the delimiter between words is the space character. For each "word" in *text\_item\_to\_convert*, PROPERCASE converts the first alphabetic character (for example, A-Z and a-z) found to uppercase and all other characters are converted to lowercase.

**Example**

PROPERCASE ("sally jo BRADLEY")

Returns: Sally Jo Bradley

**Related functions**

- ISALPHA
- ISLOWER
- ISUPPER
- LOWERCASE
- UPPERCASE

## RANDDATA

The RANDDATA function returns random text based on the properties of the output type. The RAND function returns a pseudo-random integer between 0 and RAND\_MAX (guaranteed to be at least 32,767). The RANDDATA function generates a valid random value of the item type that is being built when it is invoked.

**Syntax:**

RANDDATA ()

**Meaning:**

RANDDATA ()

**Returns:**

A single object (text, a date/time object, or a number)

RANDDATA takes no parameters. If the SEED function is called prior to RANDDATA, RANDDATA will return the same value for the same item properties it is returning to, based on the value to SEED.

## Related functions

- RAND
- SEED

## REVERSEBYTE

Use the REVERSEBYTE function when you need the bytes in the opposite sequence. REVERSEBYTE reverses the byte order of an item.

### Syntax:

REVERSEBYTE (single-item-expression)

### Meaning:

REVERSEBYTE (item\_to\_reverse)

### Returns:

A single item

This function reverses the byte order of *item\_to\_reverse*.

## Examples

- REVERSEBYTE ("HI MOM!")

Returns "!MOM IH"

- RIGHT (FullName, FIND (" ", REVERSEBYTE (FullName)) - 1)

If **FullName** is "Alyce N. Wunderland", the above example uses REVERSEBYTE to reverse the characters in **FullName** (resulting in "dnalrednuW .N ecyIA"). Then, the FIND function is evaluated to locate the first space in the resultant string (between the "W" and the ".") that would result in a value of 11. Finally, the RIGHT function is evaluated to take the rightmost 10 (11-1) characters of **FullName**; providing the final result of "Wunderland".



**Note:** The examples are used for illustration only. Do not use this function with string literals.

## RIGHT

You can use RIGHT when you need a specific part of a text item. For example, a customer number might have several uses and sometimes only the last three characters are needed. RIGHT can be used to return only the rightmost three characters.

The RIGHT function returns a specified number of characters from a text expression beginning with the rightmost byte of a text item.

**Syntax:**

RIGHT (single-text-expression , single-integer-expression)

**Meaning:**

RIGHT (text\_to\_extract\_from , number\_of\_characters\_to\_extract)

**Returns:**

A single text item

RIGHT returns the rightmost *number\_of\_characters\_to\_extract* characters from *text\_to\_extract\_from* starting at the last (the rightmost) character in *text\_to\_extract\_from*.

If *number\_of\_characters\_to\_extract* evaluates to an integer whose value is less than 1, RIGHT evaluates to "none". If *number\_of\_characters\_to\_extract* evaluates to an integer whose value is greater than the size of *text\_to\_extract\_from*, RIGHT evaluates to the entire value of *text\_to\_extract\_from*.

**Examples**

- RIGHT ("Abcd" , 2)

Returns cd

- RIGHT ("Abcd" , 6)

Returns Abcd

- RIGHT ("000000" + NUMBERTOTEXT (TransactionNum), 6)

If **TransactionNum** contains 123, this example returns 000123. If **TransactionNum** contains 123456789, this example returns 456789.

**Related functions**

- FIND
- LEFT
- MID

**SEED**

The SEED function primes the RAND and RANDDATA functions so they produce repeated results.

**Syntax:**

SEED(single-number-expression)

**Meaning:**

SEED(number\_for\_seed)



**Returns:**

NONE

**Related functions**

- RAND
- RANDDATA

**SERIESTOTEXT**

You can use the SERIESTOTEXT function to project your input data as a series and to interpret it as a text item for output.

SERIESTOTEXT converts a contiguous or non-contiguous series to a text item.

**Syntax:**

SERIESTOTEXT (series-object-expression)

**Meaning:**

SERIESTOTEXT (series\_to\_convert)

**Returns:**

A single text item

SERIESTOTEXT returns a text item containing the concatenation of the series of the input argument, including nested delimiters but excluding initiators and terminators.

**Examples**

In this example, you have the following data that represents bowler information for a bowling league:

Andrews, Jessica:980206:JBC:145:138:177:159

Little, Randy:980116:BBK:175:168

Wayne, Richard:980102:JBC:185:204:179:164:212

Each record consists of the bowler's name, the date of their last game played, a team code and one or more bowling scores. **Record** is defined as a group that is infix-delimited by a colon.

Using the rule:

= SERIESTOTEXT (Score Field:Bowler:Input)

the following results are produced, which is the concatenation of all of the scores for all of the bowlers, even though the scores are not all contiguous within the data:

145138177159175168185204179164212

However, if the rule was changed, for instance, to concatenate the list of scores to the bowler's name:

= BowlerName Field:Bowler:Input + " ->" + SERIESTOTEXT (Score Field:Bowler:Input)

the following output would be produced:

Andrews, Jessica -> 145138177159

Little, Randy -> 175168

Wayne, Richard -> 185204179164212

In this example, you have an input number that is of variable size, followed by a name. There is no syntax that separates the number from the name. You can define the number as a group with **Byte(s)** as a component and provide a component rule for **Byte(s)**, such as:

ISNUMBER (\$)

## Related functions

- PACKAGE
- TEXT

## SQUEEZE

The SQUEEZE function removes consecutive duplicate occurrences of a specified character or characters from a text item.

### Syntax:

SQUEEZE ( single-text-item , single-text-item )

### Meaning:

SQUEEZE ( text\_to\_squeeze , duplicate\_characters\_to\_remove )

### Returns:

A single text item

SQUEEZE returns a text item with all the consecutive duplicates of *duplicate\_characters\_to\_remove* removed from *text\_to\_squeeze*.

## Examples

- SQUEEZE ( "AB CDE F", " " )

Returns: AB CDE F

- SQUEEZE ( "Connolly", "n" )

Returns: Conolly

## Related functions

- LEAVEALPHA
- LEAVEALPHANUM
- LEAVENUM
- LEAVEPRINT
- SUBSTITUTE

## SUBSTITUTE

You can use the SUBSTITUTE function to replace or remove a character. You can also use this function for multiple pairs.

### Syntax:

SUBSTITUTE ( single-text-expression { , single-text-expression , single-text-expression } )

### Meaning:

SUBSTITUTE (item\_to\_convert, one-or-more-text-substitution-pairs)

Where each one-or-more-text substitution-pair is text\_to\_change , substitute\_text

### Returns:

A single text item

SUBSTITUTE returns the text string that results from replacing all instances of the first *text\_to\_change* with *substitute\_text* in *item\_to\_convert*, then replaces all instances of the second *text\_to\_change* with the *substitute\_text* in the result of the first substitution, and so forth.

## Examples

- SUBSTITUTE ( "123\*456\*7" , "\*" , "/" )

Finds 123\*456\*7 and returns 123/456/7

- SUBSTITUTE ( "120-45-6789" , "-" , "" )

Finds 120-45-6789 and returns 120456789

- =SUBSTITUTE ("ABBA" , "B" , "A" , "A" , "B")

This example illustrates multiple searches for the SUBSTITUTE function.

The first search-and-replace finds all "B"s and returns "A"s: AAAA

The next search-and-replace finds all "A"s and returns "B"s: BBBB

The end result is a return of: BBBB

## Related functions

- LEAVEALPHA
- LEAVEALPHANUM
- LEAVENUM
- LEAVEPRINT
- SQUEEZE

## TEXT

You can use the TEXT function to convert an object to a text item or when echoing entire data objects to another map.

TEXT converts the content of a group or item object to a text item.

### Syntax:

TEXT ( single-object-expression )

### Meaning:

TEXT ( object\_to\_convert )

### Returns:

A single text item

The TEXT function converts *object\_to\_convert* to a text-item, excluding the initiator and terminator of the input object.

## Examples

- TEXT ( Record:card )

Data: #1339X10A,491.38,Green,42x54@

Returns: 1339X10A,491.38,Green,42x54

In this example, the group **Record** has an initiator of the pound sign (#), a terminator of the at-sign (@), and a delimiter of a comma (,).

The initiator and terminator are not included because only the content of the object is converted to text.

## Related functions

- DATETOTEXT
- FROMDATETIME
- FROMNUMBER
- NUMBERTOTEXT
- PACKAGE



**Note:** TEXT differs from PACKAGE in that it does not include the initiator and terminator of the input object.

- SERIESTOTEXT
- TIMETOTEXT

## TEXTTOBCD

The TEXTTOBCD function converts a text item from decimal digits to BCD (Binary Coded Decimal) format.

### Syntax:

TEXTTOBCD ( single-integer-text-expression )

### Meaning:

TEXTTOBCD ( text\_to\_be\_converted )

### Returns:

A single BCD-formatted text item

TEXTTOBCD converts *text\_to\_be\_converted* (which consists of decimal digits) to BCD format. In this format, each byte contains two decimal digits represented as binary numbers. If there is an odd number of decimal digits in the input, the high-order half-byte of the leftmost output byte will contain the decimal value 15 (hex "F").

If anything other than a decimal digit is encountered in the input, TEXTTOBCD returns "none".

## Examples

- TEXTTOBCD ( "1234" )

Returns the hexadecimal value x'1234'

- TEXTTOBCD ( "123A" )

Returns "none"

- TEXTTOBCD ( "123" )

Returns the hexadecimal value x'F123'

In this example, the values shown as input ("123") are meant to represent character items in the native character set to the machine on which the map is running. On a personal computer, "123" would contain the ASCII characters for the digits that have the hexadecimal values "31", "32", and "33". The output, described as "the hexadecimal value `F123", consists of the two binary bytes "F1" and "23".

On an IBM mainframe the input string would contain EBCDIC characters for the digits that have the hexadecimal values "F1", "F2", "F3", Å°, but the output would be the same as the personal computer output.

## Related functions

- BCDTOHEX
- BCDTOINT
- BCDTOTEXT

## TEXTTONUMBER

Use TEXTTONUMBER to convert text to a number.

The TONUMBER function provides greater flexibility for specifying the format of the text item that is to be converted to a number.

### Syntax:

TEXTTONUMBER ( single-text-expression )

### Meaning:

TEXTTONUMBER ( text\_to\_convert\_to\_number )

### Returns:

A single character number

The *text\_to\_convert\_to\_number* must be in integer or ANSI-formatted (floating point) presentation. The resulting number looks like the input argument, however, nonsignificant zeroes to the right of the decimal separator will be truncated. If the input argument is in error (for example, it is not a recognizable as a valid number), the result is "none".

When specified as in ANSI-formatted presentation, the text string must meet the following requirements:

- The decimal point can be a period, a comma, or "none".
- The leading sign can be a plus sign, a minus sign, or "none".
- No thousands separator is allowed.

## Examples

- TEXTTONUMBER (OrderQty)

Returns **OrderQty** as a character number item

## Related functions

- DATETOTEXT
- FROMNUMBER
- NUMBERTOTEXT
- TEXTTODATE

- TEXTTOTIME
- TIMETOTEXT

## TEXTTOTIME

Use TEXTTOTIME when you want to convert an object defined as text that is in HHMM or HHMMSS presentation, to an item defined as time. For greater flexibility, use the TODATETIME function for specifying the format of the text item that is to be converted to a date/time.

### Syntax:

TEXTTOTIME ( single-text-expression )

### Meaning:

TEXTTOTIME ( text\_to\_convert\_to\_time )

### Returns:

A single time

The *text\_to\_convert\_to\_time* must be in HHMM or HHMMSS presentation. HH is a two-digit hour in a 24-hour format. If the result is being assigned to a time object, the resulting time looks like the output object. Otherwise, the resulting time looks like the input argument. If the input argument is in error (for example, it is not a valid time), the result is "none".

## Examples

- TEXTTOTIME ( CallTime )

Returns **CallTime** as a time item

## Related functions

- DATETOTEXT
- TEXTTONUMBER
- FROMDATETIME
- TIMETOTEXT
- NUMBERTO-TEXT
- TODATETIME
- TEXTTODATE

## TIMETOTEXT

You can use the TIMETOTEXT function to perform text concatenation. For greater flexibility, use the FROMDATETIME function for specifying the format of the resulting text item.

TIMETOTEX converts a time object or expression to a text item.

### Syntax:

TIMETOTEXT ( single-time-expression )

### Meaning:

TIMETOTEXT ( time\_to\_convert\_to\_text )

### Returns:

A single text item

If *time\_to\_convert\_to\_text* is a time object name, this returns the time as a text item formatted according to the presentation of the input date object.

If *time\_to\_convert\_to\_text* is a time expression produced by a function, this returns the time as a text item formatted according to the presentation of the output argument of that function.

## Examples

- TIMETOTEXT ( LeadTime )

In this example, **LeadTime** is converted from a time to text. If **LeadTime** has an HH:MM presentation, the resulting text item will be of that presentation.

- TIMETOTEXT ( CURRENTDATETIME ( "{HH:MM:SS}" ) )

Here, CURRENTDATETIME evaluates and returns a time in HH:MM:SS format. Then, TIMETOTEXT evaluates and returns a text string that is that time in HH:MM:SS format.

## Related functions

- DATETOTEXT
- FROMDATETIME
- NUMBERTOTEXT
- TEXTTODATE
- TEXTTONUMBER
- TEXTTOTIME
- TODATETIME

## TODATETIME

The TODATETIME function converts a text string of a specified format to a date-time item.



**Syntax:**

```
TODATETIME ( single-character-text-expression
            [, single-text-expression ] )
```

**Meaning:**

```
TODATETIME ( text_to_convert [, date_time_format_string ] )
```

**Returns:**

A single character date item

TODATETIME returns the date-time that corresponds to the value specified by *text\_to\_convert*, which is in the format specified by *date\_time\_format\_string*. If *date\_time\_format\_string* is not specified, it will be assumed that *text\_to\_convert* is in [CCYYMMDDHH24MMSS] format.

The *date\_time\_format\_string* must conform to the date-time format strings as described in "Format strings".

**Examples**

- TODATETIME ( "05/14/1999@10:14pm" , "{MM/DD/CCYY}@{HH12:MMAM/PM}" )

In this example, a text string containing a date and time is converted to a date-time item.

- RptDate = TODATETIME ( RIGHT ( GETRESOURCEName() , 8 ) , "CCYYMMDD" )

Assume that you receive a file that contains historical data. The name of the file identifies the date of the historical data. For example, a filename of **19960424** indicates that the data was produced on April 24, 1996. To map this date to **RptDate**, the TODATETIME function could be used with the RIGHT and GETRESOURCEName functions.

**Related functions**

- CURRENTDATE
- CURRENTDATETIME
- CURRENTTIME
- TEXTTODATE
- TEXTTOTIME

**TONUMBER**

The TONUMBER function converts a text string of a specified format to a number.

**Syntax:**

```
TONUMBER ( single-character-text-expression
            [, single-text-expression ] )
```

**Meaning:**

```
TONUMBER ( text_to_convert [, number_format_string ] )
```

**Returns:**

A single character number item

TONUMBER returns the number that corresponds to the value specified by *text\_to\_convert*, which is in the format specified by *number\_format\_string*. If *number\_format\_string* is not specified, it will be assumed that *text\_to\_convert* is in ANSI decimal format (for example, "{L-####[':##]}").

The *number\_format\_string* must conform to the number format strings as described in the "Format strings" topic.

**Examples**

- TONUMBER(*text\_to\_convert*, "{L+'\$'#,###}")

**L+'\$'** indicates the leading dollar sign is positive. That leading sign and the comma separators are removed when the text is converted to a number.

Input String: \$123,000,000

Output: 123000000

- TONUMBER(*text\_to\_convert*, "####T-}")

Four number signs are required for each whole number, regardless of the actual number of digits in the number.

Input string:	Output:	Note:
12345-	-12345	The output becomes a negative number.
67890	67890	No change occurs.
345-	-345	The output becomes a negative number.

- TONUMBER(*text\_to\_convert*, "####T+'K'-}")

If an invalid character, such as an X, is encountered, nothing is returned.

If a **K** is encountered, it is treated as a positive indicator.

Input string:	Output:	Note:
11212-	-11212	The output becomes a negative number.
67890X		The X is an invalid character. No number is returned.
54354	54354	No change occurs.
34567K	34567	The K is recognized as a positive sign. The character is removed and the number is returned as a positive.
345-	-345	The output becomes a negative number.

- `TONUMBER(text_to_convert, "{L-'('#;###T-}')`")

The parentheses indicating a negative number are removed and replaced with a negative sign.

Comma separators are removed when the text is converted to a number.

<b>Input string:</b>	<b>Output:</b>	<b>Note:</b>
(12,345)	-12345	The output becomes a negative number. The comma separator is removed.
67,890	67890	The comma separator is removed.
(345)	-345	The output becomes a negative number.

- `TONUMBER(text_to_convert, "{#[;]###['.##5]T+'K-}')`

The optional comma separators are removed, but the decimal points and decimal values are retained.

<b>Input string:</b>	<b>Output:</b>	<b>Note:</b>
54,345.098	54354.098	The comma separator is removed.
67890.0X		The X is an invalid character. No number is returned.
11213-	-11213	The output becomes a negative number.
34567K	34567	The K is recognized as a positive sign. The character is removed and the number is returned as a positive.
345.1-	-345.1	The output becomes a negative number.

## Related functions

- DATETONUMBER
- FROMNUMBER
- NUMBERTODATE
- NUMBERTOTEXT

## TRIMLEFT

You can use the TRIMLEFT function to remove spaces or a text string at the beginning of some text.

TRIMLEFT removes leading characters from a text item.

### Syntax:

`TRIMLEFT ( single-text-expression [ , single-text-expression ] )`

### Meaning:

`TRIMLEFT ( item_to_trim [ , text_to_trim ] )`

**Returns:**

A single text item

TRIMLEFT removes leading characters that match *text\_to\_trim* from *item\_to\_trim* and returns the result as a single text item. If *text\_to\_trim* is not specified, leading spaces are removed from *item\_to\_trim*.

**Examples**

- TRIMLEFT ( " abc" )

Returns: abc

- TRIMLEFT ( "000345" , "0" )

Returns: 345

- TRIMLEFT ( LastName Column:Row , "<WSP>" )

Returns the content of **LastName Column** after removing all leading whitespace characters (space, horizontal tab, carriage return, or line feed characters).

**Related functions**

- FILLLEFTP
- LEAVEPRINT
- FILLRIGHT
- SQUEEZE
- LEAVEALPHA
- SUBSTITUTE
- LEAVEAL-PHANUM
- TRIMRIGHT
- LEAVENUM

**TRIMRIGHT**

You can use the TRIMRIGHT function to remove spaces or a text string at the end of text.

TRIMRIGHT removes trailing characters from a text item.

**Syntax:**

```
TRIMRIGHT ( single-text-expression [ , single-text-expression ] )
```

**Meaning:**

```
TRIMRIGHT ( item_to_trim [ , text_to_trim ] )
```

**Returns:**

A single text item

TRIMRIGHT removes trailing characters that match *text\_to\_trim* from *item\_to\_trim* and returns the result as a single text item. If *text\_to\_trim* is not specified, trailing spaces are removed from *item\_to\_trim*.

**Example**

- `TRIMRIGHT ( "abc " )`

Returns: abc

- `TRIMRIGHT ( "Cat in the Hat!?!?!?" , "!?" )`

Returns: Cat in the Hat

- `TRIMRIGHT ( LastName Column:Row )`

Returns the content of **LastName Column** after removing all trailing spaces.

**Related functions**

- FILLLEFTP
- LEAVEPRINT
- FILLRIGHT
- SQUEEZE
- LEAVEALPHA
- SUBSTITUTE
- LEAVEAL-PHANUM
- TRIMLEFT
- LEAVENUM

## UPPERCASE

The UPPERCASE function converts text to all uppercase characters.

**Syntax:**

UPPERCASE ( single-text-expression )

**Meaning:**

UPPERCASE ( text\_to\_convert )

**Returns:**

A single text item

UPPERCASE produces a text item in which each byte in *text\_to\_convert* has been converted to uppercase. Any numeric or symbolic characters in *text\_to\_convert* remain unchanged.

### Examples

- UPPERCASE ( "abC123!" )

Returns ABC123!

### Related functions

- ISLOWER
- ISUPPER
- LOWERCASE

## WORD

You can use the WORD function to parse a text item that is delimited by some character, such as a space or a comma.

WORD returns the characters between two user-defined separators within a text item. The separators are counted from left to right when the third argument is positive, and from right to left when the third argument is negative, enabling the function to search from either end of the item.

**Syntax:**

WORD (single-text-expression , single-text-expression ,  
single-integer-expression)

**Meaning:**

WORD (text\_to\_search , word\_separator , number\_of\_word\_to\_get)

**Returns:**

A single text item

WORD returns the characters (word) between the  $n$ th-1 and  $n$ th *word\_separator*, where  $n$  corresponds to *number\_of\_word\_to\_get*.

Define the separator (*word\_separator*) and specify the number of the occurrence of that separator. The WORD function returns the characters between the  $n$ th-1 and  $n$ th separators in the delimited text item.

The separator is case sensitive.

## Examples

The following examples assume that a file exists named **Letter** with two text objects named **Line1** and **Line2**:

```
Line1:Congratulations, Mr Brown! You're a winner!;
```

```
Line2:You may have already won 1 million dollars!;
```

- WORD ( Line1:Letter , " " , 3 )

Returns: `Brown!`

The exclamation point is returned because it is read as a character within the word before the separator (a space).

- WORD ( Line1:Letter , " " , 6 )

Returns: `winner!`

If the  $n$ th separator is missing and the  $n$ th-1 separator exists, the function returns the characters between the last separator and the end of the delimited text item. The separator is a space; "`winner!`" (including the exclamation point) is the sixth word.

- WORD ( Line2:Letter , "!" , 3 )

Returns "none"

Both  $n$ th and  $n$ th-1 separators are missing. In this example, there is only one separator, located at the end of the text object. As a result, there is no third word because the function sees everything before "!" as the first word.

- WORD ( Line1:Letter , " " , 1 )

Returns: `Congratulations,`

If  $n-1 = 0$ , the function returns the characters between the beginning of the delimited text item and the first separator.

- WORD ( Line1:Letter , " " , -1 )

Returns: `winner!`

If  $n+1 = 0$ , then the function returns the characters between the end of the text item and the last separator.

## Related functions

- FIND
- MID

## XML functions

There are several XML functions that you can call from component rules and map rules to process input XML data. They fall under two different categories. The XML functions in one category are called through the `XMLLIB` library. They are the `XPATH` (used with `XMLLIB`), `VALIDATEEX`, `VALIDATE`, `XPATHEX`, `XSLT` (used with `XMLLIB`), and `XSLTEX` functions. The XML functions in the other category are called directly, without using the `XMLLIB` library. They are the `XPATH`, `XVALIDATE`, and `XSLT` functions.

## VALIDATE

The `VALIDATE` function validates the XML input by using the `XMLLIB` library.

This function validates XML input, which is provided as a text stream or URL, against the provided XML Schema, returning `0` if the validation succeeded or `-1` otherwise.

### Syntax:

```
VALIDATE (single-text-expression, single-text-expression)
```

### Meaning:

```
VALIDATE (xml_url_or_xml_fragment, target_namespace XML_schemaname)
```

### Returns:

A single number

## Exemple

### Examples

- `NUMBERTOTEXT( xmllib->VALIDATE( "ipo.in.xml", "urn:h17-org:v3 http://www.example.com/IPO ipo.xsd" ) )`

Returns `0` when validation of the `ipo.in.xml` file succeeds and returns `-1` otherwise.

- `VALID( NUMBERTOTEXT( xmllib->VALIDATE( PACKAGE(source), "ipo.xsd" ) ), LASTERRORMSG() )`

Returns `0` when validation of the input XML fragment succeeds or returns a validation error message otherwise.

## VALIDATEEX

The `VALIDATEEX` function validates the XML input, and logs the function processing in the directory where the map runs by using the `XMLLIB` library.



This function validates XML input, which is provided as a text stream or URL, against the provided XML Schema. The VALIDATEEX function returns 0 if the validation succeeded or -1 if validation failed, and generates the specified trace log file in the directory.

**Syntax:**

VALIDATEEX (single-text-expression, single-text-expression, tracelog)

**Meaning:**

VALIDATEEX (xml\_url\_or\_xml\_fragment, target\_namespace XML\_schemaname, trace\_file\_name.extension)

**Returns:**

A single number

**Example****Example**

```
• NUMBERTOTEXT( xmllib->VALIDATEEX( "ipo.in.xml", "urn:hl7-org:v3 http://www.example.com/IPO ipo.xsd",
  "numtotext.log" ))
```

Returns 0 when validation of the ipo.in.xml file succeeds or returns -1 when validation fails, and generates a trace log of function processing.

**XVALIDATE**

The XVALIDATE function validates the XML input.

This function validates XML input, provided as a text stream or URL, against the provided XML Schema.

**Syntax:**

XVALIDATE (single-text-expression, single-text-expression)

**Meaning:**

XVALIDATE (xml\_url\_or\_xml\_fragment, target\_namespace XML\_schemaname)

**Returns:**

This function evaluates to a Boolean "true" or "false".

**Example****Examples**

```
• IF( XVALIDATE( "ipo.in.xml", "http://www.example.com/IP ipo.xsd" ), "VALIDATION SUCCESS", LASTERRORMSG())
```

Returns the message VALIDATION SUCCESS when the validation of the ipo.in.xml file succeeds and returns a message that explains the validation error otherwise.

## XPATH functions

### XPATH

The XPATH function queries the XML input.

This function queries the XML input, which is provided as a text stream or URL, by using the specified XPATH expression and context, returning the result of the evaluation.

**Syntax:**

XPATH (single-text-expression, single-text-expression, single-text-expression)

**Meaning:**

XPATH (xml\_url\_or\_xml\_fragment, xpath\_expression, context\_expression)

**Returns:**

A single text item

**Example**

**Examples**

- XPATH( "current://ipo.in.xml", "/ipo:purchaseOrders/order/items/item[1]/shipDate", "ipo http://www.example.com/IPO" )

Returns the content of the `shipDate` element from the `ipo.in.xml` file.

- XPATH( PACKAGE(source), "/ipo:purchaseOrders/order/items/item[1]/shipDate", "ipo http://www.example.com/IPO" )

Returns the content of the `shipDate` element from the input XML fragment.

### XPATH used with XMLLIB

The XPATH function that is used with XMLLIB queries the XML input.

This function queries the XML input, which is provided as a text stream or URL, by using the specified XPATH expression and context, returning the result of the evaluation.

**Syntax:**

XPATH (single-text-expression, single-text-expression, single-text-expression)

**Meaning:**

XPATH (xml\_url\_or\_xml\_fragment, xpath\_expression, context\_expression)

**Returns:**

A single text item

**Example**

## Examples

- `xmllib->XPATH( "ipo.in.xml", "./order//item[1]/shipDate", "/ipo:purchaseOrders " )`

Returns the content of the `shipDate` element from the `ipo.in.xml` file.

- `xmllib->XPATH( PACKAGE(source), "/ipo:purchaseOrders/order/items/item[1]/shipDate", "/" )`

Returns the content of the `shipDate` element from the input XML fragment.

## XPATHEX

The XPATHEX function queries the XML input by using the XMLLIB library, and logs function processing in the directory where the map runs.

This function queries the XML input, provided as a text stream or URL, by using the specified XPath expression and context. The XPATHEX function returns the result of the evaluation and generates the specified trace log file in the directory where the map runs.

### Syntax:

XPATHEX (single-text-expression, single-text-expression, single-text-expression, tracelog)

### Meaning:

XPATHEX (xml\_url\_or\_xml\_fragment, xpath\_expression, context\_expression, trace\_file\_name.extension)

### Returns:

A single text item

### Example

#### Example

- `xmllib->XPATHEX( "ipo.in.xml", "./order//item[1]/shipDate", "/ipo:purchaseOrders ", "xpathlog.txt")`

Returns the content of the `shipDate` element from the `ipo.in.xml` file and generates a trace log of function processing.

## XSLT functions

### XSLT

The XSLT function applies an XSLT transformation.

This function applies an XSLT transformation, expressed as a text stream or URL, to the provided XML input, returning the result of the transformation.

### Syntax:

XSLT (single-text-expression, single-text-expression)

**Meaning:**

XSLT (xml\_url\_or\_xml\_fragment, xslt\_url\_or\_xslt\_fragment)

**Returns:**

A single text item

**Example**

**Examples**

- XSLT( "current://ipo.in.xml", "current://ipo.xsl" )

Applies the XSLT transformation that is defined in the `ipo.xsl` file to the `ipo.in.xml` file.

- XSLT( PACKAGE( source ), "current://ipo.xsl" )

Applies the XSLT transformation that is defined in the `ipo.xsl` file to the input XML fragment.

## XSLT used with XMLLIB

The XSLT function that is used with XMLLIB applies an XSLT transformation.

This function applies an XSLT transformation, expressed as a text stream or URL, to the provided XML input, returning the result of the transformation.

**Syntax:**

XSLT (single-text-expression, single-text-expression)

**Meaning:**

XSLT (xml\_url\_or\_xml\_fragment, xslt\_url\_or\_xslt\_fragment)

**Returns:**

A single text item

**Example**

**Examples**

- xmllib->XMLLIB XSLT( "ipo.in.xml", "ipo.xsl" )

Applies the XSLT transformation that is defined in the `ipo.xsl` file to the `ipo.in.xml` file.

- xmllib->XMLLIB XSLT( PACKAGE( source ), "ipo.xsl" )

Applies the XSLT transformation that is defined in the `ipo.xsl` file to the input XML fragment.

## XSLTEX

The XSLTEX function applies an XSLT transformation to the XML input, and logs the function processing in the directory where the map runs by using the XMLLIB library.

This function applies an XSLT transformation, expressed as a text stream or URL, to the provided XML input. The XSLTEX function returns the result of the transformation and generates the specified trace log file in the directory.

**Syntax:**

XSLTEX (single-text-expression, single-text-expression, tracelog)

**Meaning:**

XSLTEX (xml\_url\_or\_xml\_fragment, xslt\_url\_or\_xslt\_fragment, trace\_file\_name.extension)

**Returns:**

A single text item

**Example**

**Example**

```
• xmllib->XSLTEX( "ipo.in.xml", "ipo.xsl", "transform.log")
```

Applies the XSLT transformation that is defined in the `ipo.xsl` file to the `ipo.in.xml` file, and generates a trace log of function processing

## Custom functions

From the Type Designer or Map Designer, you can create custom functions to call external libraries. While the EXIT function is often used to call external libraries, the EXIT function is limited to support only text objects and is operating system dependent. When you create a custom function to use in a map rule, the function is operating system independent and supports text, number, and date-time objects.

Similar to the regular functions that are shipped with the product, a maximum of four parameters are supported per argument.

## Creating a custom function

To create your own functions, a C or C++ development tool is required, or the Java Native Interface (JNI) if you are working in a Java environment.

1. From a component rule in the Type Designer or from a map rule in the Map Designer, right-click and select the **Insert Function** option.  
(Additionally in the Map Designer, if you are not in a map rule, you can create a new function by selecting **Rules > New Function**.)
2. From the **Insert Function** window, select **Design**.  
**Result**  
The **Custom Function Modules** window is displayed.
3. In the **Path** field, enter the path of where to create the external library.  
This should always be in the `install_dir/function_libs` directory.
4. In the **Name** field, enter a name for the library.

- Click **Add** to add a function to the library.

**Result**

The **Function Specifics** window is displayed.

- In the **Name** field, enter a name for the function.
- For **Return Type**, select the function return type from the drop-down list.  
You can select up to four function parameters from the respective drop-down list. Choices include Boolean, date, number, text, time, and byte stream.
- In the space provided, enter a description for your function.  
This information will be displayed in the **Insert Function** window of both Designers.
- Click **OK** to validate the selected parameters and close the **Function Specifics** window.

- Click **Generate**.

**Result**

The Designer generates a collection of operating system specific makefiles and definition files that provide the framework for the function you are creating.

- Go to the *install\_dir/function\_libs* directory to view the results.

**Result**

As a result of the generation process, framework files are created for each operating system that HCL® OneTest™ Data supports. You can find the operating system-specific makefiles and definition files in the *install\_dir/function\_libs/your\_new\_lib* directory.

- Now you must modify the framework that was generated by the Designer.

The following functions with the parameter information that you selected were exported to the .c file:

- GetFunctionCount
- GetFunctionName
- GetInputParameter
- GetReturnType
- GetParameterCount
- GetFunctionDesc

To complete the new function, open the .c file and add your programming code for the applicable function or functions provided.

Use the .c file to build your dynamic link library (DLL) and then place the DLL in the *install\_dir/function\_libs/your\_new\_lib* directory. (All custom designed libraries and functions must be placed in the *install\_dir/function\_libs* directory.)

The new library name is listed under **Category** in the **Insert Function** window, and the new function that you created is placed in the list of functions and will remain available for future use from both the Type Designer and Map Designer applications.

### Example

The following example displays how to implement a custom function called SIN.

```
void ConvertToBytes(double returnVal, LPEXITPARAM lpep)
{
```

```

double value = 0.0;
int decimal = 2, sign = 0, j = 0, k = 0;
char byString[100];
char* lpbyData = _fcvt(returnValue, 7, &decimal, &sign );
memset(byString, 0, sizeof(byString));

if (sign)
    byString[j++] = '-';

if (decimal <= 0)
{
    byString[j++] = '0';
    byString[j++] = '.';
    while (decimal != 0)
    {
        byString[j++] = '0'; decimal++;
    }
}
else if (decimal > 0)
{
    while (decimal != 0)
    {
        byString[j++] = lpbyData[k++]; decimal--;
    }
    byString[j++] = '.';
}
while (lpbyData[k]) byString[j++] = lpbyData[k++];

byString[j] = '\0';

if (NULL == (lppep->lpDataFromApp = GlobalAllocPtr
    (GHND, j + 1)))
{
    lppep->nReturn = -1;
    lstrcpy(lppep->szErrMsg, "Memory allocation failed in Alternate");
    return;
}

memcpy(lppep->lpDataFromApp, byString, j);
lppep->dwFromLen = j;
lppep->lpDataFromApp[j++] = '\0';
}

void CALLBACK EXPORT SIN(LPEXITPARAM lppep)
{
    double value = 0.0;
    double returnValue = 0.0;
    LPEXITPARAMEXTENDED lpExtended = NULL;

    if (lppep->dwSize != sizeof(EXITPARAM))
    {
        return;
    }

    lpExtended = (LPEXITPARAMEXTENDED)lppep->lpv;
    value = atof(lpExtended->lpFirstInputParameter);

```

```

returnValue = sin(value);
ConvertToBytes(returnValue, lpep);
    lpep->wCleanupAction = GetReturnType("SIN");
    lstrcpy(lpep->szErrMsg, "SIN function was successful");

return;
}

```

### What to do next

At run time, all custom designed libraries and functions that your maps use must be in the *install\_dir/function\_libs* directory. When you deploy a map that uses a custom function to a remote host, the library is not transferred. Therefore you must manually copy the custom function library to the *install\_dir/function\_libs* directory on the remote host.

## Date and time format strings

You can use the listed format strings for numbers, dates and times in functions such as the CURRENTDATETIME, FROMNUMBER, TONUMBER, FROMDATETIME, and TODATETIME.

Create custom date and time formats by using the symbols based on the given format strings.

## Time units

### Symbol

#### Description

#### HH24

Hour (based on a 24-hour clock) in two digits (00 to 24)

#### H24

Hour (based on a 24-hour clock) in one or two digits as needed (0 to 24)

#### HH12

Hour (based on a 12-hour clock) in two digits (1-12)

#### H12

Hour (based on a 12-hour clock) in one or two digits as needed (1 to 12)

#### MM

Minute in two digits (00 to 59)

#### M

Minute in one or two digits as needed (0 to 59)

#### SS

Seconds in two digits (00 to 59)



**S**

Second in one or two digits as needed (0 to 59)

**AM/PM**

Meridian (AM/PM)

**ZZZ**

Three character abbreviation for time zone (EST, and so on)

**+/-ZZZZ**

Hours and minutes before or after Greenwich Mean Time (GMT), also known as Coordinated Universal Time (UTC)

**+/-ZZ:ZZ**

4-digit time where the format is a 2-digit hour and 2-digit minute, separated by a colon.

**+/-ZZ[:ZZ]**

4-digit time where the format is a 2-digit hour and an optional 2-digit minute, separated by colon.

**+/-ZZ[ZZ]**

4-digit time where the format is a 2-digit hour and an optional 2-digit minute.

**TZD**

4-digit time where the format is a 2-digit hour and 2-digit minute, separated by a colon. Z on output, if value is +/-00:00.

## Date units

**Symbol****Description****CCYY**

Full year including century (2001)

**YY**

Last two digits of the year (00-99)

**MM**

Month of the year in two numeric digits (01-12)

**M**

Month of the year in one or two numeric digits as needed (1 to 12)

**MON**

First three letters of the month (Jan to Dec)

**MONTH**

Full name of the month (January to December)

**DDD**

Day of the year in three numeric digits (001 to 366)

**DD**

Day of the month in two numeric digits (01 to 31)

**D**

Day of the month in one or two numeric digits as needed (1 to 31)

**DY**

First three letters of the weekday (Sun to Sat)

**DAY**

Full name of the weekday (Sunday to Saturday)

**WW**

Week of year (1-52)

**Qn**

Quarter of the year (Q1-Q4)

**EEYY**

Emperor's year, long form

**EY**

Emperor's year, short form

## Binary date and time format strings

**Sub-String Name**

**Sub-String Value**

**Binary DateTime**

[ "{" + Date + "}" ] + [ "{" + Time + "}" ]

**Date**

CCYYMMDD

YYMMDD

CCYYDDD

YYDDD

**Time**

HH24MMSS

HH24MM

**Japanese date and time format strings****Sub-String Name****Sub-String Value****Japanese DateTime**

"{ + DateTime +}"

"{ + DateTime +}" + Separator(1) + "{ + DateTime +}"

"{ + DateTime +}" + "[" + Separator(1) + "{ + DateTime +}" + "]"

**DateTime**

Date

Time

**Separator**

Separator is optional. If specified, it can be up to 120 bytes, composed of non-alphabetic characters. Symbol table values, such as <CR>, can be used to indicate non-printable characters.

The Separator is required if the second DateTime option is used and the format string of the first DateTime ends in a variable sized specification - EY, M, or D for Date or H24, H12, M, or S for Time.

**Japanese date format strings****Sub-String Name****Sub-String Value****Japanese Date**

Year + MonthSet(1)

**Year**

CCYY

YY

EEYY

EY

**MonthSet**

Separator(1) + "MM" + DayOfMonth(1)

"[" + Separator(1) + "MM" + DayOfMonth(1) + ]

Separator(1) + M + DayOfMonth(1)

[ + Separator(1) + M + DayOfMonth(1) + ]

#### **DayOfMonth**

Separator(1) + "DD" + WeekDay(1)

"[" + Separator(1) + "DD" + WeekDay(1) + "]"

Separator(1) + "D" + WeekDay(1)

"[" + Separator(1) + "D" + WeekDay(1) + "]"

#### **WeekDay**

Separator(1) + "DY"

"[" + Separator(1) + "DY" + "]"

Separator(1) + "DAY"

"[" + Separator(1) + "DAY" + "]"

#### **Separator**

Separator is optional. If specified, it can be up to 120 bytes composed of non-alphabetic characters. Symbol table values, such as <CR>, can be used to indicate non-printable characters.

MonthSet Separator required if Year is EY

DayOfMonth Separator required if MonthSet is M

WeekDay Separator required if DayOfMonth is D

## Japanese time format strings

### **Sub-String Name**

#### **Sub-String Value**

### **Japanese Time**

Meridian(1) + HMS

### **Meridian**

"AM/PM"

### **HMS**

"HH24" + MinuteSet(1)

"H24" + MinuteSet(1)

"HH12" + MinuteSet(1)

"H12" + MinuteSet(1)

### MinuteSet

Separator(1) + "MM" + SecondSet(1)

"[" + Separator(1) + "MM" + SecondSet(1) + "]"

Separator(1) + "M" + SecondSet(1)

"[" + Separator(1) + "M" + SecondSet(1) + "]"

### SecondSet

Separator(1) + "SS"

"[" + Separator(1) + "SS" + "]"

Separator(1) + "S"

"[" + Separator(1) + "S" + "]"

### Separator

Separator is optional. If specified, it can be up to 120 bytes, composed of non-alphabetic characters. Symbol table values, such as <CR> can be used to indicate non-printable characters.

MinuteSet Separator required if HMS is H24 or H12

SecondSet Separator required if MinuteSet is M

## Western date and time format strings

### Substring Name

#### Substring Value

### Western DateTime

"{" + DateTime + "}"

"{" + DateTime + "}" + Separator(1) + "{" + DateTime + "}"

"{" + DateTime + "}" + "[" + Separator(1) + "{" + DateTime + "}" + "]"

### DateTime

Date

Time

**Separator**

Separator is optional. If specified, it can be up to 120 bytes, composed of non-alphabetic characters. Symbol table values, such as <CR>, can be used to indicate non-printable characters.

The Separator is required if the second DateTime option is used and the format string of the first DateTime ends in a variable sized specification - M, MONTH, D, or DAY for Date or H24, H12, M, or S for Time.

**Western date format strings****Sub-String Name****Sub-String Value****Date**

DateUnit + DatePart2(1)

**DatePart2**

Separator(1) + DateUnit + DatePart3(1)

"[" + Separator(1) + DateUnit + DatePart3(1) + "]"

**DatePart3**

Separator(1) + DateUnit + DatePart4(1)

"[" + Separator(1) + DateUnit + DatePart4(1) + "]"

**DatePart4**

Separator(1) + DateUnit

"[" + Separator(1) + DateUnit + "]"

**Separator**

Separator is optional. If specified, it can be up to 120 bytes, composed of non-alphabetic characters. Symbol table values, such as <CR>, can be used to indicate non-printable characters.

When DateUnit has a variable length (M, MONTH, D, DAY), a Separator must follow that DateUnit if data follows.

**Western time format strings****Sub-String Name****Sub-String Value****Western Time**

Hours + MinutesSet(1) + Meridian(1) + Zone(1)

**Hours**

HH24

H24

HH12

H12

**MinutesSet**

Separator(1) + "MM" + SecondSet(1)

"[" + Separator(1) + "MM" + SecondSet(1) + "]"

Separator(1) + "M" + SecondSet(1)

"[" + Separator(1) + "M" + SecondSet(1) + "]"

**SecondSet**

Separator(1) + "SS" + FractionSet(1)

"[" + Separator(1) + "SS" + FractionSet(1) + "]"

Separator(1) + "S" + FractionSet(1)

"[" + Separator(1) + "S" + FractionSet(1) + "]"

**FractionSet**

Separator(1) + MinPlaces + "-" + MaxPlaces

"[" + Separator(1) + MinPlaces + "-" + MaxPlaces + "]"

**MinPlaces**

An integer from 0 to 9

**MaxPlaces**

An integer from 0 to 9 (must be less than MinPlaces)

**Meridian**

"AM/PM"

"[" + "AM/PM" + "]"

**Zone**

"+/--ZZZZ"

"ZZZ"

"+/-ZZ:ZZ"

"+/-ZZ[:ZZ]"

"+/-ZZ[ZZ]"

"TZD"

"[+/-ZZZZ]"

"[ZZZ]"

"[+/-ZZ:ZZ]"

"[+/-ZZ[:ZZ]]"

"[+/-ZZ[ZZ]]"

"[TZD]"

### Separator

Separator is optional. If specified, it can be up to 120 bytes, composed of non-alphabetic characters. Symbol table values, such as <CR>, can be used to indicate non-printable characters.

MinuteSet Separator required if Hours is H24 or H12

SecondSet Separator required if MinuteSet is M

FractionSet Separator required if SecondSet is S

## Number format strings

You can create custom number formats by using the given format strings.

### Decimal

"{" + Leading-Sign(1) + Whole# + Fraction + Trailing-Sign(1) + "}"

### Integer

"{" + Leading-Sign(1) + Whole# + Trailing-Sign(1) + "}"

## Leading sign format strings

Sub-String Name	Sub-String Value	Positive	Negative
Leading-Sign	"L" + Positive + Negative + Zero(1)	Req	Req
	-or-		
	"L" + "[" + Positive + "]" + Negative + Zero(1)	Opt	Req
	-or-		



Sub-String Name	Sub-String Value	Positive	Negative
	"L" + Positive + "[" + Negative + "]" + Zero(1)	Req	Opt
	-or-		
	"L" + Positive + Zero(1)	Req	-
	-or-		
	"L" + Negative + Zero(1)	-	Req

### Trailing sign format strings

Sub-String Name	Sub-String Value	Positive	Negative
Trailing-Sign	"T" + Positive + Negative + Zero(1)	Req	Req
	"T" + "[" + Positive + "]" + Negative + Zero(1)	Opt	Req
	"T" + Positive + "[" + Negative + "]" + Zero(1)	Req	Opt
	"T" + Positive + Zero(1)	Req	-
	"T" + Negative + Zero(1)	-	Req

### Substring format strings

Substring Name	Substring Value	Meaning
Positive	"+" + Value(1)	
Negative	"-" + Value(1)	
Value	A text-string enclosed in single quotation marks '. Value has a release character of / if the text contains any single quotation mark ' or forward slash / characters.	The sign value of the previous sign-indicator. If Value is not used, the default is + for positive numbers and - for negative numbers.
Zero	"Z" + Value "[" + "Z" + Value + "]"	Specifies the required leading sign value if the number is zero. If Zero is not used, there is no sign associated with a zero.  Specifies the optional leading sign value if the number is zero. If Zero is not used, there is no sign associated with a zero.

## Whole number and fraction format strings

### Whole# substring name

#### Substring value

##### Meaning

```
MinDigits(1) + "#" + Value(1) + "###" + MaxDigits(1)
```

```
MinDigits(1) + "#" + "[" + Value(1) + "]" + "###" + MaxDigits(1)
```

Specifies the thousands separator and the range of whole number digits. If min-digits is not used, the default is zero. If max-digits is not used, the default is "S". If a ThousandsItem is specified, a Value must be present as the default for the syntax item.

### Fraction substring name

#### Substring value

##### Meaning

```
"V" + ImpliedPlaces
```

Specifies the decimal to be in an explicit place, and ImpliedPlaces determines where the intended decimal separator is to be placed.

```
Value + MinDigits(1) + "###" + MaxDigits(1)
```

Specifies the value of the decimal separator to be required and the range of fraction digits. If min-digits is not used, default is zero. If max-digits is not used, default is "S"

```
"[" + Value + MinDigits(1) + "###" + MaxDigits(1) + "]"
```

Specifies the value of the decimal separator to be optional if there is no fractional portion of the number. It also specifies the range of fraction digits. If min-digits is not used, default is zero. If max-digits is not used, default is "S"

## RUN function return codes

The RUN function return codes and messages might result when using the RUN function. Return codes and messages are returned when the particular activity completes. Return codes and messages might also be recorded as specified in the audit logs, trace files, execution summary files, and so forth.

The following table lists the return codes and messages that can result when using the RUN function.

#### Return Code

##### Message

50

*Memory allocation failure*

Occurs when memory fails.

**51***Card override failure*

Occurs when memory fails.

**52***I/O initialization failure*

Occurs when memory fails.

**53***Open audit failure*

The audit log file is not accessible.

**54***No command line*

There is nothing to process.

**55***Recursive command files*

More than one command file is included in the command line.

**56***Invalid command line option -x*

The option is invalid for the command.

**57***Invalid `W' command line option*

The Work file option is invalid.

**58***Invalid `B' command line option*

The Batch (close) file option is invalid.

**59***Invalid `R' command line option*

The Refresh Rate option is invalid.

**60***Invalid `A' command line option*

The Audit option is invalid.

**61**

*Invalid `P` command line option*

The Paging option is invalid

**62**

*Invalid `Y` command line option*

The General I/O Retry option is invalid.

**63**

*Invalid `T` command line option*

The Trace option is invalid.

**64**

*Invalid `G` command line option*

The Ignore option is invalid

**65**

*Invalid `I` command line option for input x*

The Source option is invalid for the identified input.

**66**

*Invalid size in echo command line for input x*

The size specified using the Size option is greater than memory allowed.

**67**

*Invalid adapter type in command line for input x*

The adapter is not of a known adapter type. Includes -IMxxx where xxx is an unknown adapter alias.

**68**

*Invalid `O` command line option for output x*

The target option is invalid for output x. The number of characters between the single quotation marks that represent the options for an adapter exceed 258 characters in the adapter override.

**69**

*Invalid adapter type in command line for output x*

The adapter is not of a known adapter type. Includes -OMxxx where xxx is an unknown adapter alias.

**70**

*Command line memory failure*

Occurs when memory is exceeded during echo or override card commands.

**71**

*Invalid `D' command line option*

The Date option is invalid.

**72**

*Invalid `F' command line option*

The Failure option is invalid.

**73**

*Resource manager failure*

(Launcher only) The resource manager is not used, possibly a memory failure.

**74**

*Invalid `Z' command line option*

The Ignore option is invalid.

**75**

*Adapter failed to get data on input*

Enable the adapter trace to record the adapter activity to discover the cause of the error.

**76**

*Adapter failed to put data on output*

Enable the adapter trace to record the adapter activity to discover the cause of the error.

**77**

*Invalid map name*

This message can occur in two different cases. First, this message occurs when the map name specified on the command line is more than 32 bytes long in UTF-8 encoding. Also, this message can occur when there is an error in the command line such that text for another execution command is erroneously being interpreted as the map name. For example, in the command line below, the number representing the size of the echoed data is missing.

```
mymap.mmc -IE1S HereIsMyDataButIForgotToSpecifyTheSize -AED
```

Because the size is missing, it is interpreted to be 0, such that there is no echoed data. The next string encountered on the command line

```
(HereIsMyData...)
```

Because it does not start with a hyphen (-), it is assumed to be the name of the next map to execute. Because the text is longer than 32 bytes in UTF-8 encoding, the *Invalid Map Name* message is returned.

## Character set codes for CPACKAGE, CSERIESTOTEXT, and CTEXT

The second argument of the CPACKAGE, CSERIESTOTEXT, and CTEXT functions specifies the character set of the output of the function. The value of the second argument (the character set of the object content), must be a valid character set code.

All actions done that use the text string that results from one of these functions (CPACKAGE, CSERIESTOTEXT, or CTEXT) treat the text string as being of the specified character set—it is not automatically treated as Native.

<b>Character set code</b>	<b>Data language</b>
Native	Native
ASCII	ASCII (deprecated)
EBCDIC	EBCDIC (deprecated)
UNICODE_BE	UNICODE Big Endian (deprecated)
EUC	EUC (deprecated)
SJIS	SJIS (deprecated)
IBMKANJI	IBM Kanji (deprecated)
CIKANJI	CII Kanji (deprecated)
JIS	JIS (deprecated)
UNICODE_LE	UNICODE Little Endian (deprecated)
UTF-8	UTF-8 (deprecated)
Latin1	Latin1 (deprecated)
UTF-8\ibm-1208	UTF-8
UTF-16	UTF-16
UTF-16BE	UTF-16 Big Endian
UTF-16LE	UTF-16 Little Endian
UTF-32	UTF-32
UTF-32BE	UTF-32 Big Endian
UTF-32LE	UTF-32 Little Endian
UTF16_PlatformEndian	UTF-16 Platform Endian
UTF16_OppositeEndian	UTF-16 Opposite Endian
UTF32_PlatformEndian	UTF-32 Platform Endian
UTF32_OppositeEndian	UTF-32 Opposite Endian
UTF-16BE,version=1	UTF-16BE,version=1

Character set code	Data language
UTF-16LE,version=1	UTF-16LE,version=1
UTF-16,version=1	UTF-16,version=1
UTF-16,version=2	UTF-16,version=2
UTF-7	UTF-7
IMAP-mailbox-name	IMAP-mailbox-name
SCSU	SCSU
BOCU-1	BOCU-1
CESU-8	CESU-8
ISO-8859-1	Latin1
US-ASCII	US-ASCII
gb18030	gb18030
ibm-912_P100-1995	Latin2
ibm-913_P100-2000	Latin3
ibm-914_P100-1995	CESU-8
ibm-915_P100-1995	Cyrillic
ibm-1089_P100-1995	Arabic
ibm-9005_X110-2007	Greek8
ibm-813_P100-1995	ibm-813
ibm-5012_P100-1999	Hebrew
ibm-916_P100-1995	ibm-916
ibm-920_P100-1995	Latin5
iso-8859_10-1998	Latin6
Iso-8859_11-2001	Thai8
ibm-921_P100-1995	ibm-921
iso-8859_14-1998	Latin8
ibm-923_P100-1998	Latin-9
ibm-942_P12A-1999	shift_jis78
ibm-943_P15A-2003	MS_Kanji
ibm-943_P130-1999	Shift_JIS
ibm-33722_P12A-1999	EUC-JP

Character set code	Data language
ibm-33722_P120-1999	ibm-33722_P120-1999
ibm-954_P101-2000	ibm-954_P101-2000
ibm-1373_P100-2002	ibm-1373_P100-2002
windows-950-2000	Big5
ibm-950_P110-1999	ibm-950_P110-1999
ibm-1375_P100-2003	Big5-HKSCS (IBM)
ibm-5471_P100-2006	ibm-5471_P100-2006
ibm-1386_P100-2001	ibm-1386_P100-2001
windows-936-2000	GBK
ibm-1383_P110-1999	EUC-CN
ibm-5478_P100-1995	GB_2312-80
ibm-964_P110-1999	EUC-TW
ibm-949_P110-1999	ibm-949_P110-1999
ibm-949_P11A-1999	ibm-949_P11A-1999
ibm-970_P110-1995	ibm-eucKR
ibm-971_P100-1995	ibm-971_P100-1995
ibm-1363_P11B-1998	ibm-1363 (korean)
ibm-1363_P110-1997	ibm-1363_P110-1997
windows-949-2000	Windows 949 (korean)
windows-874-2000	Windows 874
ibm-874_P100-1995	ibm-874
ibm-1162_P100-1999	ibm-1162
ibm-437_P100-1995	ibm-437
ibm-720_P100-1997	ibm-720
ibm-737_P100-1997	ibm-737
ibm-775_P100-1996	ibm-775
ibm-850_P100-1995	ibm-850
ibm-851_P100-1995	ibm-851
ibm-852_P100-1995	ibm-852
ibm-855_P100-1995	ibm-855



Character set code	Data language
ibm-856_P100-1995	ibm-856
ibm-857_P100-1995	ibm-857
ibm-858_P100-1997	ibm-858
ibm-860_P100-1995	ibm-860
ibm-861_P100-1995	ibm-861
ibm-862_P100-1995	ibm-862
ibm-863_P100-1995	ibm-863
ibm-864_X110-1999	ibm-864
ibm-865_P100-1995	ibm-865
ibm-866_P100-1995	ibm-866
ibm-867_P100-1998	ibm-867
ibm-868_P100-1995	ibm-868
ibm-869_P100-1995	ibm-869
ibm-878_P100-1996	KOI8-R
ibm-901_P100-1999	ibm-901
ibm-902_P100-1999	ibm-902
ibm-922_P100-1999	ibm-922
ibm-1168_P100-2002	KOI8-U
ibm-4909_P100-1999	ibm-4909
ibm-5346_P100-1998	ibm-5346
ibm-5347_P100-1998	ibm-5347
ibm-5348_P100-1997	ibm-5348
ibm-5349_P100-1998	ibm-5349
ibm-5350_P100-1998	ibm-5350
ibm-9447_P100-2002	ibm-9447
ibm-9448_X100-2005	ibm-9448
ibm-9449_P100-2002	ibm-9449
ibm-5354_P100-1998	ibm-5354
ibm-1250_P100-1995	ibm-1250
ibm-1251_P100-1995	ibm-1251

Character set code	Data language
ibm-1252_P100-2000	ibm-1252
ibm-1253_P100-1995	ibm-1253
ibm-1254_P100-1995	ibm-1254
ibm-1255_P100-1995	ibm-1255
ibm-5351_P100-1998	ibm-5351
ibm-1256_P110-1997	ibm-1256
ibm-5352_P100-1998	ibm-5352
ibm-1257_P100-1995	ibm-1257
ibm-5353_P100-1998	ibm-5353
ibm-1258_P100-1997	ibm-1258
macos-0_2-10.2	macintosh
macos-6-10.2	x-mac-greek
macos-7_3-10.2	x-mac-cyrillic
macos-29-10.2	x-mac-ce
macos-35-10.2	x-mac-turkish
ibm-1051_P100-1995	hp-roman8
ibm-1276_P100-1995	ibm-1276 (Adobe Standard Encoding)
ibm-1006_P100-1995	ibm-1006
ibm-1098_P100-1995	ibm-1098
ibm-1124_P100-1996	ibm-1124
ibm-1125_P100-1997	ibm-1125
ibm-1129_P100-1997	ibm-1129
ibm-1131_P100-1997	ibm-1131
ibm-1133_P100-1997	ibm-1133
ISO_2022,locale=ja,version=0	ISO_2022,locale=ja,version=0
ISO_2022,locale=ja,version=1	ISO_2022,locale=ja,version=1 (JIS)
ISO_2022,locale=ja,version=2	ISO_2022,locale=ja,version=2
ISO_2022,locale=ja,version=3	ISO_2022,locale=ja,version=3 (JIS7)
ISO_2022,locale=ja,version=4	ISO_2022,locale=ja,version=4 (JIS8)
ISO_2022,locale=ko,version=0	ISO_2022,locale=ko,version=0

Character set code	Data language
ISO_2022,locale=ko,version=1	ISO_2022,locale=ko,version=1
ISO_2022,locale=zh,version=0	ISO_2022,locale=zh,version=0
ISO_2022,locale=zh,version=1	ISO_2022,locale=zh,version=1
ISO_2022,locale=zh,version=2	ISO_2022,locale=zh,version=2
HZ	HZ-GB-2312
x11-compound-text	x11-compound-text
ISCII,version=0	x-iscii-de
ISCII,version=1	x-iscii-be
ISCII,version=2	x-iscii-pa
ISCII,version=3	x-iscii-gu
ISCII,version=4	x-iscii-or
ISCII,version=5	x-iscii-ta
ISCII,version=6	x-iscii-te
ISCII,version=7	x-iscii-ka
ISCII,version=8	x-iscii-ma
LMBCS-1	LMBCS-1
ibm-37_P100-1995	ibm-037 (ebcdic-cp-us/ca/wt/nl)
ibm-273_P100-1995	ebcdic-de
ibm-277_P100-1995	EBCDIC-CP-DK/NO
ibm-278_P100-1995	ebcdic-cp-fi/se/sv
ibm-280_P100-1995	ebcdic-cp-it
ibm-284_P100-1995	ebcdic-cp-es
ibm-285_P100-1995	ebcdic-cp-gb
ibm-290_P100-1995	EBCDIC-JP-kana
ibm-297_P100-1995	ebcdic-cp-fr
ibm-420_X120-1999	ebcdic-cp-ar1
ibm-424_P100-1995	ebcdic-cp-he
ibm-500_P100-1995	ebcdic-cp-be/ch
ibm-803_P100-1999	ibm-803
ibm-838_P100-1995	IBM-Thai

<b>Character set code</b>	<b>Data language</b>
ibm-870_P100-1995	ebcdic-cp-roece/yu
ibm-871_P100-1995	ebcdic-is
ibm-875_P100-1995	ibm-875
ibm-918_P100-1995	ebcdic-cp-ar2
ibm-930_P120-1999	ibm-930
ibm-933_P110-1995	ibm-933
ibm-935_P110-1999	ibm-935
ibm-937_P110-1999	ibm-937
ibm-939_P120-1999	ibm-939
ibm-1025_P100-1995	ibm-1025
ibm-1026_P100-1995	ibm-1026
ibm-1047_P100-1995	ibm-1047
ibm-1097_P100-1995	ibm-1097
ibm-1112_P100-1995	ibm-1112
ibm-1122_P100-1999	ibm-1122
ibm-1123_P100-1995	ibm-1123
ibm-1130_P100-1997	ibm-1130
ibm-1132_P100-1998	ibm-1132
ibm-1137_P100-1999	ibm-1137
ibm-4517_P100-2005	ibm-4517
ibm-1140_P100-1997	ibm-1140 (ebcdic-us-37+euro)
ibm-1141_P100-1997	ibm-1141 (ebcdic-de-273+euro)
ibm-1142_P100-1997	ibm-1142 (ebcdic-dk/no-277+euro)
ibm-1143_P100-1997	ibm-1143 (ebcdic-fi/se-278+euro)
ibm-1144_P100-1997	ibm-1144 (ebcdic-it-280+euro)
ibm-1145_P100-1997	ibm-1145 (ebcdic-es-284+euro)
ibm-1146_P100-1997	ibm-1146 (ebcdic-gb-285+euro)
ibm-1147_P100-1997	ibm-1147 (ebcdic-fr-297+euro)
ibm-1148_P100-1997	ibm-1148 (ebcdic-international+euro)
ibm-1149_P100-1997	ibm-1149 (ebcdic-is-871+euro)

Character set code	Data language
ibm-1153_P100-1999	ibm-1153
ibm-1154_P100-1999	ibm-1154
ibm-1155_P100-1999	ibm-1155
ibm-1156_P100-1999	ibm-1156
ibm-1157_P100-1999	ibm-1157
ibm-1158_P100-1999	ibm-1158
ibm-1160_P100-1999	ibm-1160
ibm-1164_P100-1999	ibm-1164
ibm-1364_P110-1997	ibm-1364
ibm-1371_P100-1999	ibm-1371
ibm-1388_P103-2001	ibm-1388
ibm-1390_P110-2003	ibm-1390
ibm-1399_P110-2003	ibm-1399
ibm-5123_P100-1999	ibm-5123
ibm-8482_P100-1999	ibm-8482
ibm-16684_P110-2003	ibm-16684
ibm-4899_P100-1998	ibm-4899
ibm-4971_P100-1999	ibm-4971
ibm-9067_X100-2005	ibm-9067
ibm-12712_P100-1998	ebcdic-he
ibm-16804_X110-1999	ebcdic-ar
ibm-37_P100-1995,swaplfnl	ibm-37-s390
ibm-1047_P100-1995,swaplfnl	ibm-1047-s390
ibm-1140_P100-1997,swaplfnl	ibm-1140-s390
ibm-1141_P100-1997,swaplfnl	ibm-1141-s390
ibm-1142_P100-1997,swaplfnl	ibm-1142-s390
ibm-1143_P100-1997,swaplfnl	ibm-1143-s390
ibm-1144_P100-1997,swaplfnl	ibm-1144-s390
ibm-1145_P100-1997,swaplfnl	ibm-1145-s390
ibm-1146_P100-1997,swaplfnl	ibm-1146-s390

<b>Character set code</b>	<b>Data language</b>
ibm-1147_P100-1997,swaplfnl	ibm-1147-s390
ibm-1148_P100-1997,swaplfnl	ibm-1148-s390
ibm-1149_P100-1997,swaplfnl	ibm-1149-s390
ibm-1153_P100-1999,swaplfnl	ibm-1153-s390
ibm-12712_P100-1998,swaplfnl	ibm-12712-s390
ibm-16804_X110-1999,swaplfnl	ibm-16804-s390
ebcdic-xml-us	ebcdic-xml-us

# Chapter 9. Troubleshooting Guide

This guide describes how to analyze and resolve some of the common problems that you might encounter while you work with HCL OneTest™ Server.

Known problems are documented. For more information, see the download document [https://support.hcltechsw.com/csm?id=kb\\_article&sysparm\\_article=KB0081911](https://support.hcltechsw.com/csm?id=kb_article&sysparm_article=KB0081911). You can also search this information center for troubleshooting documentation.

You can contact HCL Support if you are unable to troubleshoot the problem. Gather all the required background information and provide the details to HCL Support for investigation. For more information, see [HCL Customer Support](#).

## Troubleshooting issues

You can find information about the issues or problems that you might encounter while working with HCL OneTest™ Server. Details about issues, their causes and the resolutions that you can apply to fix the issues are described.

The troubleshooting issues are presented to you in the following tables based on where or when you might encounter these issues on HCL OneTest™ Server.

- [Table 16: Troubleshooting issues: installation on page 987](#)
- [Table 17: Troubleshooting issues: server administration on page 990](#)
- [Table 18: Troubleshooting issues: resource monitoring on page 990](#)
- [Table 19: Troubleshooting issues: configuring test runs on page 990](#)
- [Table 20: Troubleshooting issues: test or stub runs on page 992](#)
- [Table 21: Troubleshooting issues: test results and reports on page 995](#)

**Table 16. Troubleshooting issues: installation**


Problem	Description	Solution
On Ubuntu, when you are installing the server software and you encounter errors in the scripts that are running.	At times, scripts might not appear to be running due to any of the following reasons: <ul style="list-style-type: none"><li>• Slow connection speeds.</li><li>• Insufficient CPU, memory, or disk resources.</li><li>• A firewall that was configured incorrectly is already enabled.</li></ul>	You can complete any of the following tasks: <ul style="list-style-type: none"><li>• To identify the issue, you can perform a diagnostic check by running the following command:<pre>journalctl -u k3s</pre>This command displays the log that you can use to check for the problem.</li></ul>

Table 16. Troubleshooting issues: installation (continued)

Problem	Description	Solution
		<ul style="list-style-type: none"> <li>Run the following command to see which pods are running and which pods are not running: <pre>kubectl get pods -A</pre> </li> <li>Run the following command to get details about a specific pod: <pre>kubectl describe pod -n &lt;namespace&gt; &lt;pod name&gt;</pre> </li> <li>Follow the on-screen instructions to resolve the errors.</li> <li>Some issues can be solved by re-running the following script: <pre>sudo ./ubuntu-init.sh</pre> </li> </ul>
On Ubuntu, DNS is not working as expected.		<p>The DNS configuration that is used by the cluster can be displayed by using the following command:</p> <pre>kubectl get cm -n kube-system coredns -ojsonpath="{.data.Corefile}"</pre> <p>The <b>forward</b> setting displays the <i>nameservers</i> that are used. For example, you might see the following in the <i>corefile</i>:</p> <pre>.:53 { : forward . 8.8.8.8 9.9.9.9 : }</pre>



Table 16. Troubleshooting issues: installation (continued)

Problem	Description	Solution
		<p>A script (<code>ubuntu-set-dns.sh</code>) is supplied for managing these values.</p> <p>For example, to set the DNS values for the values shown in the previous example:</p> <pre>sudo ./ubuntu-set-dns.sh --server 8.8.8.8 --server 9.9.9.9</pre> <p> <b>Note:</b> If you do not use <code>sudo</code> in the command, the script runs but the configuration might be lost if the cluster is restarted.</p> <p>To learn more about the behavior of the script, run the following command:</p> <pre>sudo ./ubuntu-set-dns.sh --help</pre>
When running <code>helm install</code> the created pods keep crashing, and the logs contain: <code>ACCESS_REFUSED</code> when trying to connect to RabbitMQ	In some instances, the RabbitMQ password is not automatically setup correctly.	<p>Manually apply the necessary password:</p> <pre>kubectl exec -n &lt;namespace&gt; &lt;release-name&gt;-rabbitmq-0 -- rabbitmqctl change_password user \ "\$(kubectl get secret -n &lt;namespace&gt; &lt;release-name&gt;-rabbitmq -o jsonpath='{.data.rabbitmq-p assword}'   base64 --decode)"</pre>

**Table 17. Troubleshooting issues: server administration**

<b>Problem</b>	<b>Description</b>	<b>Solution</b>
When a user is assigned an additional role, the change in the permissions is not observed in the browser.		You must log out of the session and log in again for the changed role to take effect.
You see the following message displayed on HCL OneTest™ Server:  <div style="border: 1px solid black; padding: 5px; width: fit-content;"> <p>You can't request to join a project that has no owners</p> </div>	You requested to join a project that no longer has an owner. Orphaned projects occur when the project owners are deleted. This can occur, for example, when the person leaves the organization.	Ask an administrator to take ownership of the project, and then add you as a member.

**Table 18. Troubleshooting issues: resource monitoring**

<b>Problem</b>	<b>Description</b>	<b>Solution</b>
You are not able to add a Prometheus server as a Resource Monitoring source.	The cause might be that you have not installed the Prometheus server at the time of server installation.	Verify that the Prometheus server was installed in Helm at the time of server installation. See <a href="#">Installing the server software on Ubuntu by using k3s on page 64</a> . If not, consult your cluster administrator to get the Prometheus server installed and configured.

**Table 19. Troubleshooting issues: configuring test runs**

<b>Problem</b>	<b>Description</b>	<b>Solution</b>
When you configure a run of a schedule that matches the following conditions:	The cause might be because of the following reasons:	To resolve the problem, select from either of the following methods:

Table 19. Troubleshooting issues: configuring test runs (continued)

Problem	Description	Solution
<ul style="list-style-type: none"> <li>• The schedule has two user groups configured to run on static agents when the schedule was created in HCL OneTest™ Performance V10.1.</li> <li>• One of the user groups is disabled and the asset is committed to the remote repository.</li> </ul> <p>Both the static agents are displayed as available for the test run in the <b>Location</b> tab of the <b>Execute test asset</b> dialog box when only one agent that is configured for the user group must be available.</p>	<ul style="list-style-type: none"> <li>• The schedule was created in HCL OneTest™ Performance V10.1.</li> <li>• The user group that is disabled is not removed or deleted from the test resources.</li> <li>• The agent configured on the disabled user group is already added as an agent to the server project and is available for selection.</li> </ul>	<ul style="list-style-type: none"> <li>• By using HCL OneTest™ Performance V10.1.1.</li> </ul> <p>Perform the following steps:</p> <ol style="list-style-type: none"> <li>1. Open the schedule in HCL OneTest™ Performance V10.1.1.</li> <li>2. Save the schedule and the project.</li> <li>3. Commit your test asset to the remote repository.</li> <li>4. Proceed to configure a run for the schedule on HCL OneTest™ Server V10.1.1.</li> </ol> <ul style="list-style-type: none"> <li>• By using HCL OneTest™ Performance V10.1.</li> </ul> <p>Perform the following steps:</p> <ol style="list-style-type: none"> <li>1. Select the disabled user group.</li> <li>2. Click <b>Remove</b>.</li> <li>3. Save the schedule and the project.</li> <li>4. Commit your test asset to the remote repository.</li> <li>5. Proceed to configure a run for the schedule on HCL OneTest™ Server V10.1.1.</li> </ol>
<p>You have added a remote repository to your project that contains the test assets or resources of the following types:</p>	<p>This problem occurs if the server extension is not enabled. Although the extension was enabled at the time of installation of HCL OneTest™ Server,</p>	<p>You must verify if the server extension is enabled and running by running the following command: <code>kubectl get pod -n &lt;test-system&gt;</code>, where <code>&lt;test-system&gt;</code> is the namespace that</p>

**Table 19. Troubleshooting issues: configuring test runs (continued)**

Problem	Description	Solution
<ul style="list-style-type: none"> <li>• Postman</li> <li>• JMeter</li> <li>• JUnit</li> </ul> <p>The test assets or resources are not displayed on the <b>Execution</b> page for you to select the asset for a run.</p>	<p>it was disabled subsequently by the server administrator.</p>	<p>you created to install the server software. The server extensions that are running are displayed.</p> <p>If the server extension that you want is not running implying that the server extension is not enabled. You must enable the server extension. Contact the server administrator to enable the server extension.</p>

**Table 20. Troubleshooting issues: test or stub runs**


Problem	Description	Solution										
<p>You encounter any of the following issues:</p> <ul style="list-style-type: none"> <li>• When many tests are run simultaneously on the default cluster location and you observe the following issues: <ul style="list-style-type: none"> <li>◦ Out of memory</li> </ul> </li> </ul>	<p>The issue is seen when any of the following events occur:</p> <ul style="list-style-type: none"> <li>• Many tests are run in parallel.</li> <li>• The memory that is used by the tests during the test run exceeds the allocated default memory of 1 GB.</li> <li>• The default</li> </ul>	<p>To resolve the problem, you can increase the resource allocation for test runs.</p> <p>You can enter arguments in the <b>Additional configuration options</b> field in the <b>Advanced</b> settings panel of the <b>Execute test asset</b> dialog box when configuring a test run.</p> <p> <b>Important:</b> The memory settings that you configure for a test run is persisted for the test when ever you run it. You must use this setting judiciously. Configuring all tests for an increased memory limit might affect subsequent test runs or cause other memory issues when tests run simultaneously.</p> <p>You can increase the resource allocation for test runs by using any of the following arguments:</p> <table border="1"> <thead> <tr> <th>Requirement</th> <th>Configuration option name</th> <th>Default value, if no value is set</th> <th>An example value</th> <th>Result of using the example value</th> </tr> </thead> <tbody> <tr> <td>Specifying the memory</td> <td><code>init.resource.memory.limit</code></td> <td><i>1024Mi</i></td> <td><i>2048Mi</i></td> <td>Increases the memory limit of the</td> </tr> </tbody> </table>	Requirement	Configuration option name	Default value, if no value is set	An example value	Result of using the example value	Specifying the memory	<code>init.resource.memory.limit</code>	<i>1024Mi</i>	<i>2048Mi</i>	Increases the memory limit of the
Requirement	Configuration option name	Default value, if no value is set	An example value	Result of using the example value								
Specifying the memory	<code>init.resource.memory.limit</code>	<i>1024Mi</i>	<i>2048Mi</i>	Increases the memory limit of the								

Table 20. Troubleshooting issues: test or stub runs (continued)

Problem	Description	Solution				
<p>er- rors.</p> <ul style="list-style-type: none"> <li>◦ Ob- serve that the test runs are slow with a high CPU us- age.</li> <li>◦ The Ku- ber- netes pods are get- ting evict- ed.</li> </ul> <p>• When you run an AFT suite that con- tains mul- tiple Web UI tests and you observe the fol- lowing is- sues:</p>	<p>memo- ry of the container is not ade- quate for the test run.</p> <ul style="list-style-type: none"> <li>• Pods are evicted due to low node memory.</li> </ul>	Requirement	Configura- tion option name	Default val- ue, if no val- ue is set	An example value	Result of us- ing the ex- ample value
		limit of the <i>init container</i> .				<i>init container</i> from the de- fault value to <i>2048Mi</i> .
		Configuring a larger mem- ory request for the <i>init</i> <i>container</i> to avoid pod eviction.	<code>init.re- source.memo- ry.request</code>	<i>64Mi</i>	<i>1024Mi</i>	Increases the initial mem- ory request for the <i>init</i> <i>container</i> from the de- fault value to <i>1024Mi</i> .
		Specifying the cpu re- quest for the <i>init container</i> .	<code>init.re- source.cpu- .request</code>	<i>50m</i>	<i>60m</i>	Increases the cpu re- quest for the <i>init contain- er</i> from the default value <i>60m</i> .
		Specifying the memory limit of the container used for the test run.	<code>resource.mem- ory.limit</code>	The larger of <i>3Gi</i> or <i>max- imum heap size + 1Gi</i>	<i>4Gi</i>	Changes the memory lim- it of the main container from the de- fault value to <i>4Gi</i> .
		Specifying the memory request for the container used by the test run.	<code>resource.mem- ory.request</code>	<i>64Mi</i>	<i>1024Mi</i>	Increases the memory re- quest for the main contain- er from the

**Table 20. Troubleshooting issues: test or stub runs (continued)**

Problem	Description	Solution				
<ul style="list-style-type: none"> <li>◦ Error stating that the browser might not be installed or the browser version is unsupported.</li> <li>◦ Error stating multiple random timeouts or an in-</li> </ul>		Requirement	Configuration option name	Default value, if no value is set	An example value	Result of using the example value
						default value to <i>1024Mi</i> .
		Specifying the cpu request for the main container used by the test run.	<code>resource.cpu-request</code>	<i>50m</i>	<i>70m</i>	Increases the cpu request for the main container from the default value to <i>70m</i> .
<p>In addition, in the <b>JVM Arguments</b> field under the <b>Advanced</b> settings you can set the maximum heap size for the test runtime. For example, adding the JVM argument <b>-Xmx3g</b> sets the maximum heap size to <i>3Gi</i>.</p>						

Table 20. Troubleshooting issues: test or stub runs (continued)

Problem	Description	Solution
terminal error.		
You are not able to run the Istio stubs from the <b>Execution</b> page.	The cause might be that the fully qualified domain name is not specified in the <b>Host</b> field for the stub when it was created.	Verify and ensure to add the fully qualified domain name of the server in the <b>Host</b> field when the physical transport for the stub is configured in HCL OneTest™ API.



**Note:** You can refer to the [Kubernetes documentation](#) for information about the different units that can be used for resources in the **Additional configuration option** fields.

Table 21. Troubleshooting issues: test results and reports

Problem	Description	Solution
You are not able to view the Jaeger traces for the tests you ran.	<p>The cause can be as follows:</p> <ul style="list-style-type: none"> <li>• Jaeger was not pre-installed in OpenShift.</li> <li>• The Jaeger trace is not supported for the particular test that you ran.</li> </ul>	<p>Check for any of the following solutions:</p> <ul style="list-style-type: none"> <li>• Verify that Jaeger was installed in Helm at the time of server installation. See <a href="#">Installing the server software on Ubuntu by using k3s on page 64</a>. If not, consult your administrator to get Jaeger installed and configured.</li> <li>• Verify that the tests you ran are supported for Jaeger traces. See <a href="#">Test results and reports overview on page 436</a>.</li> </ul>

**Table 21. Troubleshooting issues: test results and reports (continued)**

Problem	Description	Solution
		<ul style="list-style-type: none"> <li>• Verify if you provided the program variables when you configured the test run. See <a href="#">Considerations for using Jaeger traces in reports on page 277</a>.</li> </ul>

## Troubleshooting issues

You can find information about the issues or problems that you might encounter while working with HCL® OneTest™ Data. Details about issues, their causes and the resolutions that you can apply to fix the issues are described.

### Schema designing error or warning messages

The HCL® OneTest™ Data analyzer validates the logical and structural consistency of the data definition of a schema. In case of any inconsistency in the schema structure, the analyzer issues error or warning messages based on the type of the analysis.

- Logical analysis addresses the integrity of the relationships that you define in the schema.
- Structural analysis addresses the integrity of the underlying database.

Warnings indicate a successful analysis and are relatively insignificant. Warning messages provide information about inconsistencies that occurred, when you changed the schema, and are automatically resolved.

Error messages are important. Error messages provide information about errors in the type definitions that you must correct. An error might result in unpredictable results in the mapping of the data definitions.

### Return codes and error messages

You can find information about both error messages and warnings based on structural analysis or logical analysis.

### Schema analysis logic error messages

The following table lists the logic error messages that result from a logical analysis of a schema:

#### Return Code

#### Message

#### L100

COMPONENT neither inherited nor local: *'type name'* of TYPE: *'type name'*

Hint: Look at the super-type's component list. The component is a valid type, but the supertype has a component list that restricts you from using this type as a component. You may have added subtype



components before adding supertype components. Either remove all supertype components or add the components in error to the component list of the supertype.

**L101**

This GROUP must have at least one component - TYPE: *'type name'*

Hint: If you want to map this group, add components. If you do not want to map it, make it a category.

**L102**

Circular reference found in COMPONENT # (*'type name'*) - TYPE: *'type name'*

Hint: Look at the type of the component in error. It is probably missing an initiator or terminator.

**L103**

Circular reference found in Floating Component type - TYPE: *'type name'*

Hint: Look at the floating component type in error. It is probably missing an initiator or terminator.

**L104**

DELIMITER for TYPE - *'type name'* must have a value

Hint: All delimited groups need a delimiter. Edit delimited group properties to insert the missing delimiter.

**L105**

DELIMITER type neither inherited nor local - TYPE: *'type name'*

Hint: The delimiter name has been entered incorrectly. It should be the name of a local type, or the name of an inherited delimiter, or the name of a type in the sub-tree of the inherited delimiter.

**L106**

Default DELIMITER not specified - TYPE: *'type name'*

Hint: This Type was specified with a FIND option for its delimiter. Please add a default value to define what to use for building outputs.

**L107**

Default DELIMITER not in restriction list - TYPE: *'type name'*

Hint: This delimiter was specified as a syntax item. Add the default value to the restriction list for that syntax item.

**L108**

DELIMITER type is not a SYNTAX ITEM - TYPE: *'type name'*

Hint: Delimiters specified as an item must be specified to be interpreted as SYNTAX to set the value of the delimiter if it appears as a component in a data stream.

**L109**

DELIMITER type has no restriction list - TYPE: *'type name'*

Hint: All syntax items need a restriction list.

**L110**

RELEASE CHARACTER neither inherited nor local - TYPE: *'type name'*

Hint: The release character name has been entered incorrectly. It should be either the name of a local type, the name of an inherited release character, or the name of a type in the sub-tree of the inherited release character.

**L111**

Default RELEASE CHARACTER not specified - TYPE: *'type name'*

Hint: This Type was specified with a syntax item for its release character. Please add a default value to define a value for the release character that has not been encountered in the data.

**L112**

Default RELEASE CHARACTER not in restriction list - TYPE: *'type name'*

Hint: This Type was specified with a syntax item for its release character. Please add the default value to the restriction list of that syntax item.

**L113**

RELEASE CHARACTER type is not a SYNTAX ITEM - TYPE: *'type name'*

Hint: Release characters specified as an item must be specified to be interpreted as SYNTAX to set the value of the release character if it appears as a component in a data stream.

**L114**

RELEASE CHARACTER type has no restriction list - TYPE: *'type name'*

Hint: All syntax items need a restriction list.

**L115**

Floating Component TYPE neither inherited nor local - TYPE: *'type name'*

Hint: The floating component name has been entered incorrectly. It should be either the name of a local type, the name of an inherited floating component, or the name of a type in the sub-tree of the inherited floating component.

**L116**

INITIATOR type neither inherited nor local - TYPE: *'type name'*

Hint: The initiator name has been entered incorrectly. It should be either the name of a local type, the name of an inherited initiator, or the name of a type in the sub-tree of the inherited initiator.

**L117**

Default INITIATOR not specified - TYPE: *'type name'*

Hint: This Type was specified with a syntax item for its initiator. Add a default value to define a value for that initiator has not been encountered in the data.

**L118**

Default INITIATOR not in restriction list - TYPE: *'type name'*

Hint: This Type was specified with a syntax item for its initiator. Add the default value to the restriction list of that syntax item.

**L119**

INITIATOR type is not a SYNTAX ITEM - TYPE: *'type name'*

Hint: Initiators specified as an item must be specified to be interpreted as SYNTAX to set the value of the initiator if it appears as a component in a data stream.

**L120**

INITIATOR type has no restriction list - TYPE: *'type name'*

Hint: All syntax items need a restriction list.

**L121**

TERMINATOR type neither inherited nor local - TYPE: *'type name'*

Hint: The terminator name has been entered incorrectly. It should be either the name of a local type, the name of an inherited terminator, or the name of a type in the sub-tree of the inherited terminator.

**L122**

Default TERMINATOR not specified - TYPE: *'type name'*

Hint: This Type was specified with a syntax item for its terminator. Add a default value to define a value for that terminator has not been encountered in the data.

**L123**

Default TERMINATOR not in restriction list - TYPE: *'type name'*

Hint: This Type was specified with a syntax item for its terminator. Please add the default value to the restriction list of that syntax item.

**L124**

TERMINATOR type is not a SYNTAX ITEM - TYPE: *'type name'*

Hint: Terminators specified as an item must be specified to be interpreted as SYNTAX to set the value of the terminator if it appears as a component in a data stream.

**L125**

TERMINATOR type has no restriction list - TYPE: *'type name'*

Hint: All syntax items need a restriction list.

**L126**

COMPONENT range minimum (#) greater than range maximum (#) - COMPONENT *'type name'* - TYPE: *'type name'*

Hint: The minimum range must be less than or equal to the maximum range.

**L127**

COMPONENT range minimum (#) less than inherited range minimum(#) - COMPONENT *'type name'* - TYPE: *'type name'*

Hint: The component in error has been inherited. Look at the range of the component with the same name in the super-type's component list.

**L128**

COMPONENT range maximum (#) greater than inherited range maximum(#) - COMPONENT *'type name'* - TYPE: *'type name'*

Hint: The component in error has been inherited. Look at the range of the component with the same name in the super-type's component list.

**L129**

COMPONENT RULE references a COMPONENT later in the component list - *'type name'* - TYPE: *'type name'*

Hint: Move the component rule to the component later in the list.

**L130**

COMPONENT RULE references undefined type - COMPONENT # of TYPE: *'type name'*

Hint: Verify the spelling of the data object name. The rule should reference a data object name of the component or a data object name of a component earlier in the component list.

**L131**

COMPONENT RULE references components of a partitioned group - COMPONENT # of TYPE *type name*

**L132**

Invalid partitioning: TYPE has no SUBTYPES - TYPE: *'type name'*

Hint: Remove the partitioned option from the class window or add sub-types to the Type in error.

**L133**

Type of COMPONENT exists, but its relative name is not valid: *'type name'* in TYPE: *'type name'*

Hint: To get the correct relative name, drag the type you want to use as a component and drop it in the component list of the Type. (Remember to delete the invalid component!)

**L134**

Reference to `ANY` not allowed: COMPONENT number # of TYPE: `type name`

Hint: In this case, the Type in error is a group and it is not the root of a partitioned tree. ANY cannot be used if that component needs to be validated. So, if that group is partitioned, you cannot use ANY for a component up to and including the identifier (if there is one). If that group is not partitioned, you cannot use ANY at all.

**L135**

COMPONENT number # cannot reference a CATEGORY in TYPE: `type name` (because group is not partitioned)

Hint: In this case, the Type in error is a group and it's not the root of a partitioned tree. A category cannot be used if the component must be validated. So, if that group is partitioned, you cannot use a category for a component up to and including the identifier (if there is one). If that group is not partitioned, you cannot use a category as a component at all.

**L136**

COMPONENT `type name` occurs more than once in list - TYPE: `type name`

Hint: Each component in the same component list must have a unique type name. Try to make sub-types of the type name in error and replace each non-unique component with one of the new sub-types.

**L137**

COMPONENT `type name` and its super-type cannot be in same COMPONENT LIST (in TYPE: `type name`)

Hint: Try making another sub-type of the super-type and replace the super-type reference with the new sub-type.

**L138**

COMPONENT `type name` is same type as delimiter - TYPE: `type name`

Hint: A component and a delimiter cannot have the same name. You may need to add sub-types to the type name used in error to resolve this one.

**L139**

COMPONENT `type name` is sub-type of delimiter - TYPE: `type name`

Hint: This occurs when a syntax item is used to specify a delimiter. You can add another sub-type to the syntax item and replace the delimiter name with the new sub-type name.

**L140**

COMPONENT `type name` is super-type of delimiter - TYPE: `type name`

Hint: This occurs when a syntax item is used to specify a delimiter. You can add another sub-type to the syntax item and replace the component name with the new sub-type name.

**L141**

COMPONENT *'type name'* is same type as initiator - TYPE: *'type name'*

Hint: A component and an initiator cannot have the same name. You can add sub-types to the type name used in error and replace both the component name and the initiator name.

**L142**

COMPONENT *'type name'* is sub-type of initiator - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify an initiator. You can add another sub-type to the syntax item and replace the initiator name with the new sub-type name.

**L143**

COMPONENT *'type name'* is super-type of initiator - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify an initiator. You can add another sub-type to the syntax item and replace the component name with the new sub-type name.

**L144**

COMPONENT *'type name'* is same type as terminator - TYPE: *'type name'*

Hint: A component and a terminator cannot have the same name. Try adding sub-types to the type name used in error and replace both the component name and terminator name with one of the new sub-types.

**L145**

COMPONENT *'type name'* is sub-type of terminator - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify a terminator. You can add another sub-type to the syntax item and replace the terminator name with the new sub-type name.

**L146**

COMPONENT *'type name'* is super-type of terminator - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify a terminator. You can add another sub-type to the syntax item and replace the component name with the new sub-type name.

**L147**

COMPONENT *'type name'* is same type as Floating Component - TYPE: *'type name'*

Hint: Make both the floating component name and the component name sub-types of the floating component.

**L148**

COMPONENT *'type name'* is sub-type of Floating Component - TYPE: *'type name'*

Hint: Make both the floating component name and the component name sub-types of the floating component.

**L149**

COMPONENT *'type name'* is super-type of Floating Component - TYPE: *'type name'*

Hint: Make both the floating component name and the component name sub-types of the floating component.

**L150**

COMPONENT *'type name'* is same type as release character - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify a release character. You can add sub-types to the syntax item and replace both the component name and the release character name with the new sub-type names.

**L151**

COMPONENT *'type name'* is sub-type of release character - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify a release character. You can add another sub-type to the syntax item and replace the release character name with the new sub-type name.

**L152**

COMPONENT *'type name'* is super-type of release character - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify a release character. You can add sub-types to the syntax item and replace the component name with the new sub-type name.

**L153**

DELIMITER *'type name'* is same type as initiator - TYPE: *'type name'*

Hint: A delimiter and an initiator cannot have the same name. You may need to add sub-types to the type name used in error and replace both the delimiter and initiator names to refer to the new sub-types.

**L154**

DELIMITER *'type name'* is sub-type of initiator - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both an initiator and a delimiter. You can add another sub-type to the syntax item and replace the initiator name with the new sub-type name.

**L155**

DELIMITER *'type name'* is super-type of initiator - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both an initiator and a delimiter. You can add another sub-type to the syntax item and replace the delimiter name with the new sub-type name.

**L156**

DELIMITER *'type name'* is same type as terminator - TYPE: *'type name'*

Hint: A delimiter and a terminator cannot have the same name. You may need to add sub-types to the type name used in error and replace both the delimiter and terminator names to refer to the new sub-types.

**L157**

DELIMITER *'type name'* is sub-type of terminator - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both a delimiter and a terminator. You can add another sub-type to the syntax item and replace the terminator name with the new sub-type name.

**L158**

DELIMITER *'type name'* is super-type of terminator - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both a delimiter and a terminator. You can add another sub-type to the syntax item and replace the delimiter name with the new sub-type name.

**L159**

DELIMITER *'type name'* is same type as release character - TYPE: *'type name'*

Hint: A delimiter and a release character cannot have the same name. You may need to add sub-types to the type name used in error and replace both the delimiter and release character names to refer to the new sub-types.

**L160**

DELIMITER *'type name'* is sub-type of release character - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both a delimiter and a release character. You can add another sub-type to the syntax item and replace the release character name with the new sub-type name.

**L161**

DELIMITER *'type name'* is super-type of release character - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both a delimiter and a release character. You can add another sub-type to the syntax item and replace the delimiter name with the new sub-type name.

**L162**

DELIMITER *'type name'* is same type as Floating Component - TYPE: *'type name'*

Hint: A delimiter and a floating component cannot have the same name. Try adding sub-types to the type name used in error and replace both the delimiter and floating component names to refer to the new sub-types.

**L163**

DELIMITER *'type name'* is sub-type of Floating Component - TYPE: *'type name'*



Hint: This occurs when a syntax item is used to specify both a delimiter and a floating component. You can add another sub-type to the syntax item and replace the floating component name with the new sub-type name.

**L164**

DELIMITER *'type name'* is super-type of Floating Component - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both a delimiter and a floating component. You can add another sub-type to the syntax item and replace the delimiter name with the new sub-type name.

**L165**

INITIATOR *'type name'* is same type as terminator - TYPE: *'type name'*

Hint: An initiator and a terminator cannot have the same name. You may need to add sub-types to the type name used in error and replace both the initiator and terminator names to refer to the new sub-types.

**L166**

INITIATOR *'type name'* is sub-type of terminator - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both an initiator and a terminator. You can add another sub-type to the syntax item and replace the terminator name with the new sub-type name.

**L167**

INITIATOR *'type name'* is super-type of terminator - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both an initiator and a terminator. You can add another sub-type to the syntax item and replace the initiator name with the new sub-type name.

**L168**

INITIATOR *'type name'* is same type as release character - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both an initiator and a release character. You can add sub-types to the syntax item and replace both the initiator name and the release character name with a new sub-type name.

**L169**

INITIATOR *'type name'* is sub-type of release character - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both an initiator and a release character. You can add another sub-type to the syntax item and replace the release character name with the new sub-type name.

**L170**

INITIATOR *'type name'* is super-type of release character - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both an initiator and a release character. You can add another sub-type to the syntax item and replace the initiator name with the new sub-type name.

**L171**

INITIATOR *'type name'* is same type as Floating Component - TYPE: *'type name'*

Hint: An initiator and a floating component cannot have the same name. Try adding sub-types to the type name used in error and replace both the initiator and floating component names with the new sub-types.

**L172**

INITIATOR *'type name'* is sub-type of Floating Component - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both an initiator and a floating component. You can add another sub-type to the syntax item and replace the floating component name with the new sub-type name.

**L173**

INITIATOR *'type name'* is super-type of Floating Component - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both an initiator and a floating component. You can add another sub-type to the syntax item and replace the initiator name with the new sub-type name.

**L174**

TERMINATOR *'type name'* is same type as release character - TYPE: *'type name'*

Hint: A terminator and a release character cannot have the same name. You may need to add sub-types to the type name used in error and replace both the terminator and release character names with the new sub-types.

**L175**

TERMINATOR *'type name'* is sub-type of release character - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both a terminator and a release character. You can add another sub-type to the syntax item and replace the release character name with the new sub-type name.

**L176**

TERMINATOR *'type name'* is super-type of release character - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both a terminator and a release character. You can add another sub-type to the syntax item and replace the terminator name with the new sub-type name.

**L177**

TERMINATOR *'type name'* is same type as Floating Component - TYPE: *'type name'*

Hint: A terminator and a floating component cannot have the same name. You may need to add sub-types to the type name used in error and replace both the terminator and floating component names with the new sub-types.

**L178**

TERMINATOR *'type name'* is sub-type of Floating Component - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both a terminator and a floating component. You can add another sub-type to the syntax item and replace the floating component name with the new sub-type name.

**L179**

TERMINATOR *'type name'* is super-type of Floating Component - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both a terminator and a floating component. You can add another sub-type to the syntax item and replace the terminator name with the new sub-type name.

**L180**

RELEASE CHARACTER *'type name'* is same type as Floating Component - TYPE: *'type name'*

Hint: A release character and a floating component cannot have the same name. You may need to add sub-types to the type name used in error and replace both the release character and floating component names to refer to the new sub-types.

**L181**

RELEASE CHARACTER *'type name'* is sub-type of Floating Component - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both a release character and a floating component. You can add another sub-type to the syntax item and replace the floating component name with the new sub-type name.

**L182**

RELEASE CHARACTER *'type name'* is super-type of Floating Component - TYPE: *'type name'*

Hint: This occurs when a syntax item is used to specify both a release character and a floating component. You can add another sub-type to the syntax item and replace the release character name with the new sub-type name.

**L183**

COMPONENT NAME ambiguous: *'type name'* in TYPE: *'type name'*

Hint: This type has a component whose relative name can be associated with more than one type in the schema. Rename the conflicting types.

**L184**

RESTRICTION longer than max TYPE size - RESTRICTION # of TYPE: *'type name'*

Hint: The Type in error is an item. Either change the maximum size of the item or remove the restriction.

**L185**

RESTRICTION used in an earlier partition - RESTRICTION # of TYPE: *`type name`*

Hint: Item Partitions must have mutually exclusive restrictions. Remove the restriction from one of the partition restriction lists.

**L186**

Type of COMPONENT does not exist - *`type name`* in TYPE: *`type name`*

Hint: You probably entered an incorrect type name. Try the drag and drop approach to get the correct one.

**L187**

TYPE must be partitioned (since in a partitioned tree and has sub-types) - TYPE: *`type name`*

Hint: All types in a partitioned sub-type must have mutually exclusive data objects. Set the partitioned property for the type in error.

**L188**

TYPE is FIXED, COMPONENT # must have a maximum range value - TYPE *type name*

**L189**

TYPE is FIXED, but COMPONENT # is not fixed - TYPE: *`type name`*

Hint: If the component is not intended to be fixed in size, change the group format for the Type to implicit. If the group format is intended to be fixed, check the component: if that component is an item, make sure it has a Padded To length; if that component is a group, change its type to be of fixed syntax.

**L190**

BINARY text ITEM used as COMPONENT neither FIXED nor SIZED - COMPONENT # of TYPE: *`type name`*

Hint: The size of a binary text item must either have a Padded To length or it must be sized by the previous component.

**L191**

COMPONENT with SIZED attribute is not an UNSIGNED INTEGER ITEM TYPE - COMPONENT # of TYPE: *`type name`*

Hint: A component used to size the component that follows it must be defined as an unsigned integer item type.

**L192**

The last COMPONENT in the COMPONENT LIST may not have a SIZED attribute: TYPE: *`type name`*

Hint: Specify a component to follow the one with the sized attribute.

**L193**

Range of COMPONENT # must have a maximum value to indicate how many placeholders are needed for its series in TYPE: `type name`.

Hint: Change the range maximum to a specific value (not "s") if you may re-define the data this way.

**L194**

Cannot distinguish delimiter from terminator in TYPE: `type name`.

Hint: Make the range of the last component in the type fixed or make the delimiter of the type different from its terminator.

**L195**

Cannot distinguish delimiter contained in COMPONENT # from terminator of TYPE: `type name`.

Hint: Make that component bound or make that contained delimiter different from the type terminator.

**L196**

Cannot distinguish delimiter of COMPONENT # from delimiter of TYPE: *type name* because COMPONENT # has no placeholder.

Hint: Make that component bound or make that component's delimiter different from the type delimiter.

**L197**

Cannot distinguish delimiter of COMPONENT # from delimiter of TYPE: *type name* because COMPONENT # has no range maximum.

Hint: Make that component bound, or make that component's delimiter different from the type delimiter, or specify a range maximum that has a specific value (not "s") for the last component of COMPONENT #.

**L198**

Cannot distinguish delimiter contained in COMPONENT # from delimiter of TYPE: `type name`.

Hint: Make that contained component bound or make that contained component's delimiter different from the type delimiter.

**L200**

Cannot distinguish delimiter contained in COMPONENT # from delimiter of TYPE: `type name`.

Hint: Either make that contained component bound, make that contained component's delimiter different from the type delimiter, or specify a range maximum that has a specific value (not "s") for the last component of the contained component.

## Logic error and warning messages

The tables in this section list the logic warning messages that result from a logic analysis of a schema.

The following table lists the warnings than can result when a map is compiled.

Warnings should be resolved because they may produce unpredictable results at mapping time.

**Return Code**

**Message**

**L199**

COMPONENT # is not distinguishable from COMPONENT # that may follow in TYPE: `type name`.

Hint: Make the first COMPONENT bound, or look at the tables in ["Distinguishable objects" on page 714](#) to see how you can define the two component types as distinguishable.

**L201**

Different data objects of COMPONENT # are not distinguishable in TYPE: `type name`.

Hint: See ["Distinguishable objects" on page 714](#) for more information about distinguishable objects.

**L202**

RESTRICTION list deleted: TYPE is not an ITEM - TYPE: `type name`

Hint: Type class was changed from an item to a group or category, so the restriction list was deleted. If this was not your intent, change it back to the way it was.

**L203**

COMPONENT list deleted: TYPE is an ITEM - TYPE: `type name`

Hint: Type class was changed from a group to a item or category, so the program deleted its component list. If this was not your intent, change it back to the way it was.

**L204**

DELIMITER deleted: TYPE is not a DELIMITED GROUP - TYPE: `type name`

Hint: Group format was changed from delimited to something else, so the program deleted its delimiter. If this was not your intent, change it back to the way it was.

**L205**

COMPONENT RULE deleted: TYPE is a CATEGORY - TYPE: `type name` (warning)

Hint: Type class was changed from a group to a category, so its component rule was deleted. If this was not your intent, change it back to the way it was.

**L206**

DELIMITER cannot be found (because first component is not required) - TYPE: `type name` (warning)

Hint: If the delimiter is missing, a previously set initiator value or the default value is used.

**L251**

COMPONENT NAME could apply to more than one type:'type name' in TYPE: `type name` (warning).

## Schema analysis structure error messages

The following table lists the structure error messages that result from a structural analysis of a schema:

### Return Code

#### Message

#### S100

Invalid TYPE Name: SubTYPE # of TYPE: *'type name'*

#### S101

Invalid TYPE chain: SubTYPE # of TYPE: *'type name'*

#### S118

Invalid TYPE NAME WhereUsed chain - TYPE NAME: *'type name'* (error).

#### S133

Referenced COMPONENT not 'InUse' - COMPONENT # of TYPE: *'type name'* (error).

#### S134

COMPONENT previously referenced - COMPONENT # (COMP #) of TYPE: *'type name'* (error).

#### S149

Bad Parent COMPONENT Index - COMPONENT *'type name'* - TYPE: *'type name'* (error)

## Schema analysis structure warning messages

The following table lists the structure warning messages that result from a structural analysis of a schema:

### Return Code

#### Message

#### S102

Unused DELIMITER deleted: *'type name'* (at index #)

#### S103

Invalid DELIMITER pointer deleted - TYPE: *'type name'*

#### S104

Invalid default DELIMITER pointer deleted - TYPE: *'type name'*

#### S105

Invalid RELEASE Char pointer deleted - TYPE: *'type name'*

#### S106

Invalid default RELEASE Char pointer deleted - TYPE: *'type name'*

**S107**

Invalid INITIATOR pointer deleted - TYPE: `type name`

**S108**

Invalid default INITIATOR pointer deleted - TYPE: `type name`

**S109**

Invalid TERMINATOR pointer deleted - TYPE: `type name`

**S110**

Invalid default TERMINATOR pointer deleted - TYPE: `type name`

**S111**

Resetting DELIMITER Use Count (was # now #) - DELIMITER: `type name`

**S112**

Unused DESCRIPTION deleted: `type name` (at index #)

**S113**

Invalid DESCRIPTION pointer deleted - TYPE: `type name`

**S114**

Resetting DESCRIPTION Use Count (was # now #) - DESCRIPTION: `type name`

**S115**

Invalid Floating Component TYPE pointer deleted - TYPE: `type name`

**S116**

Invalid TYPE UsedInComp chain repaired - TYPE: `type name`

**S117**

Unused TYPE NAME deleted - TYPE NAME: `type name` (at index #)

**S119**

Resetting TYPE NAME use count (was # now #) - TYPE NAME: `type name`

**S120**

Repaired empty TYPE NAME WhereUsed chain - TYPE NAME: `type name`

**S121**

Unused RESTRICTION NAME deleted: `type name` (at index #)

**S122**

Invalid RESTRICTION NAME deleted no DESCRIPTION was available - TYPE: `type name` .

**S123**

Invalid RESTRICTION NAME deleted DESCRIPTION was `type name` - TYPE: `type name`



**S124**

Resetting RESTRICTION NAME Use Count (was # now #) - RESTRICTIONS: `type name`

**S125**

Unused RESTRICTION DESCRIPTION deleted: `type name` (at index #)

**S126**

Invalid RESTRICTION DESCRIPTION deleted - TYPE: `type name`

**S127**

Resetting RESTRICTION DESCRIPTION Use Count (was # now #) - RESTRICTIONS: `type name`

**S128**

Unused RULE deleted: `type name` (at index #)

**S129**

Invalid RULE pointer deleted - COMPONENT # of TYPE: `type name`

**S130**

Resetting RULE Use Count (was # now #) - RULE: `type name`

**S131**

Invalid COMPONENT TYPE Description pointer - COMPONENT #

**S132**

COMPONENT marked `InUse` found in Free Chain- COMPONENT #

**S135**

COMPONENT in Free Chain referenced by a TYPE - COMPONENT #

**S136**

COMPONENT recovered and added to Free Chain - COMPONENT #

**S137**

TYPE in Free Chain referenced by another TYPE - TYPE #

**S138**

TYPE recovered and added to Free Chain - TYPE X'04X'

**S139**

TYPE marked `InUse` but not referenced - TYPE #

**S140**

Referenced TYPE not marked `InUse` - TYPE #

**S141**

TYPE Free Chain not in order: sorting

**S142**

COMPONENT Free Chain not in order: sorting

**S143**

Overlap found in LIST Free Chain

**S144**

Free Chain extends into unallocated region

**S145**

Overlap found in COMPONENT LIST SPACE: list cleared COMPONENTS will be deleted

**S146**

Invalid COMPONENT LIST pointer: all COMPONENTS DELETED - TYPE: *'type name'*

**S147**

Resetting COUNT in COMPONENT LIST: some COMPONENTS may be lost

**S148**

RULE truncated (due to internal error): *'type name'* (at index #)

**S150**

CATEGORY *'type name'* was missing GROUP and/or ITEM attributes

**S151**

GROUP *'type name'* was missing GROUP attributes

**S152**

ITEM *'type name'* was missing ITEM attributes

## Compile-time error messages

When you generate the test data of a schema, you might encounter compile-time error messages. The schemas with compilation errors fail to generate the test data. You can view the error messages on the schema designer.

You can download the schema definition file (.mmc) to troubleshoot the compile-time errors from the following location of the HCL® OneTest™ Data pod:

```
/opt/hcl/hip-rest/maps/<genMapPath>
```

The following table lists the compile-time errors that can occur when you generate the test data:



**Note:** In the following messages, the characters  $x$  and  $y$  are used as variables:



- *x* = the map where the error occurred
- *y* = the output that displays the rule with error

Compile-time Error Message	Description
<pre>Map:x Output:y Output argument of rule does not match output item sub-class</pre>	<p>This error appears when you enter an invalid regular expression for an item type in <b>Item Properties</b> of the <b>Properties</b> dialog box.</p> <p>For example, If you select <b>Number</b> as <b>Item subclass</b> in <b>Item Properties</b> of the <b>Properties</b> dialog box and enter text as the value of the <b>Regular Expression</b> field, then you encounter this error.</p> <p>To resolve this error, enter the valid values for the regular expression.</p>
<pre>No such file or directory</pre>	<p>This error appears when you enter an incorrect filename for the <b>Values file</b> field while you set up the restrictions for an item type in the <b>Properties</b> dialog box.</p> <p>To resolve this error, enter the correct filename that you want HCL® OneTest™ Data to use to import the values of the item type.</p>
<pre>Map:x Output:y Rule references unknown</pre>	<p>This error appears when you apply an invalid function for any item type.</p> <p>To resolve this error, apply the correct function.</p>
<pre>Group &lt;component_name&gt; Root does not have any components (or subtypes if it is a partitioned group).</pre>	<p>This error appears when you define an invalid schema.</p> <p>To resolve this error, ensure that you correctly define the type definitions of the schema and its properties.</p>
<pre>There is a missing component in group &lt;component_name&gt; Root</pre>	<p>This error appears when the reference of any type definitions exists in the <b>Structure</b> dialog box and the definition of that type is missing in the <b>Dictionary</b> of the schema.</p> <p>For example, if in the <b>Structure</b> dialog box, you have a reference of an item type <b>Age</b>, and the definition of this item type is missing in the <b>Dictionary</b>, then you encounter this error.</p> <p>To resolve this error, you must delete the reference of the type from the <b>Structure</b> dialog box that does not exist in the dictionary.</p>

## Configuring log files

When you run into issues to trace the log files or the log files end abruptly without the complete information about the action, you must configure settings for the log file in the `configMap` file.

### Before you begin

- You must have installed HCL OneTest™ Server.
- You must have access to the Secure Shell (SSH) console.

### About this task

To troubleshoot the issue, you can configure the following settings:

- The maximum number of log files.
- The maximum file size of each log file.



**Note:** The default value of the maximum number of log files is 5 and the maximum file size of each log file is 10 MB.

1. Open the HCL® OneTest™ Data `configMap` file to edit by using the following command: `kubectl edit configmap -n {namespace} {my-ots}-data-config -o yaml`.



**Note:** You can use the following command to get the list of the `configMap` files: `kubectl get configmaps`

2. Modify the values of the following parameters that you want to change: **log\_file\_size** and **max\_no\_of\_log\_files**
3. Save the changes.
4. Restart the `onetestdata` pod by using the following command: `kubectl delete pod {my-ots}-data-app-0`.

### Results

You have configured the settings for the log files. You have modified the maximum number of log files and the maximum size of each log file.

## Audit log overview

When execution of a map to generate the test data fails, HCL® OneTest™ Data creates an audit log. The audit log records the detailed information about the execution of the failed job.

You can download the audit log of the failed job from the **Jobs** page by clicking the **Download Log File** button.

### Audit log contents

The audit log provides an execution summary of the following information:

ExecutionSummary Fields	Description
CommandLine	Specifies the name and the location of the compiled map.
ElapsedSec	Specifies the total time spent on the map execution. The time spent is presented in seconds only (rounded). For example, <code>ElapsedSec="25"</code> .
mapreturn	Returns a code to specify the result of map execution. For example, <code>mapreturn="0"</code> .
MapStatus	Specifies the status of the map executed.
Message	Notifies the map execution result.
ObjectsBuilt	Specifies the number of objects generated based on the data. Each type in a schema represents an object.
ObjectsFound	Specifies the number of objects identified based on the data read. Each type in a schema represents an object.
SourceReport	Provides a detailed report about the source adapter.



**Note:** Some of the execution information is optional, and are not always displayed in the `ExecutionSummary` of the audit log.

## Source data report

The source data report provides the following information:

- Value of the source adapter
- Byte count of the source data
- Return code of the source adapter with a message
- Adapter command line or file path
- Time stamp of the data source

## Sample of an execution summary in an audit log file

The following sample is the execution summary of the audit log file.

```
<ExecutionSummary MapStatus="Valid" mapreturn="0" ElapsedSec="0.0373" BurstRestartCount="0"> <Message>Map
completed successfully</Message> <CommandLine>'install_dir\examples\CallsSummary.mmc'</CommandLine>
<ObjectsFound>18</ObjectsFound> <ObjectsBuilt>12</ObjectsBuilt> <SourceReport card="1" adapter="File"
bytes="52" adapterreturn="0"> <Message>Data read successfully</Message> <Settings>install_dir\examples
\stores.txt</Settings> <TimeStamp>18:10:04 December 26, 2019</TimeStamp> </SourceReport> <SourceReport card="2"
adapter="File" bytes="69" adapterreturn="0"> <Message>Data read successfully</Message> <Settings>install_dir
```

```

\examples\CALLS.TXT</Settings> <TimeStamp>18:10:04 December 26, 2019</TimeStamp> </SourceReport> <TargetReport
card="1" adapter="File" bytes="119" adapterreturn="0"> <Message>Data written successfully</Message>
<Settings>install_dir\examples\summary.txt</Settings> <TimeStamp>10:14:34 Dec 27, 2019</TimeStamp>
</TargetReport> <WorkArea type="File"> <inputarea card="1" Path="install_dir\examples\CallsSummary.I01"
TimeStamp="10:14:34 Dec 27, 2019" bytes="65695"/> <inputarea card="2" Path="install_dir\examples
\CallsSummary.I02" TimeStamp="10:14:34 Dec 27, 2019" bytes="65695"/> <outputarea card="1" Path="install_dir
\examples\CallsSummary.001" TimeStamp="10:14:34 Dec 27, 2019" bytes="65695"/> </WorkArea> </ExecutionSummary>

```

## Disabling audit logs

As a default configuration, HCL® OneTest™ Data generates audit logs for each failed test data generation job. However, you might want to disable the audit logs when you no longer require them. In such case, you must change the default configuration.

### Before you begin

- You must have cluster-admin permissions.
- You must have the IP address of the computer where HCL OneTest™ Server is installed.

### About this task

You must set the value of the **AUDIT\_ENABLED** property in the `configmap` file to disable the audit log for all the failed test data generation jobs.

1. Log in to the SSH console of HCL OneTest™ Server.
2. Run the following command to edit the configuration file:

```
kubectl edit configmap -n test-system {my-ots}-data-config -o yaml
```

3. Search for the **AUDIT\_ENABLED** property in the configuration file, and then set the value as *false*.



**Note:** The default value of **AUDIT\_ENABLED** is *true*.

4. Save your changes, and then exit from the configuration file.
5. Run the following command to delete and restart the `<onetest data>` pod:

```
kubectl delete pod {my-ots}-data-app-0
```

### Results

You have successfully disabled the audit logs for all the failed test data generation jobs.

## Commands used in HCL OneTest Data

When you want to view the list of HCL® OneTest™ Data pods and to manage each of these pods, you can run the `kubectl` or `oc` commands on an SSH console.

The following table provides you the list of commands that you can use to manage the pods on the Kubernetes environment (k3s):

Commands	Purpose	Examples
<code>kubectl get pods</code>	Shows the list of all pods with the status of each pod.	-
<code>kubectl get configmaps</code>	Shows the list of all configuration files.	-
<code>kubectl exec -it &lt;podname&gt; bash</code>	Helps to access the pod.	<code>kubectl exec -it {my-ots}-data-app-0 bash</code>
<code>kubectl edit configmap -n &lt;namespace&gt; &lt;configmapName&gt; -o yaml</code>	Edits the configuration file.	<code>kubectl edit configmap -n test-system {my-ots}-data-config -o yaml</code>
<code>kubectl delete pod &lt;podname&gt;</code>	Deletes and restart the pod.	<code>kubectl delete pod {my-ots}-data-app-0</code>
<code>kubectl logs &lt;podname&gt;</code>	Shows the logs of any specific pod.	<code>kubectl logs {my-ots}-data-app-0</code>

The following table provides you the list of commands that you can use to manage the pods on the Red Hat OpenShift server:

Commands	Purpose	Examples
<code>oc get pods</code>	Shows the list of all pods with the status of each pod.	-
<code>oc get configmaps</code>	Shows the list of all configuration files.	-
<code>oc exec -it &lt;podname&gt; bash</code>	Helps to access the pod.	<code>oc exec -it {my-ots}-data-app-0 bash</code>
<code>oc edit configmap -n &lt;namespace&gt; &lt;configmapName&gt; -o yaml</code>	Edits the configuration file.	<code>oc edit configmap -n test-system {my-ots}-data-config -o yaml</code>
<code>oc delete pod &lt;podname&gt;</code>	Deletes and restart the pod.	<code>oc delete pod {my-ots}-data-app-0</code>
<code>oc logs &lt;podname&gt;</code>	Shows the logs of any specific pod.	<code>oc logs {my-ots}-data-app-0</code>

## Troubleshooting installation issues

You might encounter issues when you access HCL® OneTest™ Data after you install HCL OneTest™ Server for the first time. You can find details about the issues, their causes, and resolutions to fix those issues.

Problem	Cause	Solution
<p>When you access HCL® OneTest™ Data after you install HCL OneTest™ Server on the Kubernetes environment (k3s) or Red Hat OpenShift server for the first time, you might encounter an error stating that the page cannot be loaded.</p>	<p>The pods such as <i>HIP Server</i> and <i>Data app</i> might have failed to run successfully.</p>	<p>You must restart the pods by using the following commands:</p> <p>For <i>HIP Server</i> on k3s environment</p> <pre>kubect1 delete pod {my-ots}-hip-server-0</pre> <p>For <i>HIP Server</i> on Red Hat OpenShift server</p> <pre>oc delete pod {my-ots}-hip-server-0</pre> <p>For <i>Data app</i> on k3s environment</p> <pre>kubect1 delete pod {my-ots}-data-app-0</pre> <p>For <i>Data app</i> on Red Hat OpenShift server</p> <pre>oc delete pod {my-ots}-data-app-0</pre> <p>Where <code>{my-ots}</code> is the name of the release that is provided during the installation of the server software.</p>
<p>When you generate a test data after you install HCL OneTest™ Server on the k3s environment or Red Hat OpenShift server for the first time, you might get a page that displays <i>Error</i>.</p>	<p>The pod such as <i>HIP Rest</i> might have failed to run successfully.</p>	<p>You must restart the pod on the k3s environment by using the following command:</p> <pre>kubect1 delete pod {my-ots}-hip-rest-0</pre> <p>Use the following command to restart the pod on the Red Hat OpenShift server:</p> <pre>oc delete pod {my-ots}-hip-rest-0</pre> <p>Where <code>{my-ots}</code> is the name of the release that is provided during the installation of the server software.</p>



# Security Considerations

This document describes the actions that you can take to ensure that your installation is secure, customize your security settings, and set up user access controls.

- [Enabling secure communication between multiple applications on page mxxi](#)
- [Ports, protocols, and services on page mxxi](#)
- [Customizing your security settings on page mxxi](#)
- [Setting up user roles and access on page mxxii](#)

## Enabling secure communication between multiple applications

The majority of communications are sent over TLS to port 443 (see [Ports, protocols, and services on page mxxi](#)). During the installation, an X.509 certificate is generated for the user provided DNS name, which is used to connect to the server. This certificate is self-signed and hence untrusted by other applications.

This self-signed certificate must be replaced by a certificate signed by a certificate authority trusted by your organization. For more information, see [X.509 Certificate User Authentication](#) in the Keycloak documentation.

For information about how the self-signed certificate was created, see the `ssl.sh` file in the `<install-directory>/prepare/` directory.

For information about importing a certificate authority trusted by your organization, see [Importing Certificate Authority into a browser](#).

## Ports, protocols, and services

TCP port 443 is used by the majority of communications with the server.

The port 7085 is the default port for communications with agents registered with HCL OneTest™ Server.

The ports starting from 7085, are used in pairs such as 7085 and 7086, and are allotted for the Schedule that is executed first. The next Schedule is allotted the next pair (7087,7088), and so on for the Schedules that are running simultaneously.

You must open the required ports in pairs for each of the Schedules that you want to run simultaneously.

## Customizing your security settings

You can customize your security settings through user registration.

### User registration

By default, users can sign up themselves with the server. In some environments, this self sign-up might be undesirable. It can be changed by switching off user registration. For more information, see [User Registration](#) in the Keycloak documentation.

By default, user email addresses are not verified. This verification must be enabled in production environments. For more information, see [Email settings on page 106](#).

## **Setting up user roles and access**

You can manage user roles and access through single sign on (SSO) and administration only accounts.

### **Single sign-on**

By default, Keycloak manages users and passwords locally. In production environments, it is normally appropriate to use single sign-on. For more information, see [LDAP user administration on page 106](#).

### **Administration only accounts**

Users in the Administrator group can manage the team space where they can configure licenses and notifications and add a repository to store the System model. For more information, see [Team space overview on page 128](#).

Users in the Administrator group can also discover all projects stored on the server (including private ones) and assign themselves and others roles in those projects.

For this reason, users who use the server to perform both administration and non-administration tasks must have two different accounts, one for each purpose. For more information, see [Default user administration on page 105](#).

# Notices

This document provides information about copyright, trademarks, terms and conditions for the product documentation.

© Copyright IBM Corporation 2000, 2016 / © Copyright HCL Technologies Limited 2016, 2021

This information was developed for products and services offered in the US.

HCL® may not offer the products, services, or features discussed in this document in other countries. Consult your local HCL® representative for information on the products and services currently available in your area. Any reference to an HCL® product, program, or service is not intended to state or imply that only that HCL® product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any HCL® intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-HCL® product, program, or service.

HCL® may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

*HCL*  
*330 Potrero Ave.*  
*Sunnyvale, CA 94085*  
*USA*  
*Attention: Office of the General Counsel*

For license inquiries regarding double-byte character set (DBCS) information, contact the HCL® Intellectual Property Department in your country or send inquiries, in writing, to:

*HCL*  
*330 Potrero Ave.*  
*Sunnyvale, CA 94085*  
*USA*  
*Attention: Office of the General Counsel*

HCL TECHNOLOGIES LTD. PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. HCL® may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-HCL® websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this HCL® product and use of those websites is at your own risk.

HCL® may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

*HCL*

*330 Potrero Ave.*

*Sunnyvale, CA 94085*

*USA*

*Attention: Office of the General Counsel*

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by HCL® under terms of the HCL® Customer Agreement, HCL® International Program License Agreement or any equivalent agreement between us.

The performance data discussed herein is presented as derived under specific operating conditions. Actual results may vary.

Information concerning non-HCL® products was obtained from the suppliers of those products, their published announcements or other publicly available sources. HCL® has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-HCL® products. Questions on the capabilities of non-HCL® products should be addressed to the suppliers of those products.

Statements regarding the future direction or intent of HCL® are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

#### COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to HCL®, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. HCL®, therefore, cannot guarantee or imply reliability,

serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. HCL® shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:

© (your company name) (year).

Portions of this code are derived from HCL Ltd. Sample Programs.

© Copyright HCL Ltd. 2000, 2021.

## Trademarks

HCL®, the HCL® logo, and hcl.com® are trademarks or registered trademarks of HCL Technologies Ltd., registered in many jurisdictions worldwide. Other product and service names might be trademarks of HCL® or other companies.

## Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

### Applicability

These terms and conditions are in addition to any terms of use for the HCL® website.

### Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of HCL®.

### Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of HCL®.

### Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

HCL® reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by HCL®, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

HCL® MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

# Index

## Special Characters

- [ ] reserved symbol 736
- @ reserved symbol 738
- < > reserved symbol 736
- \$
  - in a component rule 739

## Numerics

- 1 627

## A

- ABS function 903
- ABSENT function 847
- ACOSINE function 904
- adapter commands list for R/3 627
- adapter commands See R/3 adapter commands 627
- ADDDAYS function 803
- ADDHOURS function 804
- ADDMINUTES function 805
- ALE (Application Link Enabling) interface
  - definition 626
  - overview 629
- ALL function 862
- Allow Excess Trailing Pads property
  - pad characters 656
- AMS (Message Handler) interface See Message Handler (AMS) interface 626
- analysis
  - logic 713
  - structural 714
- analyzer
  - schemas 714
- ANY
  - reserved word 700
- Application Link Enabling (ALE) interface See ALE (Application Link Enabling) interface 626
- arguments
  - of a function 747
- arithmetic operators 744
- ASFUNCTION function 756
- ASIN function 904
- ATAN function 905
- ATAN2 function 905
- attributes
  - identifier 704
  - restart 704
  - sized 704
- attributes for components 704

## B

- BAPI (Business Application Programming Interface)
  - definition 627, 630
  - overview 630
- BCDTOHEX function 772
- BCDTOINT function 773
- BCDTOTEXT function 774, 918
- bidirectional data
  - symmetric swapping 641
- bidirectional functions
  - logical to visual reordering 765
  - SETLOGICALORDER 766
  - SETORIENTATIONLTR 767
  - SETORIENTATIONRTL 767
  - SETTEXTSHAPING 768
  - SETTEXTSHAPINGOFF 768

- SETVISUALORDER 768
  - text ordering 765
- bidirectional type property 669
- big endian 645, 645
- binary values 644, 666
  - BCD 645
  - float 644
  - integer 644
  - packed 644
- BINDABS function 862
- BOR (Business Object Repository) 630
- bound objects
  - in an expression 743
- bound types 715
- Business Application Programming Interface (BAPI) See BAPI (Business Application Programming Interface) 627, 630
- business framework 630
- Business Object Repository (BOR) 630
- business objects 630
- byte order property 645

## C

- card name
  - in an expression 736
- category
  - definition of 633
  - inheritance 710, 710
  - organizing subtypes 710
  - when not to use 711
- CDATA 694
- character size constraints 643
- character values 644, 666
  - decimal 646
  - integer 645
  - zoned 646
- character zoned numbers
  - places 646
  - sign 646
  - size (content) 646
- character-set encoding 755
- choice group
  - objects 732
- CHOOSE function 871
- class
  - changing 635
  - definition of 635
- CLONE function 758
- code pages 978
  - list of supported 659
- colon
  - in a component path 737
- comment name 738
- comments
  - component rules 703
- common item properties
  - NONE 665, 665, 668, 668, 678, 678
  - pad 653, 667
  - Required on input 665, 665, 669, 669, 678, 678
  - Special Value 665, 665, 668, 668, 678, 678
  - Zero 665, 665, 678, 678
- comparison operators 745
- component
  - in a component list 737
  - in an expression 737

- setting a range for 699
- component names 699
- component path 735, 736
- components 696
  - defining 698
  - defining rules for 701
  - distinguishable 721
    - implicit group 686
  - entering rules 702
  - guidelines 698
  - inheritance 710
  - name 698
  - optional 700
  - order of 696
  - range 696, 697
    - fixed 699
    - variable 699
  - required 700
  - rules 703
    - comments 703
    - examples 702
    - object names 702
    - shorthand notation 703
    - syntax 702
    - variable 700
- CONTAINSERRORS function 817, 848
- content distinguishable
  - components 722
- control records for IDoc mapping 629
- CONVERT function 775
- COSINE function 905
- COSINEH function 906
- COUNT function 702, 906
- COUNTABS function 907
- CountsTowardMinContent property 654
- COUNTSTRING function 919
- CSERIESTOTEXT function 978
- CSIZE function 921
- CTEXT function 922
- CURRENTDATE function 806
- CURRENTDATETIME function 807
- CURRENTTIME function 808

## D

- data
  - defining 632
  - object
    - classes 633
    - definition of 633
    - objects 633
    - types 633
  - objects
    - subtypes 632
    - size of 666
- data encoding 755
- data language type property 659
- data records of IDocs 630
- data segments of IDocs 630
- Data Transfer Object (DXOB) See DXOB (Data Transfer Object) 630
- date & time subclass properties 670
  - custom date format 673
  - custom time format 674
  - date 671
  - examples 677
  - format 672
  - presentation 670
  - time 671

- date format
    - custom 673
  - date property 671
  - DATETONUMBER function 776, 809
  - DATETOTEXT function 777, 810, 923
  - DBLOOKUP function 825, 872
  - DBQUERY function 828, 875
  - DDEQUERY function 831, 878
  - DEFAULT function 758
  - delimited syntax
    - explicit group 684
    - implicit group 686
  - delimiter 688
    - definition of 684
    - literal 687
    - location of 687
    - variable 687
  - description
    - types 636
  - distinguishable data
    - explicit group 732
    - implicit group 730
  - distinguishable objects 714
    - components 722
    - implicit group 686
    - of same component 721
    - types 722
  - document type 640
  - document verification 640
  - dollar sign 703
  - duplicates 699, 699
  - DXOB (Data Transfer Object) 630
    - definition 627
- E**
- ECHOIN function 759
  - EDI (Electronic Data Interchange) interface
    - definition 626
    - overview 629
  - EITHER function 863
  - Electronic Data Interchange (EDI) interface See EDI (Electronic Data Interchange) interface 626
  - element type declarations 694
  - Empty type property 639
  - encoding, changing 755
  - errors
    - messages 714
  - evaluation
    - of component rules 739
    - of expressions 739
    - of functions 747
    - of map rules 734, 740
      - influence of card order 740
      - influence of functions 741
      - influence of object names 743
    - of operands 747
  - examples
    - JEXIT function 840
    - propagating type properties 711, 712
    - using AcceptAllPads type property 655
    - using Empty type property 639
  - EXIT architecture 755
  - EXIT function 832
  - EXP function 907
  - explicit format 683
  - explicit group
    - distinguishable data 732
  - expressions
    - bound objects in 739
    - definition of 733
    - evaluating 739
    - operators in 744
    - shorthand notation in 739
    - external systems 625, 627, 630
    - EXTRACT function 879
- F**
- FACTORIAL function 908
  - FAIL function 818
  - FILLLEFT function 924
  - FILLRIGHT function 925
  - FIND function 926
  - fixed syntax 684
  - floating component 685
  - floating component type
    - in an expression 738
  - format
    - date & time 672
    - custom 673, 674
    - group
      - explicit 683, 684, 684
      - implicit 685, 686, 686
  - FROMBASETEN function 778, 908
  - FROMDATETIME function 779, 811
  - FROMNUMBER function 780
  - Function
    - used to convert output item value 752
  - functional maps
    - running as a function 756
  - functions
    - arguments 747
    - creating custom 961
    - custom 961
    - evaluation of 747
- G**
- GETANDSET function 880
  - GETDIRECTORY function 881
  - GETFILENAME function 883
  - GETLOCALE function 883
  - GETRESOURCEALIAS function 885
  - GETRESOURCENAME function 886
  - getting started
    - guide 19
  - GETTXINSTALLDIRECTORY function 887
  - GETXMLERRORMSG function 819
  - group
    - component
      - name 698
    - components 696
    - definition of 633
    - formats
      - explicit 683, 684, 684
      - implicit 685, 686, 686
    - group subclass 681
      - choice 681
      - sequence 681
      - unordered 682
    - guide
      - getting started 19
- H**
- HANDLEIN function 760
  - hex values 703
  - HEXTEXTTOSTREAM function 781, 927
  - hierarchy
    - of a schema 632
  - how to specify 695
- I**
- identifier attribute 704, 704
    - specifying a component 709
  - IDocs (Intermediate Documents)
    - control records 629
    - data records 630
    - data segments 630
    - definition 626
    - formats 629
    - overview 629
    - types 629
  - IF function 864
  - implicit group
    - distinguishable data 730
  - IN
    - in component path 738, 738
    - reserved word 735, 744
    - RESOLVETYPE function 764, 764
  - INDEX function 887
  - INDEXABS function 888
  - indexing
    - an input 736
  - inheritance
    - category 710
    - components 710
    - item properties 710
    - restrictions 710
  - initiator 636
  - input
    - indexing 736
  - INT function 782, 909
  - Intermediate Documents (IDocs) See IDocs (Intermediate Documents) 626
  - interpret as 644
  - introduction 625
  - ISALPHA function 850, 865
  - ISERROR function 820, 850
  - ISLOWER function 851, 866
  - ISNUMBER function 852, 867
  - ISUPPER function 853, 868
  - item
    - definition of 633
    - restrictions 694
  - item properties 643
    - date & time subclass properties 670
    - interpret as 644
    - item subclass 643
    - number item subclass 644
    - syntax objects 679
    - text item subclass properties 666
  - item subclass 643
- J**
- JEXIT function 838, 840
- L**
- language options 658, 668, 687
  - language support 659
  - LAST reserved word 737
  - LASTERRORCODE function 889
  - LASTERRORMSG function 890
  - LEAVEALPHA function 928
  - LEAVEALPHANUM function 928
  - LEAVENUM function 929
  - LEAVEPRINT function 930
  - LEFT function 931
  - length property
    - bytes 645
  - literal
    - delimiter value 687
  - literals
    - definition of 734
    - in expressions 734
  - little endian 645, 645
  - local type name 736



- location property 687
  - log file, generating 956, 957, 959, 960, 961
  - LOG function 909
  - LOG10 function 910
  - logic errors
    - type tree analysis 996
  - logic warnings
    - schema analysis 1009
  - logical analysis 713
  - logical operators 745
  - LOOKDOWN function 892
  - LOOKUP function 893
  - LOWERCASE function 932
- M**
- map rules
    - evaluation 734
  - master data 627
    - for IDocs 629
  - mathlib functions
    - ACOSINE 904
    - ASIN 904
    - ATAN 905
    - ATAN2 905
    - COSINE 905
    - COSINEH 906
    - EXP 907
    - FACTORIAL 908
    - LOG 909
    - LOG10 910
    - POWER 912
    - RAND 912
    - SIN 914
    - SINH 914
  - MAX function 812, 910, 932
  - MEMBER function 854, 894
  - Message Handler (AMS) interface
    - definition 626
  - message transaction IDs (TIDs) 627
  - messages
    - errors 714
    - warnings 714
  - MID function 933
  - MIN function 812, 911, 934
  - MOD function 911
- N**
- name
    - relative 698
    - type 698
  - names
    - data object 735
    - in a component rule 735
    - in a map rule 735
  - namespaces 693
  - national language 658, 668, 687
  - native 645, 645
  - native code page 755
  - no syntax group 686
  - NONE reserved word 749
    - assigned to an output number 752
    - when an input argument of a function evaluates to 750
    - when an input argument of a functional map evaluates to 751
    - when an operand evaluates to 750
  - nonprintable values 695
  - NORMXML function 935
  - SIZE function 859, 936
  - number item subclass property 644
    - byte order 645
    - length (bytes) 645
  - places 665
    - separators 647
    - sign 651
  - numbers
    - as literal 734
  - NUMBERTODATE function 782, 813
  - NUMBERTOTEXT function 783, 936
- O**
- object names 735
  - objects
    - distinguishable 714, 721
    - names
      - component rules 702
  - OFFSET function 856
  - ONERROR function 821
  - operands
    - evaluation of 747
  - operators
    - arithmetic 744
    - comparison 745
    - in expressions 744
    - logical 745
    - order of evaluation 746
  - optional component 700
  - orientation
    - examples 642
  - output items
    - conversion of 752
  - Overview 625
- P**
- PACK function 784
  - PACKAGE function 785, 937
  - pad character
    - specifying 713
  - pad characters 654
  - pad property 653, 654, 667
  - PARSE function 761
  - PARTITION function 857
  - partitioning 705, 705
    - benefits of 706
    - convenience 706
    - groups 708, 709, 709
      - component rules 710
    - items 707, 707, 708
    - required 705
    - types 636, 707, 732
  - partitions
    - in an expression 736
  - PCDATA 694
  - places property 665
  - POWER function 912
  - PRESENT function 858
  - presentation property 670
  - propagation of type properties 711, 711, 712
    - properties
      - inheritance of 710
      - item 669
      - national language 658, 668, 687
      - propagating 711, 711, 712
      - where used 640
  - Properties view 634
  - PUT function 842
- Q**
- QUIT function 823
- R**
- R/3 adapter commands list See adapter commands list for R/3 627
  - R/3 adapters 627
    - overview 627
  - R/3 interfaces 625
  - RAND function 912
  - RANDDATA function 938
  - range
    - default 697
    - examples of 700
    - fixed 699
    - variable 699
  - range restrictions 695
    - value not in range 695
  - REFORMAT function 763
  - REJECT function 822
  - relative type name 698
  - relative type names 699
  - release characters
    - definition of 637
    - guidelines 638, 638
  - RENAMEFILE function 845
  - required component 700
  - reserved symbol 737
  - reserved words and symbols 736, 737, 739
    - @ 738
    - < > 736
    - \$ 739
    - ANY 700
    - COMPONENT 736
    - LAST 737
  - RESOLVETYPE function 764, 764
  - resourcelib functions
    - GETLOCALE 883
    - GETRESOURCEALIAS 885
    - GETTXINSTALLDIRECTORY 887
  - restart attribute 704, 704
  - restrictions
    - inheritance of 710
    - settings 694
  - REVERSEBYTE function 939
  - RFCs
    - adapters 627
  - RIGHT function 939
  - ROUND function 913
  - rules
    - component 701, 702
      - examples 702
      - syntax 702
    - defining 701
    - entering 702
  - RUN function 843, 974
- S**
- SAP
    - R/3 interfaces 626
  - SAP Integration Pack
    - R/3 interfaces 627
  - SAP Integration Package
    - about 625
    - components 627
    - R/3 interfaces 625
  - SAP R/3 interfaces 627, 630
  - SAP R/3 systems
    - exchanging data 626
  - schema analysis
    - logic errors 996
    - structure errors 1011
    - structure warnings 1011
  - schema analyzer 713
  - schemas
    - analyzer 713
    - analyzing logic 713

- analyzing structure 714
- handling large schemas 696
- hierarchy 632
- SEARCHDOWN function 895
- SEARCHUP function 896
- separators 647
  - decimal 649
  - integer 647
- SERIESTOTEXT function 786, 941
- set range 699
- SETLOGICALORDER function 766
- SETOFF function 769
- SETON function 770
- SETORIENTATIONLTR function 767
- SETORIENTATIONRTL function 767
- SETTEXTSHAPING function 768
- SETTEXTSHAPINGOFF function 768
- SETVISUALORDER function 768
- shaping
  - examples 642
- shorthand notation 739
  - component rules 703
  - in a map rule 739
- sign property 646, 651
- SIN function 914
- SINH function 914
- size 666
  - command
    - include self 705
    - digits 646
- SIZE function 859, 921
- NORMXML function 859, 936
- sized attribute 704
  - guidelines 704
- SORTDOWN function 897
- SORTDOWNBY function 899
- SORTUP function 900
- SORTUPBY function 901
- SQRT function 914
- STREAMTOHEXTEXT function 788
- structural analysis 714
- structure errors
  - schema analysis 1011
- structure warnings
  - schema analysis 1011
- subtree 635, 698, 709, 709
- subtypes 632
  - order 636
- SUM function 702, 915
- supertype
  - definition of 632
- SYMBOL function 789
- symbols
  - inserting 695
- symmetric swapping
  - examples 641
- syntax
  - initiator 636
  - none 686
  - objects 679
    - data 680
    - defining a variable syntax object 679, 679
    - example 679, 680
    - variable values 679, 679
  - of a component rule 702
  - release character 637
  - terminator 637

**T**

- terminator 637

- TESTOFF function 770, 859
- TESTON function 771, 860
- text
  - as a literal 734
- TEXT function 944
- text functions
  - RANDDATA 938
  - RENAMEFILE 845
- text item subclass properties 666
  - interpret as 644, 666
- TEXTTOBCD function 790, 945
- TEXTTODATE function 791, 814
- time format
  - custom 674
- time property 671
- time zones 675
- TIMETOTEXT function 794, 816, 948
- TOBASETEN function 795, 916
- TODATETIME function 796, 817, 948
- TONUMBER function 797, 949
- trace log file, generating 956, 957, 959, 960, 961
- trace option
  - about content length 654
- track property 683
- transaction IDs
  - for messages 627
- transactional Remote Function Calls (tRFCs) 625
- tRFCsSee transactional Remote Function Calls (tRFCs) 625
- TRIMLEFT function 951
- TRIMRIGHT function 952
- TRUNCATE function 917
- type definitions 713
- type names
  - local 736
- type properties 633
  - accessing 634
  - defining 634
  - propagating 711, 711, 712
- type tree analysis
  - logic warnings 1009
- type trees
  - analyzer 714
  - analyzing 713
  - handling large schemas 696
- types
  - bound 715
  - category 633
  - class 635
  - classes of 633
  - definition of 633
  - description 636
  - distinguishable 722
  - groups 633, 696
  - inheritance 710
  - item 633
  - name 634, 698
  - organizing under a category 710
  - partitioned 636
  - propagating 711, 711, 712
  - properties 711, 711, 712
  - relative name 698
  - subtypes 632
  - supertypes 632

**U**

- unbound restrictions 695
- Unicode support 659
- UNIQUE function 902

- UNPACK function 799
- UNZONE function 800
- UPPERCASE function 954

**V**

- VALID function 824, 861
- VALIDATE function 956, 956, 956, 956
- VALIDATEEX function 956, 956, 956, 957
- value not in range 695
- variable component name 700
- variable delimiter 687
- version compatibility 630

**W**

- warning messages 714
- WHEN function 702, 870
- where used 640
- whitespace syntax
  - build as 686
  - character set 686
- wildcards
  - element and attribute 694
- WORD function 954

**X**

- XML
  - identity constraints 693
  - Location setting for metadata 691
  - metadata location 691
  - normalizing a fragment 935
  - properties in the type tree 690
  - supported constructs 690
  - time zone formats 676
  - using Empty property 639
  - wildcards 694
  - Xerces 691
- XML DTD Importer 694
- XML functions 956, 956
- XML Schema
  - datatypes and type tree constructs 691
- XML Schema Importer 693, 694
- XML type property See document type 640
- XML validation error 819
- XML whitespace 935
- XMLLIB functions
  - VALIDATE 956, 956
  - VALIDATEEX 956, 957
  - XPATH 958, 958
  - XPATHEX 959, 959
  - XSLT 960, 960
  - XSLTEX 960, 961
- XPATH function 956, 956, 958, 958, 958, 958
- XPATHEX function 956, 956, 959, 959
- XSDL hint
  - Xerces schema 691
- xml:noNamespaceSchemaLocation
  - Xerces XML schema 691
- xml:schemaLocation
  - Xerces XML schema 691
- XSLT function 956, 956, 959, 959, 960, 960
- XSLTEX function 956, 956, 960, 961
- XVALIDATE function 956, 956, 957, 957

**Z**

- ZONE function 801