Oracle® Fusion Middleware

Oracle Data Integrator Tools Reference

12*c* (12.2.1.4.0)
E95626-02
September 2023

Oracle Fusion Middleware Oracle Data Integrator Tools Reference, 12*c* (12.2.1.4.0)

E95626-02

Contents

Preface

This guide describes how to use and develop Open Tools using Oracle Data Integrator to design integration scenarios.

Audience

This document is intended for Oracle Data Integrator application developers who will use Open Tools to design integration scenarios.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the following documents in Oracle Data Integrator Library:

- *Release Notes for Oracle Data Integrator Release Notes*
- *Understanding Oracle Data Integrator*
- *Developing Integration Projects with Oracle Data Integrator*
- *Installing and Configuring Oracle Data Integrator*
- *Upgrading Oracle Data Integrator*
- Integrating Big Data with Oracle Data Integrator
- *Application Adapters Guide for Oracle Data Integrator*
- *Developing Knowledge Modules with Oracle Data Integrator*
- *Connectivity and Knowledge Modules Guide for Oracle Data Integrator*
- *Migrating From Oracle Warehouse Builder to Oracle Data Integrator*
- *Oracle Data Integrator Tools Reference*
- *Data Services Java API Reference for Oracle Data Integrator*

- *Open Tools Java API Reference for Oracle Data Integrator*
- *Getting Started with SAP ABAP BW Adapter for Oracle Data Integrator*
- *Java API Reference for Oracle Data Integrator*
- *Getting Started with SAP ABAP ERP Adapter for Oracle Data Integrator*
- *Oracle Data Integrator 12c Online Help*, which is available in ODI Studio through the JDeveloper Help Center when you press **F1** or from the main menu by selecting **Help**, and then **Search** or **Table of Contents**.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Using Oracle Data Integrator Open Tools

This appendix provides a reference of Oracle Data Integrator (ODI) tools. It describes how to use Open Tools to develop new scenarios in Oracle Data Integrator.
This appendix includes the following sections:

Using Oracle Data Integrator Tools

Oracle Data Integrator tools (also called Oracle Data Integrator commands) are commands provided for performing specific tasks at runtime. These tasks can be as simple as waiting for a certain time or producing a sound, or as sophisticated as executing Ant scripts or reading e-mail from a server.

Oracle Data Integrator tools are used in Packages, Procedure Commands, Knowledge Modules Commands, or directly from a command line.

> **Note:**
>
> Previous versions of Oracle Data Integrator supported calling built-in tools from Jython or Java scripts using their internal Java classes (such as `SnpsSendMail` and `SendMail`). This approach is no longer supported.

> **Note:**
>
> Carriage returns in commands are not permitted.

Using a Tool in a Package

Adding and using an Oracle Data Integrator tool in a Package is described in Adding Oracle Data Integrator Tool Steps in *Developing Integration Projects with Oracle Data Integrator*.

You can sequence the tool steps within the package and organize them according to their success and failure. For more information about sequencing, see Arranging the Steps Layout and Defining the Sequence of Steps in *Developing Integration Projects with Oracle Data Integrator*.

You can use variable values, sequences, or Oracle Data Integrator substitution method calls directly in tool parameters.

Using a Tool in a Knowledge Module or Procedure Command

Using an Oracle Data Integrator tool in a Knowledge Module or Procedure is described in Working with Procedures section in *Developing Integration Projects with Oracle Data Integrator* Guide.

You can use variable values, sequences, Oracle Data Integrator substitution method calls, or the results from a `SELECT` statement directly in tool parameters.

Using a Tool From a Command Line

Command line scripts for Oracle Data Integrator tools are run from the *DOMAIN_HOME*`/bin` directory. To run a tool from a command line, you must first create an ODI Physical Agent instance in the ODI Topology and configure an ODI Standalone Agent instance in a Domain. For more information about performing these tasks, see *Installing and Configuring Oracle Data Integrator*.

When you run a tool from a command line, you must specify the `-INSTANCE` parameter, where `<agent_name>` is the name of the physical agent you configured (for example, `OracleDIAgent1`).

To use an Oracle Data Integrator tool from a command line:

1. Launch the command shell for your environment (Windows or UNIX).
2. Navigate to the *DOMAIN_HOME*`/bin` directory.
3. Launch the `startcmd.cmd` (Windows) or `startcmd.sh` (UNIX) command and run an Oracle Data Integrator tool with the following syntax:

```
startcmd.<cmd|sh> -INSTANCE=<agent_name> <command_name>
[<command_parameters>]*
```

Command names and command parameters are case-sensitive.

Example 1-1 Important Notes

Note the following:

- On Windows platforms, command arguments that contain equal (=) signs or spaces must be surrounded with double quotation marks. This differs from the UNIX command call. For example:

```
startcmd.cmd OdiSleep "-INSTANCE=OracleDIAgent1" "-DELAY=5000"
./startcmd.sh OdiSleep -INSTANCE=OracleDIAgent1 -DELAY=5000
```

- The following tools do not support direct invocation through a command line:
  - OdiRetrieveJournalData
  - OdiRefreshJournalCount

Using Open Tools

The Open Tools feature provides an extensible platform for developing custom third-party tools that you can use in Packages and Procedures. As with the standard tools delivered with Oracle Data Integrator, Open Tools can interact with the operating system and manipulate data.

Open Tools are written in Java. Writing your own Open Tools is described in Developing Open Tools.

Open Tools are delivered as a Java package (`.zip` or `.jar`) that contains several files:

- A compiled Java `.class` file
- Other resources, such as icon files

Installing and Declaring an Open Tool

Before you can use an Open Tool, you must install and add it.

Installing an Open Tool

To install an Open Tool, you must add the Open Tool JAR into the classpath or the component using the tool.

Open Tool JARs must be added to the *DOMAIN_HOME*`/lib` directory. Drivers are added to the same location.

To deploy an Open Tool JAR with a Java EE agent, generate a server template for this agent. The Open Tool is displayed in the **Libraries and Drivers** list in the Template Generation Wizard. See Creating a Server Template for the Java EE Agent in *Administering Oracle Data Integrator* for more information.

> **Note:**
>
> This operation must be performed for each Oracle Data Integrator Studio from which the tool is being used, and for each agent that will run sessions using this tool.

Declaring a New Open Tool

This operation declares an Open Tool in a master repository and enables the tool to be displayed in Oracle Data Integrator Studio.

To declare an Open Tool, a JAR must be added in `<ide.user.dir>/oracledi/userlib`.

To declare a new tool:

1. In Oracle Data Integrator Studio, select the **ODI** menu and then select **Add Remove/ Open Tools**. The **Add Open Tools** dialog is displayed.
2. Enter the name of the class in the **Open Tool Class Name** field.

or:

1. Click **Find in the ClassPath**, then browse to the name of the Open Tool's Java class. To search for the class by name, enter part of the name in the field at the top.
2. Click **OK**.

   Note that all classes currently available to Oracle Data Integrator are displayed, including those that are not Open Tools. You must know the name of your class in order to add it.
3. Click **Add Open Tool**.
4. Select the line containing your Open Tool.
   - If the tool was correctly found on the classpath, the supplied icons and the tool's syntax, description, provider, and version number are displayed.
   - If the tool was not found, an error message is displayed. Change the classpath, or move the Open Tool to the correct directory.

> **Note:**
>
> This operation to declare a new Open Tool must be performed only once for a given master repository.

> **Note:**
>
> An Open Tool, a name cannot start with `Snp` or `Odi`. An Open Tool with a name that starts with these strings is ignored.

Using Open Tools in a Package or Procedure

You can use Open Tools in a Package or Procedure, similar to the tools provided with Oracle Data Integrator.

Developing Open Tools

An Open Tool is a Java package that contains a compiled Java class that implements the interface `oracle.odi.sdk.opentools.IOpenTool`. For a complete description of classes and methods, see the *Oracle Data Integrator Open Tools Java API Reference* (JavaDoc).

An Open Tool package typically should also contain two icons, which are used to represent the Open Tool in the Oracle Data Integrator graphical interface.

Classes

The following table lists and describes Open Tool classes and interfaces.

Class or Interface	Description
`IOpenTool`	Interface that every Open Tool must implement.
`OpenToolAbstract`	Abstraction of the interface with some helper methods. Preferably extend this class rather than implementing the interface directly.
`IOpenToolParameter`	Interface that parameters used by Open Tools must implement. In most cases, `OpenToolParameter` should be used rather than implementing this interface.
`OpenToolParameter`	Complete implementation of `IOpenToolParameter`. Each `OpenToolParameter` holds one parameter.
`OpenToolsExecutionException`	Exception class that should be thrown if necessary by Open Tool methods.
`SimpleOpenToolExample`	A simple example of an Open Tool, which can be used as a starting point.

Developing a New Open Tool

The following steps describe the development of a basic Open Tool, SimpleMessageBox. The source code for this class is available in the `demo/plugins/src` directory.

1. Define the syntax. In this example, the Open Tool is called as follows:

   ```
   SimpleMessageBox "-TEXT=<text message>" "-TITLE=<window title>"
   ```

2. Create 16x16 and 32x32 icons (usually in `.gif` format).
3. Create and implement the class. See Implementing the Class.
4. Compile the class and create a package with the two icon files.
5. Install and declare the Open Tool as described in Installing and Declaring an Open Tool.

Implementing the Class

Implementing the class consists of the following steps:

1. Declaration
2. Importing Packages
3. Defining the Parameters
4. Implementing Informational Functions
5. Execution

Declaration

Before you declare the class, you must name the package.

Naming the Package

Put the class in a package named appropriately. The package name is used to identify the Open Tool when installing it.

```
package com.myCompany.OpenTools;
```

Declaring the Class

There are two basic approaches to developing an Open Tool:

- Extend an existing class that you want to convert into an Open Tool. In this case, simply implement the interface `IOpenTool` directly on the existing class.
- Develop a new class. In this case, it is easiest to extend the abstract class `OpenToolAbstract`. This abstract class also contains additional helper methods for working with parameters.

  ```
  public class SimpleMessageBox extends OpenToolAbstract {
  ```

Importing Packages

Almost every Open Tool must import the following Open Tool SDK packages:

```
import oracle.odi.sdk.opentools.IOpenTool; /* All Open Tool classes need these three
classes */
```

```
import oracle.odi.sdk.opentools.IOpenToolParameter;

import oracle.odi.sdk.opentools.OpenToolExecutionException;

import oracle.odi.sdk.opentools.OpenToolAbstract; /* The abstract extended for
the Open Tool */

import oracle.odi.sdk.opentools.OpenToolParameter; /* The class used for
parameters */
```

In this particular example, a package to create the message box is also needed:

```
import javax.swing.JOptionPane; /* Needed for the message box used in this
example */
```

Defining the Parameters

Add a property to store the `OpenToolParameter` objects. This is used to both define them for the syntax, and to retrieve the values of the parameters from the eventual user. It is easiest to define the parameters of the Open Tool with a static array as follows. This array should be private, as it will be accessed through an accessor function.

```
private static final IOpenToolParameter[] mParameters = new IOpenToolParameter[]
{
    new OpenToolParameter("-TEXT", "Message text", "Text to show in the
messagebox (Mandatory).", true),
    new OpenToolParameter("-TITLE", "Messagebox title", "Title of the
messagebox.", false)
};
```

The four parameters passed to the `OpenToolParameter()` constructor are as follows:

1. The code of the parameter, including the initial hyphen. This code must correspond to the syntax returned by `getSyntax()`.
2. The user-friendly name, which is used if the user is using the graphical interface to set parameters.
3. A descriptive help text.
4. Whether the parameter is mandatory. This is an indication to the user.

> **Note:**
>
> Oracle Data Integrator does not enforce the mandatory flag on parameters. Your class must be able to handle any combination of parameters being provided.

You must implement the accessor function `getParameters()` to retrieve the parameters:

```
public IOpenToolParameter[] getParameters()
{
      return mParameters;
}
```

Implementing Informational Functions

Implement functions to return information about your Open Tool: `getDescription()`, `getVersion()`, `getProvider()`.

```
public String getDescription() { return "This Open Tool displays a message box when
executed."; }
public String getVersion() { return "v1.0"; }
public String getProvider() { return "My Company, Inc."; }
```

The `getSyntax()` function determines the name of the Open Tool as it is displayed in the Oracle Data Integrator graphical interface, and also the initial values of the parameter. Make sure the names of the parameters here match the names of the parameters returned by `getParameters()`.

```
public String getSyntax()
{
        return "SimpleMessageBox \"-TEXT=<text message>\" \"-TITLE=<window title>\"";
}
```

The `getIcon()` method should then return paths to two appropriately sized images. It should look something like this:

```
public String getIcon(int pIconType)
{
        switch (pIconType)
        {
               case IOpenTool.SMALL_ICON:
               return "/com/myCompany/OpenTools/images/SimpleMessageBox_16.gif";
             case IOpenTool.BIG_ICON:
             return "/com/myCompany/OpenTools/images/SimpleMessageBox_32.gif";
             default:
             return "";
     }
}
```

Execution

Finally, the `execute()` method, which carries out the functionality provided by the Open Tool. In this case, a message box is shown. If you are extending the `OpenToolAbstract` class, use the `getParameterValue()` method to easily retrieve the values of parameters, as they are set at runtime.

> **Note:**
>
> You must catch all exceptions and only raise an `OpenToolExecutionException`.

```
public void execute() throws OpenToolExecutionException
{
     try
     {
     if (getParameterValue("-TITLE") == null || getParameterValue("-
TITLE").equals("")) /* title was not filled in by user */
     {
               JOptionPane.showMessageDialog(null, (String) getParameterValue("-
```

```
TEXT"), (String) "Message", JOptionPane.INFORMATION_MESSAGE);
          } else
          {
               JOptionPane.showMessageDialog(null, (String) getParameterValue("-
TEXT"),
                         (String) getParameterValue("-TITLE"),
                         JOptionPane.INFORMATION_MESSAGE);
          }
     }
     /* Traps any exception and throw them as OpenToolExecutionException */
     catch (IllegalArgumentException e)
     {
          throw new OpenToolExecutionException(e);
     }
}
```

Open Tools at Runtime

In general, your Open Tool class is instantiated only very briefly, and is used in the following ways.

Installation

When the user chooses to install an Open Tool, Oracle Data Integrator instantiates the class and calls the methods `getDescription()`, `getProvider()`, `getIcon()`, and `getVersion()` to retrieve information about the class.

Use in a Package

When the Open Tool is used in a package, the class is instantiated briefly to call the methods `getDescription()`, `getProvider()`, `getIcon()`, and `getVersion()`. Additionally, `getSyntax()` is called to retrieve the code name of the Open Tool and its default arguments. The method `getParameters()` is called to display the list of arguments to the user.

Execution

Each time the Open Tool is executed in a package or procedure, the class is instantiated again; it has no persistence after its execution. The `execute()` method is called just once.

> **Tip:**
>
> See also Using Open Tools and Open Tools SDK documentation (JavaDoc).

2

Oracle Data Integrator Tools

This chapter lists all the Oracle Data Integrator Tools by category and describes its commands and parameters.

Oracle Data Integrator Tools by Category

This section lists Oracle Data Integrator tools by category.

Metadata

- OdiReverseGetMetaData
- OdiReverseManageShortcut
- OdiReverseResetTable
- OdiReverseSetMetaData

Oracle Data Integrator Objects

- OdiApplyDeploymentArchive
- OdiCreateDeploymentArchive
- OdiDeleteScen
- OdiExportAllScen
- OdiExportEnvironmentInformation
- OdiExportLog
- OdiExportMaster
- OdiExportObject
- OdiExportScen
- OdiExportWork
- OdiGenerateAllScen
- OdiImportObject
- OdiImportScen
- OdiRollbackDeploymentArchive
- OdiLockUnlockVCSRepository

Utilities

- OdiAnt
- OdiBeep

- OdiEnterpriseDataQuality
- OdiKillAgent
- OdiObjectStorageDelete
- OdiObjectStorageDownload
- OdiObjectStorageUpload
- OdiOSCommand
- OdiPingAgent
- OdiPurgeLog
- OdiReinitializeSeq
- OdiRemoveTemporaryObjects
- OdiRetrieveHadoopLog
- OdiStorageCSDownload
- OdiStorageCSUpload
- OdiStartLoadPlan
- OdiStartOwbJob
- OdiStartScen
- OdiUpdateAgentSchedule

Internet Related Tasks

- OdiFtp
- OdiFtpGet
- OdiFtpPut
- OdiInvokeRESTfulService
- OdiInvokeWebService
- OdiReadMail
- OdiScpGet
- OdiScpPut
- OdiSftp
- OdiSftpGet
- OdiSftpPut
- OdiSendMail

Files

- OdiFileAppend
- OdiFileCopy
- OdiFileDelete
- OdiFileMove

- OdiFileWait
- OdiMkDir
- OdiOutFile
- OdiSqlUnload
- OdiUnZip
- OdiZip

SAP

- OdiSAPALEClient and OdiSAPALEClient3
- OdiSAPALEServer and OdiSAPALEServer3

XML

- OdiXMLConcat
- OdiXMLSplit

Event Detection

- OdiFileWait
- OdiReadMail
- OdiSleep
- OdiWaitForChildSession
- OdiWaitForData
- OdiWaitForLoadPlans
- OdiWaitForLogData
- OdiWaitForTable

Changed Data Capture

- OdiManageOggProcess
- OdiOggCommand
- OdiRefreshJournalCount
- OdiRetrieveJournalData
- OdiWaitForData
- OdiWaitForLogData
- OdiWaitForTable

Alphabetical List of Oracle Data Integrator Tools

This section lists Oracle Data Integrator tools in alphabetical order.

- OdiAnt

- OdiApplyDeploymentArchive
- OdiBeep
- OdiCreateDeploymentArchive
- OdiDeleteScen
- OdiEnterpriseDataQuality
- OdiExportAllScen
- OdiExportEnvironmentInformation
- OdiExportLog
- OdiExportMaster
- OdiExportObject
- OdiExportScen
- OdiExportWork
- OdiFileAppend
- OdiFileDelete
- OdiFileCopy
- OdiFileMove
- OdiFileWait
- OdiFtp
- OdiFtpGet
- OdiFtpPut
- OdiGenerateAllScen
- OdiImportObject
- OdiImportScen
- OdiInvokeRESTfulService
- OdiInvokeWebService
- OdiKillAgent
- OdiManageOggProcess
- OdiMkDir
- OdiObjectStorageDownload
- OdiObjectStorageUpload
- OdiObjectStorageDelete
- OdiOggCommand
- OdiOSCommand
- OdiOutFile
- OdiPingAgent
- OdiPurgeLog
- OdiReadMail

- OdiRefreshJournalCount
- OdiReinitializeSeq
- OdiRemoveTemporaryObjects
- OdiRetrieveHadoopLog
- OdiRetrieveJournalData
- OdiReverseGetMetaData
- OdiReverseManageShortcut
- OdiReverseResetTable
- OdiReverseSetMetaData
- OdiRollbackDeploymentArchive
- OdiSAPALEClient and OdiSAPALEClient3
- OdiSAPALEServer and OdiSAPALEServer3
- OdiScpGet
- OdiScpPut
- OdiSendMail
- OdiSftp
- OdiSftpGet
- OdiSftpPut
- OdiSleep
- OdiSqlUnload
- OdiStartLoadPlan
- OdiStartOwbJob
- OdiStartScen
- OdiStorageCSDownload
- OdiStorageCSUpload
- OdiLockUnlockVCSRepository
- OdiUnZip
- OdiUpdateAgentSchedule
- OdiWaitForChildSession
- OdiWaitForData
- OdiWaitForLoadPlans
- OdiWaitForLogData
- OdiXMLConcat
- OdiXMLSplit
- OdiZip

OdiAnt

Use this command to execute an Ant buildfile.For more details and examples of Ant buildfiles, refer to the online documentation: http://jakarta.apache.org/ant/manual/index.html

Usage

```
OdiAnt -BUILDFILE=<file> -LOGFILE=<file> [-TARGET=<target>]
[-D<property name>=<property value>]* [-PROJECTHELP] [-HELP]
[-VERSION] [-QUIET] [-VERBOSE] [-DEBUG] [-EMACS]
[-LOGGER=<classname>] [-LISTENER=<classname>] [-FIND=<file>]
```

Parameters

Parameters	Mandatory	Description
-BUILDFILE=<file>	Yes	Ant buildfile. XML file containing the Ant commands.
-LOGFILE=<file>	Yes	Use given file for logging.
-TARGET=<target>	No	Target of the build process.
-D<property name>=<property value>	No	Used to pass the properties with its value to Ant buildfile
-PROJECTHELP	No	Displays the help on the project.
-HELP	No	Displays Ant help.
-VERSION	No	Displays Ant version.
-QUIET	No	Run in nonverbose mode.
-VERBOSE	No	Run in verbose mode.
-DEBUG	No	Prints debug information.
-EMACS	No	Displays the logging information without adornments.
-LOGGER=<classname>	No	Java class performing the logging.
-LISTENER=<classname>	No	Adds a class instance as a listener.
-FIND=<file>	No	Looks for the Ant buildfile from the root of the file system and uses it.

Examples

Download the `*.html` files from the directory `/download/public` using FTP from `ftp.example.com` to the directory `C:\temp`.

Step 1: Generate the Ant buildfile.

```
OdiOutFile -FILE=c:\temp\ant_cmd.xml
<?xml version="1.0"?>
<project name="myproject" default="ftp" basedir="/">
     <target name="ftp">
          <ftp action="get" remotedir="/download/public"
          server="ftp.example.com" userid="anonymous"
          password="me@example.com">
                <fileset dir="c:\temp">
```

```
            <include name="**/*.html"/>
                 </fileset>
     </ftp>
   </target>
</project>
```

Step 2: Run the Ant buildfile.

```
OdiAnt -BUILDFILE=c:\temp\ant_cmd.xml -LOGFILE=c:\temp\ant_cmd.log
```

OdiApplyDeploymentArchive

Use this command to apply an Initial/Patch Deployment Archive (DA) onto an ODI repository.

Usage

```
OdiApplyDeploymentArchive -ARCHIVE_FILE_NAME=<archive_file_name>
[-APPLY_WITHOUT_CIPHER_DATA=<yes|no>] [-EXPORT_KEY=<Export_Key>]
[-CREATE_ROLLBACK_ARCHIVE=<yes|no>]
[-ROLLBACK_FILE_NAME=<rollback_file_name>]
[-INCLUDE_PHYSICAL_TOPOLOGY=<yes|no>]
```

Parameters

Parameter	Mandatory	Description
`ARCHIVE_FILE_NAME=<archive_file_name>`	Yes	Full path/Complete name of the deployment archive zip file.
`APPLY_WITHOUT_CIPHER_DATA=<yes|no>`	No[1]	If set to yes, any cipher data present in the deployment archive will be made null. If set to no, the export key will be used to migrate the cipher data. The default value is No.

Parameter	Mandatory	Description
`EXPORT_KEY=<Export_Key>`	No	Specifies a cryptographic private key used to migrate cipher data in the deployment archive objects.

Note:

The `EXPORT_KEY` parameter should be an encrypted string. For information on the encoding process, see Encoding a Password in *Administering Oracle Data Integrator*.

Parameter	Mandatory	Description
CREATE_ROLLBACK_ARCHIVE=<yes|no>	No[2]	Specifies if a rollback deployment archive must be created. If set to Yes, a rollback deployment archive will be created before applying the patch. If set to No, the rollback deployment archive will not be created. **Note:** This option is applicable only to the patch deployment archive.
ROLLBACK_FILE_NAME=<rollback_file_name>	No	Complete file name of the rollback deployment archive.
INCLUDE_PHYSICAL_TOPOLOGY=<yes|no>	No	Specifies if the Physical Topology Objects in the deployment archive should be applied onto the target repository. The default value is Yes.

[1] If the APPLY_WITHOUT_CIPHER_DATA parameter is set to No, the EXPORT_KEY parameter must be specified.
[2] If the CREATE_ROLLBACK_ARCHIVE parameter is set to Yes, the ROLLBACK_FILE_NAME parameter must be specified.

Examples

Patch a repository using a patch deployment archive using export key and create a rollback deployment archive.

```
OdiApplyDeploymentArchive -ARCHIVE_FILE_NAME=archive_file_name
-APPLY_WITHOUT_CIPHER_DATA=no -EXPORT_KEY=Export_Key
-CREATE_ROLLBACK_ARCHIVE=yes
-ROLLBACK_FILE_NAME=rollback_file_name -INCLUDE_PHYSICAL_TOPOLOGY=yes
```

OdiBeep

Use this command to play a default beep or sound file on the machine hosting the agent.

The following file formats are supported by default:

- WAV
- AIF
- AU

> **Note:**
>
> To play other file formats, you must add the appropriate JavaSound Service Provider Interface (JavaSound SPI) to the application classpath.

Usage

```
OdiBeep [-FILE=<sound_file>]
```

Parameters

Parameters	Mandatory	Description
-FILE	No	Path and file name of sound file to be played. If not specified, the default beep sound for the machine is used.

Examples

```
OdiBeep -FILE=c:\wav\alert.wav
```

OdiCreateDeploymentArchive

Use this command to create a Deployment Archive (DA) from the ODI repository or VCS label/tag.

Usage

In SVN

```
OdiCreateDeploymentArchive -ARCHIVE_NAME=<archive_name>
-ARCHIVE_FILE_NAME=<archive_file_name>
[-SOURCE_TYPE=VCS|ODI]
[-ARCHIVE_TYPE=INITIAL|PATCH]
[-CREATE_WITHOUT_CIPHER_DATA=<yes|no>]
[-EXPORT_KEY=<Export_Key>]
[-VCS_LABEL=<vcs_label>]
[-VCS_TYPE=<vcs_type>]
[-VCS_AUTH_TYPE=<vcs_auth_type>]
[-VCS_URL=<vcs_url>]
[-VCS_USER=<vcs_user>]
[-VCS_PASS=<vcs_pass>]
[-VCS_PROXY_HOST=<vcs_proxy_host>]
[-VCS_PROXY_PORT=<vcs_proxy_port>]
[-VCS_PROXY_USER=<vcs_proxy_user>]
[-VCS_PROXY_PASS=<vcs_proxy_pass>]
[-INCLUDE_PHYSICAL_TOPOLOGY=<yes|no>]
```

In Git

```
OdiCreateDeploymentArchive -ARCHIVE_NAME=<archive_name>
-ARCHIVE_FILE_NAME=<archive_file_name>
[-SOURCE_TYPE=VCS|ODI]
[-ARCHIVE_TYPE=INITIAL|PATCH]
[-CREATE_WITHOUT_CIPHER_DATA=<yes|no>]
[-EXPORT_KEY=<Export_Key>]
[-DESCRIPTION=<Description>]
```

```
[-VCS_TAG=<vcs_tag>]
[-VCS_TYPE=<vcs_type>]
[-VCS_AUTH_TYPE=<vcs_auth_type>]
[-VCS_URL=<vcs_url>]
[-VCS_USER=<vcs_user>]
[-VCS_PASS=<vcs_pass>]
[-VCS_PROXY_HOST=<vcs_proxy_host>]
[-VCS_PROXY_PORT=<vcs_proxy_port>]
[-VCS_PROXY_USER=<vcs_proxy_user>]
[-VCS_PROXY_PASS=<vcs_proxy_pass>]
[-VCS_SSH_PRIVATE_KEY_PATH=<vcs_ssh_private_key_path>]
[-VCS_SSH_PASS_PHRASE=<vcs_ssh_pass_phrase>]
[-VCS_SSH_PORT=<vcs_ssh_port>]
[-VCS_SSL_CERT_PATH=<vcs_ssl_cert_path>]
[-VCS_SSL_PASS_PHRASE=<vcs_ssl_pass_phrase>]
[-INCLUDE_PHYSICAL_TOPOLOGY=<yes|no>]
```

Parameters

Parameter	Mandatory	Description
ARCHIVE_NAME=<archive_name>	Yes	Name of deployment archive.
ARCHIVE_FILE_NAME=<archive_file_name>	Yes	Full path/Complete name of the deployment archive zip file.
SOURCE_TYPE=VCS|ODI	No[1]	Source from which the deployment archive needs to be created. The source can be: • VCS in case of initial/patch deployment archive creation from VCS label/tag. • ODI in case of initial deployment archive creation from entire ODI repository.
VCS_TAG=<vcs_tag>	No	VCS tag name.
ARCHIVE_TYPE=INITIAL|PATCH	No	Type of deployment archive. Can be INITIAL or PATCH. The default value is PATCH.
CREATE_WITHOUT_CIPHER_DATA=<yes|no>	No[2]	If set to Yes, any cipher data present in the deployment archive will be made null. If set to No, the export key will be used to migrate the cipher data. The default value is No.
EXPORT_KEY=<Export_Key>	No	Specifies a cryptographic private key used to encrypt cipher data in the deployment archive objects.
VCS_LABEL=<vcs_label>	No	VCS label name.
DESCRIPTION=<Description>	No	Description for this deployment archive.
VCS_TYPE=<vcs_type>	No	Type of VCS. Can be SVN or Git.

Parameter	Mandatory	Description
VCS_AUTH_TYPE=<vcs_auth_type>	No[3]	Authentication type of the VCS used. The value is: • GITBASIC for Git authentication. • SVNBASIC for SVN authentication. • HTTPBASIC for HTTP Basic authentication. • HTTPPROXY for HTTP Proxy authentication. • GIT_SSH for SSH authentication. • GIT_SSL for HTTPS authentication. • FILE for File based authentication.
VCS_URL=<vcs_url>	No	VCS URL.
VCS_USER=<vcs_user>	No	VCS Username.
VCS_PASS=<vcs_pass>	No	VCS Password.
VCS_PROXY_HOST=<vcs_proxy_host>	No	VCS Proxy Host.
VCS_PROXY_PORT=<vcs_proxy_port>	No	VCS Proxy Port.
VCS_PROXY_USER=<vcs_proxy_user>	No	VCS Proxy User.
VCS_PROXY_PASS=<vcs_proxy_pass>	No	VCS Proxy Password.
VCS_SSH_PRIVATE_KEY_PATH=<vcs_ssh_private_key_path>	No	VCS SSH private key file path, in case of private key authentication.
VCS_SSH_PASS_PHRASE=<vcs_ssh_pass_phrase>	No	VCS SSH pass phrase, if provided during private key generation.
VCS_SSH_PORT=<vcs_ssh_port>	No	VCS SSH port.
VCS_SSL_CERT_PATH=<vcs_ssl_cert_path>	No	VCS HTTP SSL certificate path.
VCS_SSL_PASS_PHRASE=<vcs_ssl_pass_phrase>	No	VCS SSL pass phrase.
INCLUDE_PHYSICAL_TOPOLOGY=<yes|no>	No	Specifies if the Physical Topology Objects in the repository should be included in the deployment archive. The default value is Yes.

[1] If the SOURCE_TYPE parameter is specified as VCS, the VCS_TAG/VCS_LABEL parameter must be specified.

[2] If the CREATE_WITHOUT_CIPHER_DATA parameter is set to No, the EXPORT_KEY parameter must be specified.

[3] If the VCS_AUTH_TYPE parameter is specified as GITBASIC or SVNBASIC, the VCS_URL, VCS_USER, and VCS_PASS parameters must be specified.

If the VCS_AUTH_TYPE parameter is specified as SVNBASIC, the VCS_SSH_PORT parameter must be specified.

If the VCS_AUTH_TYPE parameter is specified as HTTPPROXY, the VCS_PROXY_HOST, VCS_PROXY_PORT, VCS_PROXY_USER, and VCS_PROXY_PASS parameters must be specified.

If the VCS_AUTH_TYPE parameter is specified as GIT_SSH, the VCS_SSH_PRIVATE_KEY_PATH and VCS_SSH_PASS_PHRASE parameters must be specified.

If the VCS_AUTH_TYPE parameter is specified as GIT_SSL, the VCS_SSL_CERT_PATH and VCS_SSL_PASS_PHRASE parameters must be specified.

Examples

Create a patch deployment archive from SVN label with cipher.

```
OdiCreateDeploymentArchive -ARCHIVE_NAME=archive_name
-ARCHIVE_FILE_NAME=archive_file_name
-SOURCE_TYPE=VCS
-ARCHIVE_TYPE=PATCH
-CREATE_WITHOUT_CIPHER_DATA=no
-EXPORT_KEY=Export_Key
-VCS_LABEL=vcs_label
-VCS_TYPE=SVN
-VCS_AUTH_TYPE=BASIC
-VCS_URL=vcs_url
-VCS_USER=vcs_user
-VCS_PASS=vcs_pass
-INCLUDE_PHYSICAL_TOPOLOGY=yes
```

Create an initial deployment archive from a Git tag.

```
OdiCreateDeploymentArchive -ARCHIVE_NAME=<archive_name>
-ARCHIVE_TYPE=INITIAL
-SOURCE_TYPE=VCS
-VCS_TAG=<vcs_tag>
-ARCHIVE_FILE_NAME=<archive_file_name>
-CREATE_WITHOUT_CIPHER_DATA=<yes|no>
-EXPORT_KEY=<Export_Key>
-INCLUDE_PHYSICAL_TOPOLOGY=<yes|no>
-VCS_URL=<vcs_url>
-VCS_USER=<vcs_user>
-VCS_PASS=<vcs_pass>
```

OdiDeleteScen

Use this command to delete a given scenario version.

Usage

```
OdiDeleteScen -SCEN_NAME=<name> -SCEN_VERSION=<version>
```

Parameters

Parameters	Mandatory	Description
-SCEN_NAME=<name>	Yes	Name of the scenario to delete.
-SCEN_VERSION=<version>	Yes	Version of the scenario to delete.

Examples

Delete the DWH scenario in version 001.

```
OdiDeleteScen -SCEN_NAME=DWH -SCEN_VERSION=001
```

OdiEnterpriseDataQuality

Use this command to invoke an Oracle Enterprise Data Quality (Datanomic) job.

> **Note:**
>
> The OdiEnterpriseDataQuality tool supports Oracle Enterprise Data Quality version 8.1.6 and later.

Usage

```
OdiEnterpriseDataQuality "-JOB_NAME=<EDQ job name>"
"-PROJECT_NAME=<EDQ project name>" "-CONTEXT=<context>"
"-LSCHEMA=<logical_schema>" "-SYNCHRONOUS=<yes|no>"
```

Parameters

Parameters	Mandatory	Description
`-JOB_NAME=<EDQ job name>`	Yes	Name of the Enterprise Data Quality job.
`-PROJECT_NAME=<EDQ project name>`	Yes	Name of the Enterprise Data Quality project.
`-SYNCHRONOUS=<yes|no>`	No	If set to Yes (default), the tool waits for the quality process to complete before returning, with possible error code. If set to No, the tool ends immediately with success and does not wait for the quality process to complete.

Examples

Execute the Enterprise Data Quality job `CLEANSE_CUSTOMERS` located in the project `CUSTOMERS`.

```
OdiEnterpriseDataQuality "-JOB_NAME=CLEANSE_CUSTOMERS" "-PROJECT_NAME=CUSTOMERS"
"-CONTEXT=Development" "-LSCHEMA=EDQ Logical Schema" "-SYNCHRONOUS=yes"
```

OdiExportAllScen

Use this command to export a group of scenarios from the connected repository.

The export files are named `SCEN_<scenario name><scenario version>.xml`. This command reproduces the behavior of the export feature available in Designer Navigator and Operator Navigator.

Usage

```
OdiExportAllScen -TODIR=<directory> [-FORCE_OVERWRITE=<yes|no>]
[-FROM_PROJECT=<project_id>] [-FROM_FOLDER=<folder_id>]
[-FROM_PACKAGE=<package_id>] [-RECURSIVE_EXPORT=<yes|no>]
[-XML_VERSION=<1.0>] [-XML_CHARSET=<charset>]
```

```
[-JAVA_CHARSET=<charset>] [-EXPORT_KEY=<key>] [-EXPORT_MAPPING=<yes|no>]
[-EXPORT_PACK=<yes|no>] [-EXPORT_POP=<yes|no>]
[-EXPORT_TRT=<yes|no>] [-EXPORT_VAR=<yes|no>]
[EXPORT_WITHOUT_CIPHER_DATA=<yes|no>]
```

Parameters

Parameters	Mandatory	Description
-TODIR=<directory>	Yes	Directory into which the export files are created.
-FORCE_OVERWRITE=<yes|no>	No	If set to Yes, existing export files are overwritten without warning. The default value is No.
-FROM_PROJECT=<project_id>	No	ID of the project containing the scenarios to export. This value is the **Global ID** that displays in the **Version** tab of the project window in Studio. If this parameter is not set, scenarios from all projects are taken into account for the export.
-FROM_FOLDER=<folder_id>	No	ID of the folder containing the scenarios to export. This value is the **Global ID** that displays in the **Version** tab of the folder window in Studio. If this parameter is not set, scenarios from all folders are taken into account for the export.
-FROM_PACKAGE=<package_id>	No	ID of the source package of the scenarios to export. This value is the **Global ID** that displays in the **Version** tab of the package window in Studio. If this parameter is not set, scenarios from all components are taken into account for the export.
-RECURSIVE_EXPORT=<yes|no>	No	If set to Yes (default), all child objects (schedules) are exported with the scenarios.
-XML_VERSION=<1.0>	No	Sets the XML version shown in the XML header. The default value is 1.0.
-XML_CHARSET=<charset>	No	Encoding specified in the XML export file in the tag <?xml version="1.0" encoding="ISO-8859-1"?>. The default value is ISO-8859-1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html

Parameters	Mandatory	Description
-JAVA_CHARSET=<charset>	No	Target file encoding. The default value is ISO-8859-1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html
-EXPORT_KEY=<key>	No[1]	Specifies a cryptographic private key used to encrypt sensitive cipher data. You must specify this key again when importing the exported object in order to import the cipher data.
-EXPORT_MAPPING=<yes|no>	No	Indicates if the mapping scenarios should be exported. The default value is No.
-EXPORT_PACK=<yes|no>	No	Indicates if the scenarios attached to packages should be exported. The default value is Yes.
-EXPORT_POP=<yes|no>	No	Indicates if the scenarios attached to mappings should be exported. The default value is No.
-EXPORT_TRT=<yes|no>	No	Indicates if the scenarios attached to procedures should be exported. The default value is No.
-EXPORT_VAR=<yes|no>	No	Indicates if the scenarios attached to variables should be exported. The default value is No.
-EXPORT_WITHOUT_CIPHER_DATA=<yes|no>	No[2]	When set to Yes, specifies that sensitive (cipher) values should be set to null in the object when it is exported. When set to No or when this parameter is omitted, you must include the -EXPORT_KEY parameter and specify a valid key. The default value is No.

1 If the -EXPORT_KEY parameter is not specified, the -EXPORT_WITHOUT_CIPHER_DATA parameter must be specified, and must be set to Yes.

2 If -EXPORT_WITHOUT_CIPHER_DATA is not specified, or if it is specified and set to No, you must specify the -EXPORT_KEY parameter with a valid key value.

Examples

Export all scenarios from the DW01 project of **Global ID** 2edb524d-eb17-42ea-8aff-399ea9b13bf3 into the /temp/ directory, with all dependent objects, using the key examplekey1 to encrypt sensitive data.

```
OdiExportAllScen -FROM_PROJECT=2edb524d-eb17-42ea-8aff-399ea9b13bf3 -TODIR=/
temp/ -RECURSIVE_EXPORT=yes -EXPORT_KEY=examplekey1
```

OdiExportEnvironmentInformation

Use this command to export the details of the technical environment into a comma separated (`.csv`) file into the directory of your choice. This information is required for maintenance or support purposes.

Usage

```
OdiExportEnvironmentInformation -TODIR=<toDir> -FILE_NAME=<FileName>
[-CHARSET=<charset>] [-SNP_INFO_REC_CODE=<row_code>]
[-MASTER_REC_CODE=<row_code>] [-WORK_REC_CODE=<row_code>]
[-AGENT_REC_CODE=<row_code>] [-TECHNO_REC_CODE=<row_code>]
[-RECORD_SEPARATOR_HEXA=<rec_sep>] [-FIELD_SEPARATOR_HEXA=<field_sep]
[-TEXT_SEPARATOR=<text_sep>]
```

Parameter

Parameters	Mandatory	Description
`-TODIR=<toDir>`	Yes	Target directory for the export.
`-FILE_NAME=<FileName>`	Yes	Name of the CSV export file. The default value is `snps_tech_inf.csv`.
`-CHARSET=<charset>`	No	Character set of the export file.
`-SNP_INFO_REC_CODE=<row_code>`	No	Code used to identify rows that describe the current version of Oracle Data Integrator and the current user. This code is used in the first field of the record. The default value is `SUNOPSIS`.
`-MASTER_REC_CODE=<row_code>`	No	Code for rows containing information about the master repository. The default value is `MASTER`.
`-WORK_REC_CODE=<row_code>`	No	Code for rows containing information about the work repository. The default value is `WORK`.
`-AGENT_REC_CODE=<row_code>`	No	Code for rows containing information about the various agents that are running. The default value is `AGENT`.
`-TECHNO_REC_CODE=<row_code>`	No	Code for rows containing information about the data servers, their versions, and so on. The default value is `TECHNO`.
`-RECORD_SEPARATOR_HEXA=<rec_sep>`	No	One or several characters in hexadecimal code separating lines (or records) in the file. The default value is `0D0A`.
`-FIELD_SEPARATOR_HEXA=<field_sep>`	No	One or several characters in hexadecimal code separating the fields in a record. The default value is `2C`.

Parameters	Mandatory	Description
-TEXT_SEPARATOR=<text_sep>	No	Character in hexadecimal code delimiting a STRING field. The default value is 22.

Examples

Export the details of the technical environment into the /temp/snps_tech_inf.csv export file.

```
OdiExportEnvironmentInformation "-TODIR=/temp/"
"-FILE_NAME=snps_tech_inf.csv" "-CHARSET=ISO8859_1"
"-SNP_INFO_REC_CODE=SUNOPSIS" "-MASTER_REC_CODE=MASTER"
"-WORK_REC_CODE=WORK" "-AGENT_REC_CODE=AGENT"
"-TECHNO_REC_CODE=TECHNO" "-RECORD_SEPARATOR_HEXA=0D0A"
"-FIELD_SEPARATOR_HEXA=2C" "-TEXT_SEPARATOR_HEXA=22"
```

OdiExportLog

Use this command to export the execution log into a ZIP export file.

Usage

```
OdiExportLog -TODIR=<toDir> [-EXPORT_TYPE=<logsToExport>] [-EXPORT_KEY=<key>]
[-ZIPFILE_NAME=<zipFileName>] [-XML_CHARSET=<charset>]
[-JAVA_CHARSET=<charset>] [-FROMDATE=<from_date>] [-TODATE=<to_date>]
[-AGENT=<agent>] [-CONTEXT=<context>] [-STATUS=<status>] [-USER_FILTER=<user>]
[-NAME=<sessionOrLoadPlanName>] [EXPORT_WITHOUT_CIPHER_DATA=<yes|no>]
```

Parameters

Parameters	Mandatory	Description
-EXPORT_TYPE=<logsToExport>	No	Export the log of: • LOAD_PLAN_RUN: All Load Plan runs that match the export criteria are exported, including all sessions launched by the Load Plan runs along the child session's hierarchy. • SESSION: All session logs that match the export filter criteria are exported. All Load Plan sessions will be excluded when exporting the session logs. • ALL: All Load Plan runs and session logs that match the filter criteria are exported.
-EXPORT_KEY=<key>	No[1]	Specifies a cryptographic private key used to encrypt sensitive cipher data. You must specify this key again when importing the exported object in order to import the cipher data.
-TODIR=<toDir>	Yes	Target directory for the export.

Parameters	Mandatory	Description
-ZIPFILE_NAME=<zipFileName>	No	Name of the compressed file.
-XML_CHARSET=<charset>	No	XML version specified in the export file. Parameter xml version in the XML file header. <?xml version="1.0" encoding="ISO-8859-1"?>. The default value is ISO-8859-1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html
-JAVA_CHARSET=<charset>	No	Result file Java character encoding. The default value is ISO8859_1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html
-FROMDATE=<from_date>	No	Beginning date for the export, using the format yyyy/MM/dd hh:mm:ss. All sessions from this date are exported.
-TODATE=<to_date>	No	End date for the export, using the format yyyy/MM/dd hh:mm:ss. All sessions to this date are exported.
-AGENT=<agent>	No	Exports only sessions executed by the agent <agent>.
-CONTEXT=<context>	No	Exports only sessions executed in the context code <context>.
-STATUS=<status>	No	Exports only sessions in the specified state. Possible states are Done, Error, Queued, Running, Waiting, and Warning.
-USER_FILTER=<user>	No	Exports only sessions launched by <user>.
-NAME=<sessionOrLoadPlanName>	No	Name of the session or Load Plan to be exported.
-EXPORT_WITHOUT_CIPHER_DATA=<yes|no>	No[2]	When set to Yes, specifies that sensitive (cipher) values should be set to null in the object when it is exported. When set to No or when this parameter is omitted, you must include the -EXPORT_KEY parameter and specify a valid key. The default value is No.

[1] If the -EXPORT_KEY parameter is not specified, the -EXPORT_WITHOUT_CIPHER_DATA parameter must be specified, and must be set to Yes.
[2] If -EXPORT_WITHOUT_CIPHER_DATA is not specified, or if it is specified and set to No, you must specify the -EXPORT_KEY parameter with a valid key value.

Examples

Export and compress the log into the /temp/log2.zip export file.

```
OdiExportLog "-EXPORT_TYPE=ALL" "-EXPORT_KEY=examplekey1"
"-TODIR=/temp/" "-ZIPFILE_NAME=log2.zip" "-XML_CHARSET=ISO-8859-1"
"-JAVA_CHARSET=ISO8859_1"
```

OdiExportMaster

Use this command to export the master repository to a directory or ZIP file. The versions and/or solutions stored in the master repository are optionally exported.

Usage

```
OdiExportMaster -TODIR=<toDir> [-ZIPFILE_NAME=<zipFileName>]
[-EXPORT_KEY=<key>] [-EXPORT_SOLUTIONS=<yes|no>] [-EXPORT_VERSIONS=<yes|no>]
[-XML_CHARSET=<charset>] [-JAVA_CHARSET=<charset>]
[EXPORT_WITHOUT_CIPHER_DATA=<yes|no>]
```

Parameters

Parameters	Mandatory	Description
`-TODIR=<toDir>`	Yes	Target directory for the export.
`-ZIPFILE_NAME=<zipFileName>`	No	Name of the compressed file.
`-EXPORT_KEY=<key>`	No[1]	Specifies a cryptographic private key used to encrypt sensitive cipher data. You must specify this key again when importing the exported object in order to import the cipher data.
`-EXPORT_SOLUTIONS=<yes|no>`	No	Exports all solutions that are stored in the repository. The default value is No.
`-EXPORT_VERSIONS=<yes|no>`	No	Exports all versions of objects that are stored in the repository. The default value is No.
`-XML_CHARSET=<charset>`	No	XML version specified in the export file. Parameter `xml version` in the XML file header. `<?xml version="1.0" encoding="ISO-8859-1"?>`. The default value is `ISO-8859-1`. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html

Parameters	Mandatory	Description
-JAVA_CHARSET=<charset>	No	Result file Java character encoding. The default value is ISO8859_1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html
-EXPORT_WITHOUT_CIPHER_DATA=<yes|no>	No[2]	When set to Yes, specifies that sensitive (cipher) values should be set to null in the object when it is exported. When set to No or when this parameter is omitted, you must include the -EXPORT_KEY parameter and specify a valid key. The default value is No.

[1] If the -EXPORT_KEY parameter is not specified, the -EXPORT_WITHOUT_CIPHER_DATA parameter must be specified, and must be set to Yes.
[2] If -EXPORT_WITHOUT_CIPHER_DATA is not specified, or if it is specified and set to No, you must specify the -EXPORT_KEY parameter with a valid key value.

Examples

Export and compress the master repository into the export.zip file located in the /temp/ directory.

```
OdiExportMaster "-TODIR=/temp/" "-ZIPFILE_NAME=export.zip"
"-EXPORT_KEY=examplekey1" "-XML_CHARSET=ISO-8859-1"
"-JAVA_CHARSET=ISO8859_1" "-EXPORT_VERSIONS=YES"
```

OdiExportObject

Use this command to export an object from the current repository. This command reproduces the behavior of the export feature available in the user interface.

Usage

```
OdiExportObject -CLASS_NAME=<class_name> -I_OBJECT=<object_id>
[-EXPORT_KEY=<key>] [-EXPORT_DIR=<directory>]
[-EXPORT_NAME=<export_name>|-FILE_NAME=<file_name>] [-FORCE_OVERWRITE=<yes|no>] [-
RECURSIVE_EXPORT=<yes|no>] [-XML_VERSION=<1.0>] [-XML_CHARSET=<charset>]
[-JAVA_CHARSET=<charset>] [EXPORT_WITHOUT_CIPHER_DATA=<yes|no>]
```

Parameters

Parameters	Mandatory	Description
-CLASS_NAME=<class_name>	Yes	Class of the object to export (see the following list of classes).

Parameters	Mandatory	Description
-I_OBJECT=<object_id>	Yes	Object identifier. This value is the **Global ID** that displays in the **Version** tab of the object edit window.
-EXPORT_KEY=<key>	No[1]	Specifies a cryptographic private key used to encrypt sensitive cipher data. You must specify this key again when importing the exported object in order to import the cipher data.
-FILE_NAME=<file_name>	No	Export file name. Absolute path or relative path from EXPORT_DIR. This file name may or may not comply with the Oracle Data Integrator standard export file prefix and suffix. To comply with these standards, use the -EXPORT_NAME parameter instead. This parameter cannot be used if -EXPORT_NAME is set.
-EXPORT_DIR=<directory>	No	Directory where the object will be exported. The export file created in this directory is named based on the -FILE_NAME and -EXPORT_NAME parameters. If -FILE_NAME or -EXPORT_NAME are not specified, the export file is automatically named <object_prefix>_<object_name>.xml. For example, a project named Datawarehouse would be exported to PRJ_Datawarehouse.xml.
-EXPORT_NAME=<export_name>	No	Export name. Use this parameter to generate an export file named <object_prefix>_<export_name>.xml. This parameter cannot be used with -FILE_NAME.
-FORCE_OVERWRITE=<yes|no>	No	If set to Yes, an existing export file with the same name is forcibly overwritten. The default value is No.
-RECURSIVE_EXPORT=<yes|no>	No	If set to Yes (default), all child objects are exported with the current object. For example, if exporting a project, all folders, KMs, and so on in this project are exported into the project export file.
-XML_VERSION=<1.0>	No	Sets the XML version that appears in the XML header. The default value is 1.0.

Parameters	Mandatory	Description
-XML_CHARSET=<charset>	No	Encoding specified in the XML file, in the tag <?xml version="1.0" encoding="ISO-8859-1"?>. The default value is ISO-8859-1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html
-JAVA_CHARSET=<charset>	No	Target file encoding. The default value is ISO8859_1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html
-EXPORT_WITHOUT_CIPHER_DATA=<yes|no>	No[2]	When set to Yes, specifies that sensitive (cipher) values should be set to null in the object when it is exported. When set to No or when this parameter is omitted, you must include the -EXPORT_KEY parameter and specify a valid key. The default value is No.

[1] If the -EXPORT_KEY parameter is not specified, the -EXPORT_WITHOUT_CIPHER_DATA parameter must be specified, and must be set to Yes.
[2] If -EXPORT_WITHOUT_CIPHER_DATA is not specified, or if it is specified and set to No, you must specify the -EXPORT_KEY parameter with a valid key value.

List of Classes

Object	Class Name
Column	SnpCol
Condition/Filter	SnpCond
Context	SnpContext
Data Server	SnpConnect
Datastore	SnpTable
Folder	SnpFolder
Interface	SnpPop
Language	SnpLang
Loadplan	SnpLoadPlan
Mapping	SnpMapping
Model	SnpModel
Package	SnpPackage
Physical Schema	SnpPschema
Procedure or KM	SnpTrt
Procedure or KM Option	SnpUserExit
Project	SnpProject

Object	Class Name
Reference	SnpJoin
Reusable Mapping	SnpMapping
Scenario	SnpScen
Sequence	SnpSequence
Step	SnpStep
Sub-Model	SnpSubModel
Technology	SnpTechno
User Functions	SnpUfunc
Variable	SnpVar
Version of an Object	SnpVer

Examples

Export the `DW01` project of **Global ID** `2edb524d-eb17-42ea-8aff-399ea9b13bf3` into the `/temp/dw1.xml` export file, with all dependent objects.

```
OdiExportObject -CLASS_NAME=SnpProject
-I_OBJECT=2edb524d-eb17-42ea-8aff-399ea9b13bf3 -EXPORT_KEY=examplekey1
-FILE_NAME=/temp/dw1.xml -FORCE_OVERWRITE=yes
-RECURSIVE_EXPORT=yes
```

OdiExportScen

Use this command to export a scenario from the current work repository.

Usage

```
OdiExportScen -SCEN_NAME=<scenario_name> -SCEN_VERSION=<scenario_version>
[-EXPORT_KEY=<key>] [-EXPORT_DIR=<directory>]
[-FILE_NAME=<file_name>|EXPORT_NAME=<export_name>] [-FORCE_OVERWRITE=<yes|no>]
[-RECURSIVE_EXPORT=<yes|no>] [-XML_VERSION=<1.0>] [-XML_CHARSET=<encoding>]
[-JAVA_CHARSET=<encoding>] [EXPORT_WITHOUT_CIPHER_DATA=<yes|no>]
```

Parameters

Parameters	Mandatory	Description
`-SCEN_NAME=<scenario_name>`	Yes	Name of the scenario to be exported.
`-SCEN_VERSION=<scenario_version>`	Yes	Version of the scenario to be exported.

Parameters	Mandatory	Description
-EXPORT_KEY=<key>	No[1]	Specifies a cryptographic private key used to encrypt sensitive cipher data. You must specify this key again when importing the exported object in order to import the cipher data. **Note:** The EXPORT_KEY parameter should be an encrypted string. For information on the encoding process, see Encoding a Password in *Administering Oracle Data Integrator*.
-FILE_NAME=<file_name>	Yes	Export file name. Absolute path or relative path from -EXPORT_DIR. This file name may or not comply with the Oracle Data Integrator standard export file prefix and suffix for scenarios. To comply with these standards, use the -EXPORT_NAME parameter instead. This parameter cannot be used if -EXPORT_NAME is set.

Parameters	Mandatory	Description
-EXPORT_DIR=<directory>	No	Directory where the scenario will be exported. The export file created in this directory is named based on the -FILE_NAME and -EXPORT_NAME parameters. If -FILE_NAME or -EXPORT_NAME are not specified, the export file is automatically named SCEN_<scenario_name><scenario_version>.xml.
-EXPORT_NAME=<export_name>	No	Export name. Use this parameter to generate an export file named SCEN_<export_name>.xml. This parameter cannot be used with -FILE_NAME.
-FORCE_OVERWRITE=<yes|no>	No	If set to Yes, overwrites the export file if it already exists. The default value is No.
-RECURSIVE_EXPORT=<yes|no>	No	Forces the export of the objects under the scenario. The default value is Yes.
-XML_VERSION=<1.0>	No	Version specified in the generated XML file, in the tag <?xml version="1.0" encoding="ISO-8859-1"?>. The default value is 1.0.
-XML_CHARSET=<encoding>	No	Encoding specified in the XML file, in the tag <?xml version="1.0" encoding="ISO-8859-1"?>. The default value is ISO-8859-1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html
-JAVA_CHARSET=<encoding>	No	Target file encoding. The default value is ISO8859_1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html
-EXPORT_WITHOUT_CIPHER_DATA=<yes|no>	No[2]	When set to Yes, specifies that sensitive (cipher) values should be set to null in the object when it is exported. When set to No or when this parameter is omitted, you must include the -EXPORT_KEY parameter and specify a valid key. The default value is No.

[1] If the -EXPORT_KEY parameter is not specified, the -EXPORT_WITHOUT_CIPHER_DATA parameter must be specified, and must be set to Yes.

[2] If -EXPORT_WITHOUT_CIPHER_DATA is not specified, or if it is specified and set to No, you must specify the -EXPORT_KEY parameter with a valid key value.

Examples

Export the LOAD_DWH scenario in version 1 into the /temp/load_dwh.xml export file, with all dependent objects.

```
OdiExportScen -SCEN_NAME=LOAD_DWH -SCEN_VERSION=1 -EXPORT_KEY=examplekey1
-FILE_NAME=/temp/load_dwh.xml -RECURSIVE_EXPORT=yes
```

OdiExportWork

Use this command to export the work repository to a directory or ZIP export file.

Usage

```
OdiExportWork -TODIR=<directory> [-ZIPFILE_NAME=<zipFileName>]
[-EXPORT_KEY=<key>] [-XML_CHARSET=<charset>] [-JAVA_CHARSET=<charset>]
[EXPORT_WITHOUT_CIPHER_DATA=<yes|no>]
```

Parameters

Parameters	Mandatory	Description
-TODIR=<directory>	Yes	Target directory for the export.
-ZIPFILE_NAME=<zipFileName>	No	Name of the compressed file.
-EXPORT_KEY=<key>	No[1]	Specifies a cryptographic private key used to encrypt sensitive cipher data. You must specify this key again when importing the exported object in order to import the cipher data.
-XML_CHARSET=<charset>	No	XML version specified in the export file. Parameter xml version in the XML file header. <?xml version="1.0" encoding="ISO-8859-1"?>. The default value is ISO-8859-1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html
-JAVA_CHARSET=<charset>	No	Result file Java character encoding. The default value is ISO8859_1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html
-EXPORT_WITHOUT_CIPHER_DATA=<yes|no>	No[2]	When set to Yes, specifies that sensitive (cipher) values should be set to null in the object when it is exported. When set to No or when this parameter is omitted, you must include the -EXPORT_KEY parameter and specify a valid key. The default value is No.

[1] If the -EXPORT_KEY parameter is not specified, the -EXPORT_WITHOUT_CIPHER_DATA parameter must be specified, and must be set to Yes.
[2] If -EXPORT_WITHOUT_CIPHER_DATA is not specified, or if it is specified and set to No, you must specify the -EXPORT_KEY parameter with a valid key value.

Examples

Export and compress the work repository into the /temp/workexport.zip export file.

```
OdiExportWork "-TODIR=/temp/" "-ZIPFILE_NAME=workexport.zip"
"-EXPORT_KEY=examplekey1"
```

OdiFileAppend

Use this command to concatenate a set of files into a single file.

Usage

```
OdiFileAppend -FILE=<file> -TOFILE=<target_file> [-OVERWRITE=<yes|no>]
[-CASESENS=<yes|no>] [-HEADER=<n>] [-KEEP_FIRST_HEADER=<yes|no]
```

Parameters

Parameters	Mandatory	Description
-FILE=<file>	Yes	Full path of the files to concatenate. Use * to specify generic characters. Examples: /var/tmp/*.log (all files with the log extension in the folder /var/tmp) arch_*.lst (all files starting with arch_ and with the extension lst) The file location is always relative to the data schema directory of its logical schema.
-TOFILE=<target_file>	Yes	Target file.
-OVERWRITE=<yes|no>	No	Indicates if the target file must be overwritten if it already exists. The default value is No.
-CASESENS=<yes|no>	No	Indicates if file search is case-sensitive. By default, Oracle Data Integrator searches files in uppercase (set to No).
-HEADER=<n>	No	Number of header lines to be removed from the source files before concatenation. By default, no lines are removed. When the -HEADER parameter is omitted, the concatenation does not require file edition, and therefore runs faster.
-KEEP_FIRST_HEADER=<yes|no>	No	Keep the header lines of the first file during the concatenation. The default value is Yes.

Examples

Concatenate the files `*.log` of the folder `/var/tmp` into the file `/home/all_files.log`.

```
OdiFileAppend -FILE=/var/tmp/*.log -TOFILE=/home/all_files.log
```

OdiFileDelete

Use this command to delete files or directories.

The most common uses of this tool are described in the following table where:

- x means is supplied
- o means is omitted

-DIR	-FILE	-RECURSE	Behavior
x	x	x	Every file with the name or with a name matching the mask specified in `-FILE` is deleted from `-DIR` and from all of its subdirectories.
x	o	x	The subdirectories from `-FILE` are deleted.
x	x	o	Every file with the name or with a name matching the mask specified in `-FILE` is deleted from `-DIR`.
x	o	o	The `-DIR` is deleted.

Usage

```
OdiFileDelete -DIR=<directory> -FILE=<file> [-RECURSE=<yes|no>]
[-CASESENS=<yes|no>] [-NOFILE_ERROR=<yes|no>] [-FROMDATE=<from_date>]
[-TODATE=<to_date>]
```

Parameters

Parameters	Mandatory	Description
`-DIR=<directory>`	Yes if `-FILE` is omitted	If `-FILE` is omitted, specifies the name of the directory (folder) to delete. If `-FILE` is supplied, specifies the path where files should be deleted from. The directory location is always relative to the data schema directory of its logical schema.

Parameters	Mandatory	Description
-FILE=<file>	Yes if -DIR is omitted	Name or mask of file(s) to delete. If -DIR is not specified, provide the full path. Use * to specify wildcard characters. Examples: /var/tmp/*.log (all files with the log extension of the directory /var/tmp) /arch_*.lst (all files starting with arch_ and with the extension lst) The file location is always relative to the data schema directory of its logical schema.
-RECURSE=<yes|no>	No	If -FILE is omitted, the -RECURSE parameter has no effect: all subdirectories are implicitly deleted. If -FILE is supplied, the -RECURSE parameter specifies if the files should be deleted from this directory and from all of its subdirectories. The default value is Yes.
-CASESENS=<yes|no>	No	Specifies that Oracle Data Integrator should distinguish between uppercase and lowercase when matching file names. The default value is No.
-NOFILE_ERROR=<yes|no>	Yes	Indicates that an error should be generated if the specified directory or files are not found. The default value is Yes.
-FROMDATE=<from_date>	No	All files with a modification date later than this date are deleted. Use the format yyyy/MM/dd hh:mm:ss. The -FROM_DATE is not inclusive. If -FROMDATE is omitted, all files with a modification date earlier than the -TODATE date are deleted. If both -FROMDATE and -TODATE are omitted, all files matching the -FILE parameter value are deleted.

Parameters	Mandatory	Description
-TODATE=<to_date>	No	All files with a modification date earlier than this date are deleted. Use the format yyyy/MM/dd hh:mm:ss. The TO_DATE is not inclusive. If -TODATE is omitted, all files with a modification date later than the -FROMDATE date are deleted. If both -FROMDATE and -TODATE parameters are omitted, all files matching the -FILE parameter value are deleted.

> **Note:**
>
> You cannot delete a file and a directory at the same time by combining the -DIR and -FILE parameters. To achieve that, you must make two calls to OdiFileDelete.

Examples

Delete the file my_data.dat from the directory c:\data\input, generating an error if the file or directory is missing.

```
OdiFileDelete -FILE=c:\data\input\my_data.dat -NOFILE_ERROR=yes
```

Delete all .txt files from the bin directory, but not .TXT files.

```
OdiFileDelete "-FILE=c:\Program Files\odi\bin\*.txt" -CASESENS=yes
```

This statement has the same effect:

```
OdiFileDelete "-DIR=c:\Program Files\odi\bin" "-FILE=*.txt" -CASESENS=yes
```

Delete the directory /bin/usr/nothingToDoHere.

```
OdiFileDelete "-DIR=/bin/usr/nothingToDoHere"
```

Delete all files under the C:\temp directory whose modification time is between 10/01/2008 00:00:00 and 10/31/2008 22:59:00, where 10/01/2008 and 10/31/2008 are not inclusive.

```
OdiFileDelete -DIR=C:\temp -FILE=* -NOFILE_ERROR=NO -FROMDATE=FROMDATE=10/01/2008
00:00:00 -TODATE=10/31/2008 22:59:00
```

Delete all files under the C:\temp directory whose modification time is earlier than 10/31/2008 17:00:00.

```
OdiFileDelete -DIR=C:\temp -FILE=* -NOFILE_ERROR=YES -TODATE=10/31/2008 17:00:00
```

Delete all files under the C:\temp directory whose modification time is later than 10/01/2008 08:00:00.

```
OdiFileDelete -DIR=C:\temp -FILE=* -NOFILE_ERROR=NO -FROMDATE=10/01/2008 08:00:00
```

OdiFileCopy

Use this command to copy files or folders.

Usage

```
OdiFileCopy -DIR=<directory> -TODIR=<target_directory> [-OVERWRITE=<yes|no>]
[-RECURSE=<yes|no>] [-CASESENS=<yes|no>]

OdiFileCopy -FILE=<file> -TOFILE=<target_file>|-TODIR=<target_directory>
[-OVERWRITE=<yes|no>] [-RECURSE=<yes|no>] [-CASESENS=<yes|no>]
```

Parameters

Parameters	Mandatory	Description
`-DIR=<directory>`	Yes if `-FILE` is omitted	Directory (or folder) to copy. The directory location is always relative to the data schema directory of its logical schema.
`-FILE=<file>`	Yes if `-DIR` is omitted	The full path of the files to copy. Use `*` to specify the generic character. Examples: `/var/tmp/*.log` (all files with the `log` extension in folder `/var/tmp`) `arch_*.lst` (all files starting with `arch_` and with the extension `lst`) The file location is always relative to the data schema directory of its logical schema.
`-TODIR=<target_directory>`	Yes if `-DIR` is specified	Target directory for the copy. If a directory is copied (`-DIR`), this parameter indicates the name of the copied directory. If one or several files are copied (`-FILE`), this parameter indicates the destination directory.
`-TOFILE=<target_file>`	Yes if `-TODIR` is omitted	Destination file(s). This parameter cannot be used with parameter `-DIR`. This parameter contains: • The name of the destination file if only one file is copied (no generic character). • The mask of the new name of the destination files if several files are copied. Note that `-TODIR` and `-TOFILE` are exclusive parameters. If both are specified, only `-TODIR` is taken into account, and `-TOFILE` is ignored.

Parameters	Mandatory	Description
-SRC_LSCHEMA=<source_file>	No	The file located on a data server, based on the Logical Schema value. For example, the logical schema may point to a Hadoop Data Server and the tool will access the file from that data server if the file needs to be accessed from HDFS.
-TGT_LSCHEMA=<target_file>	No	The file located on a data server, based on the Logical Schema value. For example, the logical schema may point to a Hadoop Data Server and the tool will access the file from that data server if the file needs to be accessed from HDFS.
-OVERWRITE=<yes|no>	No	Indicates if the files of the folder are overwritten if they already exist. The default value is No.
-RECURSE=<yes|no>	No	Indicates if files are copied recursively when the directory contains other directories. The value No indicates that only the files within the directory are copied, not the subdirectories. The default value is Yes.
-CASESENS=<yes|no>	No	Indicates if file search is case-sensitive. By default, Oracle Data Integrator searches for files in uppercase (set to No).

Examples

Copy the file `hosts` from the directory `/etc` to the directory `/home`.

```
OdiFileCopy -FILE=/etc/hosts -TOFILE=/home/hosts
```

Copy all `*.csv` files from the directory `/etc` to the directory `/home` and overwrite.

```
OdiFileCopy -FILE=/etc/*.csv -TODIR=/home -OVERWRITE=yes
```

OdiFileMove

Use this command to move or rename files or a directory into files or a directory.

Usage

```
OdiFileMove -FILE=<file> -TODIR=<target_directory> -TOFILE=<target_file>
[-OVERWRITE=<yes|no>] [-RECURSE=<yes|no>] [-CASESENS=<yes|no>]

OdiFileMove -DIR=<directory> -TODIR=<target_directory> [-OVERWRITE=<yes|no>]
[-RECURSE=<yes|no>] [-CASESENS=<yes|no>]
```

Parameters

Parameters	Mandatory	Description
-DIR=<directory>	Yes if -FILE is omitted	Directory (or folder) to move or rename. The directory location is always relative to the data schema directory of its logical schema.
-FILE=<file>	Yes if -DIR is omitted	Full path of the file(s) to move or rename. Use * for generic characters. Examples: /var/tmp/*.log (all files with the log extension in the directory /var/tmp) arch_*.lst (all files starting with arch_ and with the extension lst) The file location is always relative to the data schema directory of its logical schema.
-TODIR=<target_directory>	Yes if -DIR is specified	Target directory of the move. If a directory is moved (-DIR), this parameter indicates the new name of the directory. If a file or several files are moved (-FILE), this parameter indicates the target directory.
-TOFILE=<target_file>	Yes if -TODIR is omitted	Target file(s). This parameter cannot be used with parameter -DIR. This parameter is: • The new name of the target file if one single file is moved (no generic character). • The mask of the new file names if several files are moved.
-OVERWRITE=<yes|no>	No	Indicates if the files or directory are overwritten if they exist. The default value is No.
-RECURSE=<yes|no>	No	Indicates if files are moved recursively when the directory contains other directories. The value No indicates that only files contained in the directory to move (not the subdirectories) are moved. The default value is Yes.
-CASESENS=<yes|no>	No	Indicates if file search is case-sensitive. By default, Oracle Data Integrator searches for files in uppercase (set to No).

Examples

Rename the `hosts` file to `hosts.old`.

```
OdiFileMove -FILE=/etc/hosts -TOFILE=/etc/hosts.old
```

Move the file `hosts` from the directory `/etc` to the directory `/home/odi`.

```
OdiFileMove -FILE=/etc/hosts -TOFILE=/home/odi/hosts
```

Move all files `*.csv` from directory `/etc` to directory `/home/odi` with overwrite.

```
OdiFileMove -FILE=/etc/*.csv -TODIR=/home/odi -OVERWRITE=yes
```

Move all `*.csv` files from directory `/etc` to directory `/home/odi` and change their extension to `.txt`.

```
OdiFileMove -FILE=/etc/*.csv -TOFILE=/home/odi/*.txt -OVERWRITE=yes
```

Rename the directory `C:\odi` to `C:\odi_is_wonderful`.

```
OdiFileMove -DIR=C:\odi -TODIR=C:\odi_is_wonderful
```

Move the directory `C:\odi` and its subfolders into the directory `C:\Program Files\odi`.

```
OdiFileMove -DIR=C:\odi "-TODIR=C:\Program Files\odi" -RECURSE=yes
```

OdiFileWait

Use this command to manage file events. This command regularly scans a directory and waits for a number of files matching a mask to appear, until a given timeout is reached. When the specified files are found, an action on these files is triggered.

Usage

```
OdiFileWait -DIR=<directory> -PATTERN=<pattern>
[-ACTION=<DELETE|COPY|MOVE|APPEND|ZIP|NONE>] [-TODIR=<target_directory>]
[-TOFILE=<target_file>] [-OVERWRITE=<yes|no>] [-CASESENS=<yes|no>]
[-FILECOUNT=<n>] [-TIMEOUT=<n>] [-POLLINT=<n>] [-HEADER=<n>]
[-KEEP_FIRST_HEADER=<yes|no>] [-NOFILE_ERROR=<yes|no>]
```

Parameters

Parameters	Mandatory	Description
-ACTION=<DELETE|COPY|MOVE|APPEND|ZIP|NONE>	No	Action taken on the files found: DELETE: Delete the files found. COPY: Copy the files found into the directory -TODIR. MOVE: Move or rename the files found into folder -TODIR by naming them as specified by -TOFILE. APPEND: Concatenates all files found and creates a result file -TOFILE. Source files are deleted. ZIP: Compress the files found and store them in ZIP file -TOFILE. NONE (default): No action is performed.
-DIR=<directory>	Yes	Directory (or folder) to scan. The directory location is always relative to the data schema directory of its logical schema.
-PATTERN=<pattern>	Yes	Mask of file names to scan. Use * to specify the generic characters. Examples: *.log (all files with the log extension) arch_*.lst (all files starting with arch_ and with the extension lst)
-TODIR=<target_directory>	No	Target directory of the action. When the action is: COPY: Directory where the files are copied. MOVE: Directory where the files are moved.

Parameters	Mandatory	Description
-TOFILE=<target_file>	No	Destination file(s). When the action is: MOVE: Renaming mask of the moved files. APPEND: Name of the file resulting from the concatenation. ZIP: Name of the resulting ZIP file. COPY: Renaming mask of the copied files. Renaming rules: • Any alphanumeric character is replaced in the original file name with the alphanumeric characters specified for <target_file>. • ? at -TOFILE leaves origin symbol on this position. • * at -TOFILE means all remaining symbols from origin file name.
-OVERWRITE=<yes|no>	No	Indicates if the destination file(s) will be overwritten if they exist. The default value is No. Note that if this option is used with APPEND, the target file will only contain the contents of the latest file processed.
-CASESENS=<yes|no>	No	Indicates if file search is case-sensitive. By default, Oracle Data Integrator searches files in uppercase (set to No).
-FILECOUNT=<n>	No	Maximum number of files to wait for (the default value is 0). If this number is reached, the command ends. The value 0 indicates that Oracle Data Integrator waits for all files until the timeout is reached. If this parameter is 0 and the timeout is also 0, this parameter is then forced implicitly to 1.
-TIMEOUT=<n>	No	Maximum waiting time in milliseconds (the default value is 0). If this delay is reached, the command yields control to the following command and uses its value -FILECOUNT. The value 0 is used to specify an infinite waiting time (wait until the maximum number of messages to read as specified in the parameter -FILECOUNT).

Parameters	Mandatory	Description
-POLLINT=<n>	No	Interval in milliseconds to search for new files. The default value is 1000 (1 second), which means that Oracle Data Integrator looks for new messages every second. Files written during the OdiFileWait are taken into account only after being closed (file size unchanged) during this interval.
-HEADER=<n>	No	This parameter is valid only for the APPEND action. Number of header lines to suppress from the files before concatenation. The default value is 0 (no processing).
-KEEP_FIRST_HEADER=<yes|no>	No	This parameter is valid only for the APPEND action. Keeps the header lines of the first file during the concatenation. The default value is Yes.
-NOFILE_ERROR=<yes|no>	No	Indicates the behavior if no file is found. The default value is No, which means that no error is generated if no file is found.

Examples

Wait indefinitely for file `flag.txt` in directory `c:\events` and proceed when this file is detected.

```
OdiFileWait -ACTION=NONE -DIR=c:\events -PATTERN=flag.txt -FILECOUNT=1
-TIMEOUT=0 -POLLINT=1000
```

Wait indefinitely for file `flag.txt` in directory `c:\events` and suppress this file when it is detected.

```
OdiFileWait -ACTION=DELETE -DIR=c:\events -PATTERN=flag.txt -FILECOUNT=1
-TIMEOUT=0 -POLLINT=1000
```

Wait for the sales files `*.dat` for 5 minutes and scan every second in directory `c:\sales_in`, then concatenate into file `sales.dat` in directory `C:\sales_ok`. Keep the header of the first file.

```
OdiFileWait -ACTION=APPEND -DIR=c:\sales_in -PATTERN=*.dat
TOFILE=c:\sales_ok\sales.dat -FILECOUNT=0 -TIMEOUT=350000 -POLLINT=1000
-HEADER=1 -KEEP_FIRST_HEADER=yes -OVERWRITE=yes
```

Wait for the sales files `*.dat` for 5 minutes every second in directory `c:\sales_in`, then copy these files into directory `C:\sales_ok`. Do not overwrite.

```
OdiFileWait -ACTION=COPY -DIR=c:\sales_in -PATTERN=*.dat -TODIR=c:\sales_ok
-FILECOUNT=0 -TIMEOUT=350000 -POLLINT=1000 -OVERWRITE=no
```

Wait for the sales files `*.dat` for 5 minutes every second in directory `c:\sales_in` and then archive these files into a ZIP file.

```
OdiFileWait -ACTION=ZIP -DIR=c:\sales_in -PATTERN=*.dat
-TOFILE=c:\sales_ok\sales.zip -FILECOUNT=0 -TIMEOUT=350000
-POLLINT=1000 -OVERWRITE=yes
```

Wait for the sales files `*.dat` for 5 minutes every second into directory `c:\sales_in`, then move these files into directory `C:\sales_ok`. Do not overwrite. Append `.bak` to the file names.

```
OdiFileWait -ACTION=MOVE -DIR=c:\sales_in -PATTERN=*.dat
-TODIR=c:\sales_ok -TOFILE=*.bak -FILECOUNT=0 -TIMEOUT=350000
-POLLINT=1000 -OVERWRITE=no
```

OdiFtp

Use this command to use the FTP protocol to connect to a remote system and to perform standard FTP commands on the remote system. Trace from the script is recorded against the Execution Details of the task representing the OdiFtp step in Operator Navigator.

Usage

```
OdiFtp -HOST=<ftp server host name> -USER=<ftp user>
[-PASSWORD=<ftp user password>] -REMOTE_DIR=<remote dir on ftp host>
-LOCAL_DIR=<local dir> [-PASSIVE_MODE=<yes|no>] [-TIMEOUT=<time in seconds>]
[-STOP_ON_FTP_ERROR=<yes|no>] -COMMAND=<command>
```

Parameters

Parameters	Mandatory	Description
`-HOST=<ftp server host name>`	Yes	Host name of the FTP server.
`-USER=<ftp user>`	Yes	User on the FTP server.
`-PASSWORD=<ftp user password>`	No	Password of the FTP user.
`-REMOTE_DIR=<remote dir on ftp host>`	Yes	Directory path on the remote FTP host.
`-LOCAL_DIR=<local dir>`	Yes	Directory path on the local machine.
`-PASSIVE_MODE=<yes|no>`	No	If set to No, the FTP session uses Active Mode. The default value is Yes, which means the session runs in passive mode.
`-TIMEOUT=<time in seconds>`	No	Time in seconds after which the socket connection times out.
`-STOP_ON_FTP_ERROR=<yes|no>`	No	If set to Yes (default), the step stops when an FTP error occurs instead of running to completion.

Parameters	Mandatory	Description
-COMMAND=<command>	Yes	Raw FTP command to execute. For a multiline command, pass the whole command as raw text after the OdiFtp line without the -COMMAND parameter. Supported commands: APPE, CDUP, CWD, DELE, LIST, MKD, NLST, PWD, QUIT, RETR, RMD, RNFR, RNTO, SIZE, STOR

Examples

Execute a script on a remote host that makes a directory, changes directory into the directory, puts a file into the directory, and checks its size. The script appends another file, checks the new size, and then renames the file to dailyData.csv. The -STOP_ON_FTP_ERROR parameter is set to No so that the script continues even if the directory exists.

```
OdiFtp -HOST=machine.example.com -USER=odiftpuser -PASSWORD=<password>
-LOCAL_DIR=/tmp -REMOTE_DIR=c:\temp -PASSIVE_MODE=YES -STOP_ON_FTP_ERROR=No
MKD dataDir
CWD dataDir
STOR customers.csv
SIZE customers.csv
APPE new_customers.csv customers.csv
SIZE customers.csv
RNFR customers.csv
RNTO dailyData.csv
```

OdiFtpGet

Use this command to download a file from an FTP server.

Usage

```
OdiFtpGet -HOST=<ftp server host name> -USER=<ftp user>
[PASSWORD=<ftp user password>] -REMOTE_DIR=<remote dir on ftp host>
[-REMOTE_FILE=<file name under the -REMOTE_DIR>] -LOCAL_DIR=<local dir>
[-LOCAL_FILE=<file name under the -LOCAL_DIR>] [-PASSIVE_MODE=<yes|no>]
[-TIMEOUT=<time in seconds>]
```

> **Note:**
>
> If a Local or Remote file name needs to have % as part of its name, %25 needs to be passed instead of just %.
>
> %25 will resolve automatically to %.
>
> For example, if file name needs to be temp%result, it should be passed as REMOTE_FILE=temp%25result or -LOCAL_FILE=temp%25result.

Parameters

Parameters	Mandatory	Description
-HOST=<host name of the ftp server>	Yes	Host name of the FTP server.
-USER=<host name of the ftp user>	Yes	User on the FTP server.
-PASSWORD=<password of the ftp user>	No	Password of the FTP user.
-REMOTE_DIR=<dir on the ftp host>	Yes	Directory path on the remote FTP host.
-REMOTE_FILE=<file name under -REMOTE DIR>	No	File name under the directory specified in the -REMOTE_DIR argument. If this argument is missing, the file is copied with the -LOCAL_FILE file name. If the -LOCAL_FILE argument is also missing, the -LOCAL_DIR is copied recursively to the -REMOTE_DIR.
-LOCAL_DIR=<local dir path>	Yes	Directory path on the local machine.
-LOCAL_FILE=<local file>	No	File name under the directory specified in the -LOCAL_DIR argument. If this argument is missing, all files and directories under the -LOCAL_DIR are copied recursively to the -REMOTE_DIR. To filter the files to be copied, use * to specify the generic characters. Examples: • *.log (all files with the log extension) • arch_*.lst (all files starting with arch_ and with the extension lst)
-PASSIVE_MODE=<yes|no>]	No	If set to No, the FTP session uses Active Mode. The default value is Yes, which means the session runs in passive mode.
-TIMEOUT=<time in seconds>	No	The time in seconds after which the socket connection times out.
-TGT_LSCHEMA=<target_file>	No	The file located on a data server resolved based on the Logical Schema value. For example, the LSCHEMA may point to a Hadoop Data Server and the tool will access the file from that data server.

Examples

Copy the remote directory /test_copy555 on the FTP server recursively to the local directory C:\temp\test_copy.

```
OdiFtpGet -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password> -
LOCAL_DIR=C:\temp\test_copy -REMOTE_DIR=/test_copy555
```

Copy all files matching the `Sales*.txt` pattern under the remote directory `/` on the FTP server to the local directory `C:\temp\` using Active Mode for the FTP connection.

```
OdiFtpGet -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password> -
LOCAL_DIR=C:\temp -LOCAL_FILE=Sales*.txt -REMOTE_DIR=/ -PASSIVE_MODE=NO
```

OdiFtpPut

Use this command to upload a local file to an FTP server.

Usage

```
OdiFtpPut -HOST=<ftp server host name> -USER=<ftp user>
[PASSWORD=<ftp user password>] -REMOTE_DIR=<remote dir on ftp host>
[-REMOTE_FILE=<file name under the -REMOTE_DIR>] -LOCAL_DIR=<local dir>
[-LOCAL_FILE=<file name under the -LOCAL_DIR>] [-PASSIVE_MODE=<yes|no>]
[-TIMEOUT=<time in seconds>]
```

> **Note:**
>
> If a Local or Remote file name needs to have % as part of its name, %25 needs to be passed instead of just %.
>
> %25 will resolve automatically to %.
>
> For example, if file name needs to be `temp%result`, it should be passed as `REMOTE_FILE=temp%25result` or `-LOCAL_FILE=temp%25result`.

Parameters

Parameters	Mandatory	Description
`-HOST=<host name of the ftp server>`	Yes	Host name of the FTP server.
`-USER=<host name of the ftp user>`	Yes	User on the FTP server.
`-PASSWORD=<password of the ftp user>`	No	Password of the FTP user.
`-REMOTE_DIR=<dir on the ftp host>`	Yes	Directory path on the remote FTP host.
`-REMOTE_FILE=<file name under -REMOTE DIR>`	No	File name under the directory specified in the `-REMOTE_DIR` argument. If this argument is missing, the file is copied with the `-LOCAL_FILE` file name. If the `-LOCAL_FILE` argument is also missing, the `-LOCAL_DIR` is copied recursively to the `-REMOTE_DIR`.
`-LOCAL_DIR=<local dir path>`	Yes	Directory path on the local machine.

Parameters	Mandatory	Description
-LOCAL_FILE=<local file>	No	File name under the directory specified in the -LOCAL_DIR argument. If this argument is missing, all files and directories under the -LOCAL_DIR are copied recursively to the -REMOTE_DIR. To filter the files to be copied, use * to specify the generic characters. Examples: • *.log (all files with the log extension) • arch_*.lst (all files starting with arch_ and with the extension lst)
-PASSIVE_MODE=<yes|no>	No	If set to No, the FTP session uses Active Mode. The default value is Yes, which means the session runs in passive mode.
-TIMEOUT=<time in seconds>	No	The time in seconds after which the socket connection times out.

> **Note:**
>
> For OdiFtp execution to be successful, you must have LIST privilege in the user's home directory.

Examples

Copy the local directory C:\temp\test_copy recursively to the remote directory / test_copy555 on the FTP server.

```
OdiFtpPut -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password>
 -LOCAL_DIR=C:\temp\test_copy -REMOTE_DIR=/test_copy555"
```

Copy all files matching the Sales*.txt pattern under the local directory C:\temp\ to the remote directory / on the FTP server.

```
OdiFtpPut -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password> -
LOCAL_DIR=C:\temp -LOCAL_FILE=Sales*.txt -REMOTE_DIR=/
```

Copy the Sales1.txt file under the local directory C:\temp\ to the remote directory / on the FTP server as a Sample1.txt file.

```
OdiFtpPut -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password> -
LOCAL_DIR=C:\temp -LOCAL_FILE=Sales1.txt -REMOTE_DIR=/Sample1.txt
```

OdiGenerateAllScen

Use this command to generate a set of scenarios from design-time components (Packages, Mappings, Procedures, or Variables) contained in a folder or project, filtered by markers.

Usage

```
OdiGenerateAllScen -PROJECT=<project_id> [-FOLDER=<folder_id>]
[-MODE=<REPLACE|REGENERATE|CREATE>] [-GRPMARKER=<marker_group_code>
[-MARKER=<marker_code>] [-MATERIALIZED=<yes|no>]
[-GENERATE_MAP=<yes|no>] [-GENERATE_PACK=<yes|no>]
[-GENERATE_POP=<yes|no>] [-GENERATE_TRT=<yes|no>]
[-GENERATE_VAR=<yes|no>]
```

Parameters

Parameters	Mandatory	Description
-PROJECT=<project_id>	Yes	Internal ID of the Project containing the components to generate scenarios for.
-FOLDER=<folder_id>	No	Global ID of the Folder containing the components to generate scenarios for.

Parameters	Mandatory	Description
-MODE=<REPLACE|REGENERATE|CREATE>	No	Scenario generation mode: • REPLACE (default): Overwrites for each object the last scenario version with a new one with the same internal ID, name and version. Sessions, scenario reports and schedules are deleted. If no scenario exists for an object, a scenario with version number 001 is created. • REGENERATE: Overwrites for each object the last scenario version with a new one with the same internal ID, name and version. It preserves the schedule, sessions, scenario reports, variable selections, and concurrent execution control settings. If no scenario exists for an object, no scenario is created using this mode. • CREATE: Creates a new scenario with the same name as the latest scenario generated for the component, with the version number automatically incremented (if the latest version is an integer) or set to the current date (if the latest version is not an integer). If no scenario exists for an object, a scenario named after the object with version number 001 is created. New scenarios are named after the component according to the *Scenario Naming Convention* user parameter.
-GRPMARKER=<marker_group_code>	No	Group containing the marker used to filter the components for which scenarios must be generated. When -GRPMARKER and -MARKER are specified, scenarios will be (re-)generated only for components flagged with the marker identified by the marker code and the marker group code.

Parameters	Mandatory	Description
-MARKER=<marker_code>	No	Marker used to filter the components for which scenarios must be generated. When -GRPMARKER and -MARKER are specified, scenarios will be (re-)generated only for components flagged with the marker identified by the marker code and the marker group code.
-MATERIALIZED=<yes|no>	No	Specifies whether scenarios should be generated as if all underlying objects are materialized. The default value is No.
-GENERATE_MAP=<yes|no>	No	Specifies whether scenarios should be generated from the mapping. The default value is No.
-GENERATE_PACK=<yes|no>	No	Specifies whether scenarios attached to packages should be (re-)generated. The default value is Yes.
-GENERATE_POP=<yes|no>	No	Specifies whether scenarios attached to mappings should be (re-)generated. The default value is No.
-GENERATE_TRT=<yes|no>	No	Specifies whether scenarios attached to procedures should be (re-)generated. The default value is No.
-GENERATE_VAR=<yes|no>	No	Specifies whether scenarios attached to variables should be (re-)generated. The default value is No.

Examples

Generate all scenarios in the project whose ID is 1003 for the current repository.

```
OdiGenerateAllScen -PROJECT=1003
```

OdiImportObject

Use this command to import the contents of an export file into a repository. This command reproduces the behavior of the import feature available from the user interface.

Use caution when using this tool. It may work incorrectly when importing objects that depend on objects that do not exist in the repository. It is recommended that you use this API for importing high-level objects (projects, models, and so on).

> **WARNING:**
>
> The import type and the order in which objects are imported into a repository should be carefully specified. Refer to the chapter Exporting and Importing in *Developing Integration Projects with Oracle Data Integrator* for more information on import.

Usage

```
OdiImportObject -FILE_NAME=<FileName> [-WORK_REP_NAME=<workRepositoryName>]
[-IMPORT_MODE=<DUPLICATION|SYNONYM_INSERT|SYNONYM_UPDATE|SYNONYM_INSERT_UPDATE>]
[-EXPORT_KEY=<key>] [-UPGRADE_KEY=<upgradeKey>]
[IMPORT_WITHOUT_CIPHER_DATA=<yes|no>]
```

Parameters

Parameters	Mandatory	Description
`-FILE_NAME=<FileName>`	Yes	Name of the XML export file to import.
`-WORK_REP_NAME=<workRepositoryName>`	No	Name of the work repository into which the object must be imported. This work repository must be defined in the connected master repository. If this parameter is not specified, the object is imported into the current master or work repository.
`-IMPORT_MODE=<DUPLICATION|SYNONYM_INSERT|SYNONYM_UPDATE|SYNONYM_INSERT_UPDATE>`	Yes	Import mode for the object. The default value is `DUPLICATION`. For more information about import types, see Import Modes in *Developing Integration Projects with Oracle Data Integrator*.
`-EXPORT_KEY=<key>`	No[1]	Specifies a cryptographic private key used to encrypt sensitive cipher data. You must specify this key when importing the exported object in order to import the cipher data.
`-UPGRADE_KEY=<upgradeKey>`	No	Upgrade key to import repository objects from earlier versions of Oracle Data Integrator (pre-12*c*).
`-IMPORT_WITHOUT_CIPHER_DATA=<yes|no>`	No[2]	When set to Yes, specifies that sensitive (cipher) values should be set to null in the object when it is imported. When set to No or when this parameter is omitted, you must include the `-EXPORT_KEY` parameter and specify a valid key. The default value is No.

[1] If the `-EXPORT_KEY` parameter is not specified, the `-IMPORT_WITHOUT_CIPHER_DATA` parameter must be specified, and must be set to Yes.

[2] If `-EXPORT_WITHOUT_CIPHER_DATA` is not specified, or if it is specified and set to No, you must specify the `-EXPORT_KEY` parameter with a valid key value.

Examples

Import the `/temp/DW01.xml` export file (a project) into the `WORKREP` work repository using `DUPLICATION` mode.

```
OdiImportObject -FILE_NAME=/temp/DW01.xml -WORK_REP_NAME=WORKREP
-IMPORT_MODE=DUPLICATION -EXPORT_KEY=examplekey1
```

OdiImportScen

Use this command to import a scenario into the current work repository from an export file.

Usage

```
OdiImportScen -FILE_NAME=<FileName>
[-IMPORT_MODE=<DUPLICATION|SYNONYM_INSERT|SYNONYM_UPDATE|SYNONYM_INSERT_UPDATE>]
[-EXPORT_KEY=<key>] [-IMPORT_SCHEDULE=<yes|no>] [-FOLDER=<parentFolderGlobalId>]
[-UPGRADE_KEY=<upgradeKey>] [IMPORT_WITHOUT_CIPHER_DATA=<yes|no>]
```

Parameters

Parameters	Mandatory	Description
`-FILE_NAME=<FileName>`	Yes	Name of the export file.
`-IMPORT_MODE=<DUPLICATION|SYNONYM_INSERT|SYNONYM_UPDATE|SYNONYM_INSERT_UPDATE>`	No	Import mode of the scenario. The default value is `DUPLICATION`. For more information about import types, see Import Modes in *Developing Integration Projects with Oracle Data Integrator*.
`-EXPORT_KEY=<key>`	No[1]	Specifies a cryptographic private key used to encrypt sensitive cipher data. You must specify this key when importing the exported object in order to import the cipher data.
`-IMPORT_SCHEDULE=<yes|no>`	No	Imports the schedules contained in the scenario export file. The default value is No.
`-FOLDER=<parentFolderGlobalId>`	No	Global ID of the parent scenario folder. This parameter is used only when the import mode is set to `DUPLICATION`. It is ignored for all other import modes since the `OdiImportScen` command only looks to change parents when the import mode is `DUPLICATION`.
`-UPGRADE_KEY=<upgradeKey>`	No	Upgrade key to import repository objects from earlier versions of Oracle Data Integrator (pre-12*c*).

Parameters	Mandatory	Description
-IMPORT_WITHOUT_CIPHER_DATA=<yes|no>	No[2]	When set to Yes, specifies that sensitive (cipher) values should be set to null in the object when it is imported. When set to No or when this parameter is omitted, you must include the -EXPORT_KEY parameter and specify a valid key. The default value is No.

[1] If the -EXPORT_KEY parameter is not specified, the -IMPORT_WITHOUT_CIPHER_DATA parameter must be specified, and must be set to Yes.
[2] If -IMPORT_WITHOUT_CIPHER_DATA is not specified, or if it is specified and set to No, you must specify the -EXPORT_KEY parameter with a valid key value.

Examples

Import the /temp/load_dwh.xml export file (a scenario) into the current work repository using DUPLICATION mode.

```
OdiImportScen -FILE_NAME=/temp/load_dwh.xml -IMPORT_MODE=DUPLICATION
-EXPORT_KEY=examplekey1
```

OdiInvokeRESTfulService

InvokeRESTfulService tool is used to invoke any REST service from ODI. The request sent to the service can be either provided in a request file, or directly provided in the tool command (<RequestBody>). The response of the RESTful service request is fed to a file that can be used in Oracle Data Integrator. The tool also supports multipart request and parameters are provided to specify multipart body part details.

The tool does not accept a REST URL directly. Instead, the tool accepts a REST data server and an operation to invoke, from which the REST URL is derived for the REST invocation. For details of REST operation and data server and how they are defined using ODI Studio UI, refer to the REST data server and Studio UI documentation.

InvokeRESTfulService tool has two important features:

- **Paginated Invocation** — There are certain REST services that restrict the maximum amount of data (or records) that can be retrieved in a single service invocation. Such services provide the response of the request in multiple pages and thus these responses may extend to subsequent pages. You can configure the tool, to invoke such services and fetch the complete response in a single invocation of the tool. The tool will internally make repeated invocations for each page and fetch a complete response by specifying some pagination related tool parameters.
- **Chunk Upload Support** — There are certain REST services which imposes restriction on the amount of data that can be uploaded in a single invocation. These services have support for uploading data in chunks. This REST tool performs such chunk upload operation in a single tool invocation.

Based on the specified parameters, the tool identifies whether the invocation is a regular invocation, chunk upload invocation or pagination.

Usage

Use this command to invoke a Restful service from ODI.

```
OdiInvokeRESTfulService [-CONTEXT=<ODI_Context>]
-LSCHEMA=<Logical_Schema> [-REQUEST_BODY_FILE=<Request_File> |
-REQUEST_BODY=<RequestBody> | <RequestBody>] [-
REQUEST_HDFS_LSCHEMA=<Request_HDFS_Logical_Schema>] [-
CHUNK_SIZE=<Chunk_Size>] [-CHUNK_PREFIX=<Chunk_Prefix>][-
PRE_OPERATION=<Pre_Operation>] [-
PRE_OPERATION_BODY=<Pre_Operation_Body>] -OPERATION=<Rest Operation>
[-POST_OPERATION=<Post_Operation>] [-
POST_OPERATION_BODY=<Post_Operation_Body>] [-HEADER.<name>=<value>]*
[-HEADER_IGNORE_DEFAULT=<YES|NO>] [-
REQUEST_BODY_PART_NAME=<Body_Part_Name> & -
REQUEST_BODY_PART_CONTENT_TYPE=<Body_Part_Content_Type> & -
REQUEST_BODY_PART_VALUE=<Body Part Value>]* [[-
REQUEST_QUERY.name=<value>]* | -
ENCODED_REQUEST_QUERY_STRING=<Encoded_Request_Query_String>] [-
REQUEST_QUERY_IGNORE_DEFAULT=<YES|NO>]  [-
REQUEST_TEMPLATE.<Variable_Name>=<Variable_Value>]*  [-
REQUEST_TEMPLATE_IGNORE_DEFAULT=<YES|NO>] [-
RESPONSE_FILE=<Response_File>] [-RESPONSE_HDFS_LSCHEMA=<Response HDFS
LSchema>][-RESPONSE_MODE=<NEW_FILE|FILE_APPEND>] [-
RESPONSE_FILE_CHARSET=<javaCharset>][-TIMEOUT=<timeout>] [-
RETRY_COUNT=<Retry_Count>] [-RETRY_INTERVAL=<Retry_Interval>] [-
FAILURE_STATUS_CODES=<Failure Status codes>] [-
TRACE_FILE=<Trace_File>] [-TRACE_FILE_MODE=<NEW_FILE|FILE_APPEND>] [-
NEXT_REQUEST_RESOLVER=<Next_Request_Resolver>] [-
TOTAL_COUNT_FIELD_RESOLVER=<Total_Count_Field_Resolver>] [-
RESOLVER_OVERWRITE_CLASS=<Resolver_Overwrite_Class>] [-
RESPONSE_DATA_CONTAINER=<Response_Data_Container>]
[<RequestBody>]
```

Parameters

Parameter	Mandatory	Description
CONTEXT	No	ODI Context. If not specified, the execution will happen in the context of the calling session.
LSCHEMA	Yes	Logical schema configured for REST data source.
REQUEST_BODY_FILE	No	The name of the file containing the request body. The request body can be directly provided (on the next line after) the tool call (<RequestBody>) or as the value of -REQUEST_BODY instead of specifying this as a parameter. If all are specified, -REQUEST_BODY_FILE parameter will be considered. This parameter will be ignored for multipart request.
REQUEST_BODY	No	Request body can be given in this parameter. See REQUEST_BODY_FILE and <RequestBody> parameters for more details.

Parameter	Mandatory	Description
REQUEST_HDFS_LSCHEMA	No	HDFS file configuration for the request file. This is applicable only if request body is a file which is in HDFS format. This configuration is applicable for multipart body contents that are files.
CHUNK_SIZE	No	The input file specified through parameter is REQUEST_BODY_FILE is broken into smaller chunk of this specified size. These files will be created in a temporary directory. For Example — If the input file name is data.txt, then the chunk names will be data-a,data-b,..data-aa,data-ab,..,data-aaa,data-aab etc. The tool will also maintain the following runtime templates associated with chunk depending upon which chunk is being considered: odi.CHUNK_PATH odi.CHUNK_NAME odi.CHUNK_SIZE odi.CHUNK_INDEX The presence of this parameter indicates that the tool invocation is for **Chunk Upload**.
CHUNK_PREFIX	No	If a directory path is specified in REQUEST_BODY_FILE parameter, then all the files in that directory with file names starting with this prefix will be considered as the chunk files. If REQUEST_BODY_FILE parameter is pointing to the actual input file, then this prefix will be used to construct the chunk file names. Suppose the given value is “segment_”, then chunk files will be named segment_a, segment_b, segment_c….., segment_aa, segment_ab….., segment_aaa and so on. If this is not specified, and REQUEST_BODY_FILE parameter contains actual input file path, then name of the input file will be taken as the prefix. Suppose the name of input file is data.txt, then prefix will be “data”. This will also be as the value of runtime template odi.CHUNK_PREFIX at the time of chunk file creation. If this prefix parameter is not specified and the REQUEST_BODY_FILE parameter points to a directory, then all file contents of the directory are considered as chunk files to be uploaded. The presence of this parameter indicates that the tool invocation is for **Chunk Upload**.
PRE_OPERATION	No	REST operation defined in the physical schema. If specified, this operation will be the first operation to get invoked and it will be invoked only once. This operation will not consider the following tool parameters: HEADER, REQUEST_QUERY, ENCODED and REQUEST_TEMPLATE while invoking the REST request.

Parameter	Mandatory	Description
PRE_OPERATION_BODY	No	This is used to specify the request body for the invocation of PRE_OPERATION. User can specify a file path or runtime template {odi.CHUNK_PATH}. If a file path (HDFS or Local) is specified, then that will be added as the request body when this operation is invoked. If runtime template {odi.CHUNK_PATH} is specified, then the following runtime templates also will be assigned the values corresponding to first chunk file: • odi.CHUNK_PATH — Gets the path of the first chunk • odi.CHUNK_NAME — Name of the first chunk file • odi.CHUNK_SIZE — Size of first chunk file • odi.CHUNK_INDEX — 0 (Index of first chunk) If REQUEST_BODY_PART_VALUE parameter is specified, then this chunk is uploaded as a multipart body part. Otherwise this chunk file is set in the request body. If this parameter is not specified, then the PRE_OPERATION will be invoked without a request body.
OPERATION	Yes	REST operation defined in the physical schema. This is the main operation and it is mandatory. If specified, this operation will get invoked after the operation specified by PRE_OPERATIONand before the operation specified by POST_OPERATION. This is the only operation that may get called repeatedly. This operation may be called multiple times only for Chunk Upload type invocation or Pagination invocation. For Chunk Upload invocation, this operation will be called in a loop for each file chunk available. If the first chunk is uploaded by PRE_OPERATION (see parameter PRE_OPERATION_BODY), then this operation won't be invoked for first chunk. Similarly if the last chunk is expected to be uploaded by POST_OPERATION (see parameter POST_OPERATION_BODY), then this operation won't be invoked for last chunk. For example: - The total chunks created or identified is 10, PRE_OPERATION_BODY = {odi.CHUNK_PATH} and POST_OPERATION_BODY = {odi.CHUNK_PATH}, then operation specified by OPERATION parameter will be executed 8 times. When this operation is being invoked for i th chunk, then runtime templates in this operation will be replaced with following values. • odi.CHUNK_PATH— denotes path of i th chunk • odi.CHUNK_NAME — denotes the name of i th chunk • odi.CHUNK_SIZE — denotes size of i thchunk • odi.CHUNK_INDEX — denotes (i-1). Since the index starts from 0.
POST_OPERATION	No	REST operation defined in the physical schema. If specified, this operation will be the last operation to get invoked and it will be invoked only once. This operation will not consider the tool parameters such as HEADER, REQUEST_QUERY, ENCODED and REQUEST_TEMPLATE while invoking the REST request.

Parameter	Mandatory	Description
POST_OPERATION_BODY	No	This is used to specify the request body for the invocation of POST_OPERATION. User can specify a file path or runtime template {odi.CHUNK_PATH}. If a file path (HDFS or Local) is specified, then that will be added as the request body when this operation is invoked. If runtime template {odi.CHUNK_PATH}is specified, then the following runtime templates also will also be assigned the values corresponding to last chunk file in addition to odi.CHUNK_PATH which gets the path of the last chunk. • odi.CHUNK_NAME — denotes name of the last chunk file • odi.CHUNK_SIZE — denotes size of last chunk file • odi.CHUNK_INDEX — 0 (Index of last chunk) If REQUEST_BODY_PART_VALUE parameter is specified, then this chunk is uploaded as a multipart body part. Otherwise this chunk file is set in the request body. If this parameter is not specified, then this POST_OPERATION will be invoked without a request body.
HEADER	No	The http headers to be used while invoking the REST service. Multiple headers can be passed like this during the tool invocation as given below : -HEADER.Content-Type=application/text -HEADER.Authorization=DAD9AFFA34D5==
HEADER_IGNORE_DEFAULT=YES/NO	No	This is applicable only when invoking the main operation specified by OPERATION parameter. It is a flag that indicates, if the headers defined in the physical schema are to be ignored or not. If "Yes" is selected, the headers defined in the physical schema operation are ignored while creating the invocation requests from main operation. If "No" is selected, then headers defined in the physical schema operation are merged with the headers specified using HEADER parameter in the tool. In case of conflict HEADER parameters get preference over the parameters defined in physical schema. Default value is "**NO**".
REQUEST_BODY_PART_NAME	No	This is used to specify the name of multipart content. This parameter is specified along with - REQUEST_BODY_PART_VALUE and – REQUEST_BODY_PART_CONTENT_TYPE parameters. When multipart fields are provided and if user is not explicitly specifying any multipart content type in the header, then the parameter Header.ContentType will be set to "multipart-mixed" by default. For multipart requests - REQUEST_BODY_FILE and <RequestBody> will be ignored.

Parameter	Mandatory	Description
REQUEST_BODY_PART_CONTENT_TYPE	No	This can be used to specify the content type of the multipart content. This can be repeated for each content in the body part. A sample invocation is as below: `OdiInvokeRESTfulService "-CONTEXT=GLOBAL" "-LSCHEMA=LSchema" "-OPERATION=PostOperationWithMultipartBody""-HEADER.Content-Type=multipart/mixed" "-REQUEST_BODY_PART_NAME=empDataAsXml""-REQUEST_BODY_PART_NAME=empDataAsJson" "-REQUEST_BODY_PART_VALUE=/path/employee.xml" "-REQUEST_BODY_PART_VALUE=/path/employee.json" "-REQUEST_BODY_PART_CONTENT_TYPE=application/xml" "-REQUEST_BODY_PART_CONTENT_TYPE=application/json"` Here multipart body has two body parts - an XML and a JSON
REQUEST_BODY_PART_VALUE	No	The presence of this parameter will be considered as the presence of multipart content in the request. This parameter will be used to specify the value of body part. It can be either a file path or a value. The value will be treated as text if the corresponding -REQUEST_BODY_PART_CONTENT_TYPE = "none", and will be treated as file path for any other body part content type value. User can also specify a runtime template {odi.CHUNK_PATH}.
REQUEST_QUERY	No	This parameter is used for passing query parameters which will be attached to the REST URL. **Note:** Query parameter values are in plain text format (i.e. not URI encoded). For Example : • -REQUEST_QUERY.userId={userId} • -REQUEST_QUERY.appId=1324 • -REQUEST_QUERY.loginKey={loginId} These parameters will be ignored if -ENCODED_REQUEST_QUERY_STRING is specified.

Parameter	Mandatory	Description
ENCODED_REQUEST_QUERY_STRING	No	This specifies the alternate way of specifying request query parameters. Repeated -REQUEST_QUERY parameters can be avoided by using this parameter. The value is expected to be encoded. It can be used along with -REQUEST_TEMPLATE which supplies values. userId={userId}&appId=1234 If this parameter is specified, all -REQUEST_QUERY parameters will be ignored.
REQUEST_QUERY_IGNORE_DEFAULT	No	This is only applicable when initializing the main operation specified by OPERATION parameter. It is a flag that indicates if the request queries defined in the physical schema are to be ignored or not. If "Yes" is selected, the request queries defined in the physical schema operation are ignored while creating the invocation requests from main operation. If "No" is selected, then request queries defined in the physical schema operation are merged with the request queries specified using REQUEST_QUERY parameter in the tool. In case of conflict REQUEST_QUERY parameters get preference over the parameters defined in physical schema. Default value is "**NO**".
REQUEST_TEMPLATE	No	These are regular templates provided by user. This will be used for substituting template variables (enclosed in curly braces) in the REST resource path, header parameter and query parameters (specified through -REQUEST_QUERY |-ENCODED_REQUEST_QUERY_STRING) For Example: • -REQUEST_TEMPLATE.userId=#GLOBAL.USER_ID • -REQUEST_TEMPLATE.loginId=test • -REQUEST_TEMPLATE.pwd=#GLOBAL.MY_REST_PWD
REQUEST_TEMPLATE_IGNORE_DEFAULT	No	This is only applicable when invoking the main operation specified by OPERATION parameter. It is a flag that indicates, if the request templates defined in the physical schema are to be ignored or not. If “Yes” is selected, the request templates defined in the physical schema operation are ignored while creating the invocation requests from main operation. If “No” is selected, then request templates defined in the physical schema operation are merged with the request queries specified using REQUEST_TEMPLATE parameter in the tool. In case of conflict REQUEST_TEMPLATE parameters get preference over the parameters defined in physical schema. Default value is "**NO**".
RESPONSE_FILE	No	This parameter specifies the name of the file to which the REST call response will be written into. If this parameter is not specified, then the response generated by REST call will be discarded, if any.
RESPONSE_HDFS_LSCHEMA	No	This parameter denotes the HDFS file configuration for the response file. This is applicable when response file is to be created in HDFS.

Parameter	Mandatory	Description
APPEND_RESPONSE	No	This is ignored if RESPONSE_FILE parameter is not specified. If "NO" is specified, a new file is always created to store response. If "Yes" is specified, then response is appended to the response file if it already exists. If it doesn't exist, a new response file is created. Default value is "**NO**".
RESPONSE_FILE_CHARSET	No	This parameter is applicable only if the REST service is expected to give a response which is character data. When this character data is written into a response file, this encoding will be used. If the REST response is expected to be a binary file, then this parameter must not be specified.
TIMEOUT	No	This parameter indicates the specified amount of waiting time (in milliseconds) for which the REST service invocation waits for, before considering that the server will not provide a response and an error is produced. If no value is given, it indicates infinite waiting time and there will be no time-out during invocation.
RETRY_COUNT	No	This parameter represents the number of times REST tool should attempt retry to access a REST service which is failing due to network related issues like time-out. Default value is **3**.
RETRY_INTERVAL	No	This parameter specifies the time interval (in milliseconds) after which retry should be attempted. Default value is **10000**.
FAILURE_STATUS_CODES	No	This parameter specifies the comma separated values of the status codes which indicate failure. When REST invocation gives any of these status codes, the tool execution will be stopped and tool will raise an error. For Example — "-FAILURE_STATUS_CODE=403,404"
TRACE_FILE	No	All available response status and response header information will be written into this file for debugging purpose.
APPEND_TRACE	No	This parameter is ignored, if TRACE_FILE parameter is not specified. If "NO" is specified, a new file is always created to store trace information. If "YES" is specified, then trace information is appended to a trace file, if it already exists. If it doesn't exist, a new file is created. Default value is "**NO**".

Parameter	Mandatory	Description
NEXT_REQUEST_RESOLVER	No	This can contain an Xpath expression, a JSONPath expression, a Header Name, a Link header relation or a positive numeric expression (which indicates page size). If the value is an Xpath expression or a JSONPath expression, it is applied in the response data to retrieve a value which is either an HTTP URL or a string value. If this resolved value is a complete REST URL, it will be set as the value for the runtime template odi.OPERATION_URL_OVERRIDE. If the resolved value is not a URL, then it is set as the value for the runtime template odi.PAGE_TOKEN (for Pagination invocation) or odi.UPLOAD_ID (for Chunk upload invocation). For Example — JsonPath : - $.account.container[0].name Xpath - /account/container[1]/name/text() If the value is numeric (page size), then the tool will use it to increase the page offset. The starting value for page offset is 0. New page offset value will be set to the runtime template odi.PAGE_TOKEN. User can define the operation to contain the required query parameter that contains this runtime template in the value. If the value is string, it will be treated as a response header key or a Link header relation. First the response header will be checked for the header key. If header key is not found in the response headers, it will be considered as Link header. Link header in the REST response will be parsed and using the relation value the next page link will be identified. Possible values for Link header relation are next, next page etc. Tool will not support if the Link header relation is numeric. Once the next page URL is identified, it will be set to runtime template odi.OPERATION_URL_OVERRIDE. For Example — GitHub REST API provides next page URL information in the Link header https://developer.github.com/guides/traversing-with-pagination/ If the parameter NEXT_REQUEST_RESOLVER contains an Xpath expression, a JSONPath expression or Link header relation which cannot be resolved, or is resolved to a value representing null or empty (0,"","" null" etc) , then it will be treated as end of results (last page). **Note:** Once the runtime template odi.OPERATION_URL_OVERRIDE is set, if any operation is initialized, then this value will be used to override the resource path defined in the operation. For example: - Suppose after the PRE_OPERATION invocation, this runtime template is set, then

Parameter	Mandatory	Description
		OPERATION and POST_OPERATION will use this value as the resource path. The resource path defined in these operations will be ignored.
TOTAL_COUNT_FIELD_RESOLVER	No	This parameter is used by pagination type invocation. If the total count of records is present in the paginated response, user needs to specify an Xpath expression or a JsonPath expression to extract it from the REST response. This total count value will be read from each paginated response and will be checked to see if it is less than newly calculated value of page offset. If provided, this will be used to identify the last page (Termination condition). For more information, see Handling Paged Responses section in LinkedIn REST API example.

Parameter	Mandatory	Description

RESOLVER_OVERWRITE_CLASS	No	If the NEXT_REQUEST_RESOLVER parameter is not sufficient to identify the next page incase of pagination type invocation and Upload URL incase of Chunk Upload type invocation, then user can provide their own implementation. You have to specify the fully qualified name of the class that implements oracle.odi.runtime.rest.INextRequestResolver interface. In that class, you have to override process() . This class must be available in ODI class path.
		
		Note:
		These publicly visible classes are available in SDK library.
		
		`public interface oracle.odi.runtime.rest.INextRequestResolver {`
		
		`public boolean process(oracle.odi.runtime.rest.INextRequestResolver restToolProvider ,`
		
		`oracle.odi.runtime.rest.IOdiToolRestRequest request,`
		`java.io.InputStream response /*REST response input stream */,`
		`javax.ws.rs.core.MultivaluedMap<String,Object>`
		
		`responseHeaders /*REST response headers*/) throws Exception;`
		`}`
		
		1. **restToolProvider** – Provides access to tool parameters. For Example — `String tokenParamName= restToolProvider.getToolParameterValue("-PAGE_TOKEN_PARAM_NAME", String.class);`. It also has the following methods:
		• `getRuntimeTemplates()` – Gives access to runtime templates.
		• `getChunkFiles()` – Gives access to chunk files created or identified by the tool
		2. **request** — It contains the details of the last REST request made by the tool. The details provided are `requestPath, requestQueries and requestHeaders and httpMethod`. User can modify these details and return true from process (), if more pages exist in a pagination invocation. This modified request object will be used to make the very next REST call, if it is of the same operation.

Parameter	Mandatory	Description
		3. **response**— The response stream obtained as the result of last invocation. User can process this to obtain any details which is required to construct the next request 4. **responseHeaders**— The response headers obtained from last REST request. User can use this to construct the next request This implementation should handle the termination condition for Pagination type invocation. This means for Pagination type invocation when the last page is reached, process () method should return false. For Pagination type invocation, REST tool will be continuously running until false is returned from process(). `OdiToolRestRequest interface parameter` has the following methods and you can access and modify these values: • `String getRequestPath()` — Get request path of last request • `void setRequestPath(String requestPath)` — Modify the request path for next invocation • `MultivaluedMap<String,Object> getRequestQueries()`— Get request queries used in last invocation • `void setRequestQueries(Multivalued<MapString>, Object requestQueries)`- Set request queries for next invocation • `MultivaluedMap<String,Object> getRequestHeaders()` — Get request headers of last invocation • `setRequestHeaders(MultivaluedMap<String,Object> requestHeaders()`— Set request headers for next invocation • `getMethod()` — Get http method used in last invocation • `setMethod(String method)` — Set HTTP method for next invocation You can use these setter methods to modify the behavior of next request only if it is of the same operation. For Example — If process() is called, as a result of the invocation of `PRE_OPERATION` , then any changes made to this request object wont' be useful because the next invocation will be of a different operation specified by parameter `OPERATION`. New request object will be invoked from this operation. But you have the option to override the resource path across all upcoming operations using runtime template `odi.OPERATION_URL_OVERRIDE`. If you set this runtime template in the process() method invocation after `PRE_OPERATION`, then the operations - `OPERATION` and `POST_OPERATION` will take this as the request path and ignore the request path defined in those operations.

Parameter	Mandatory	Description
RESPONSE_DATA_CONTAINER	No	The value will be an Xpath (for XML response) or JSONPath (for JSON response) to extract the actual data from response. The resolved value of this field in all REST call responses will have to be accumulated to get the full response. If the response is plain text type, then whole response will be taken. If the value is an Xpath expression "/account/", then "account " will be the root element in the final accumulated xml response. Paginated JSON response data will always be in array format. The arrays of each paginated response data can be merged to get the final response which will be an array again.
<RequestBody>	No	Body of the REST request. If REQUEST_BODY_FILE value is specified, then this field is not used. This message should be provided on the line immediately following the OdiInvokeRESTfulService call (on a separate line or with the -REQUEST_BODY parameter].

Usage Recommendations

For usage recommendations of odiInvokeRESTfulService tool, see Usage Recommendations.

Examples

Listed below are the examples for OdiInvokeRESTfulService for Pagination and Chunk Upload functions:

- Examples for Pagination
- Examples for Chunk Upload

Usage Recommendations for odiInvokeRESTfulService tool

General usage recommendations for odiInvokeRESTfulService tool are:

1. The tool helps to identify the invocation type based on the parameters provided. There are three types of invocations – Regular, Chunk Upload and Pagination. If CHUNK_SIZE or CHUNK_PREFIX parameters are present then it is Chunk Upload type invocation. If these parameters are absent and NEXT_REQUEST_RESOLVER or RESOLVER_OVERWRITE_CLASS parameter is present, then the invocation is of Pagination type. All other invocations are considered as regular invocations and it is recommended to use OPERATION parameter only for specifying REST operation.
2. You may pass the parameter indicating the page size (maxResults,count etc) as query parameter for optimized retrieval. If not, the REST service will return the pages with default size. In both the cases these tool parameters can be used to make repetitive calls to get the complete response. If you are specifying the page size, then it should not exceed the maximum limit imposed by the REST service.
3. XPath equivalent of JSON is JsonPath. For more details, see https://github.com/jayway/JsonPath. The link http://jsonpath.herokuapp.com/ can be used to evaluate JSONPath expression. REST tool uses Jayway implementation.
4. The tool will relay on the resolver expressions (provided in NEXT_REQUEST_RESOLVER, TOTAL_COUNT_RESOLVER or RESPONSE_DATA_CONTAINER_RESOLVER) to determine if the

response content is JSON or XML. For example, suppose the expected response is a JSON, you are not expected to provide Xpath resolver expressions.

5. If an Xpath expression or JSONPath expression is given to the parameter `NEXT_REQUEST_RESOLVER`, then these expressions should be resolved into a string value (indicates token, upload id or a URL). A JSON array containing a single item is also considered as valid resolved value for JSONPath expression.

   - For **JSONPath**, a valid resolved value may be depicted as – "ABCDEF", 12345, ["ABCDEF"], [12345], `http://host:port/context/resourcepath/` etc.
   - For **Xpath expression**, a valid resolved value may be depicted as — ABCDEF, 12345, `http://host:port/context/resourcepath/`etc

6. You can use the runtime templates while constructing the URL path, query and headers. The values of these templates will be determined at the runtime. All runtime templates will start with **odi** prefix in the template name. This will distinguish runtime templates with regular user defined templates. Runtime templates can also be used as the values of the parameters – `PRE_OPERATION_BODY, POST_OPERATION_BODY and REQUEST_BODY_PART_VALUE.`

7. Following are the runtime templates that are identified by this tool. The values of runtime templates are set once or multiple times by the tool for all REST invocations. You can specify a value for a runtime template, as the initial value, in the operation definition in a REST physical schema, just like a regular user defined template. But initialization of runtime template using the tool parameter `REQUEST_TEMPLATE`, is not allowed.

   - `odi.CHUNK_PREFIX`— This template represents the value of `CHUNK_PREFIX` parameter and this value is set once chunk files are created.
   - `odi.UPLOAD_ID` — This runtime template will contain the upload session id once this is resolved from the first invocation response (using `NEXT_REQUEST_RESOLVER`). This is set in Chunk Upload invocation and is set only once.

     You can provide you own implementation using `RESOLVER_OVERWRITE_CLASS` parameter.
   - `odi.PAGE_TOKEN` — This runtime template will contain the page token or page offset for next page request. This is used for Pagination invocation and a new value will be set after each page response and is processed using `NEXT_REQUEST_RESOLVER`.

     You can provide your own implementation using `RESOLVER_OVERWRITE_CLASS` parameter.
   - `odi.OPERATION_URL_OVERRIDE` – If `NEXT_REQUEST_RESOLVER` is returning a URL, then that URL will be set to this runtime template. For Chunk upload type this is set only once. For Pagination type, this is set after each page response is processed. You can provide your own implementation using `RESOLVER_OVERWRITE_CLASS` parameter.

   a. For chunk upload type invocation, once the chunks files are identified or created, the following template is set:

      `odi.TOTAL_CHUNK_SIZE` — It is the total size of all chunk files combined.

   b. The following runtime templates are set when a particular chunk file is being chosen for upload:

- `odi.CHUNK_PATH` — Absolute path of chunk file in the temp directory
- `odi.CHUNK_NAME` — Name of the chunk
- `odi.CHUNK_SIZE` — Size of the chunk. Mostly this will be same for all chunks. But for last chunk it may differ.
- `odi.CHUNK_INDEX` — Index of the chunk being used starting from 0. If 1st index is being used, this will represent a value 0.

Examples for Pagination

The following are the examples for Pagination function of OdiInvokeRESTfulService tool.

Twitter Followers API

Reference Link : https://dev.twitter.com/overview/api/cursoring

Let's consider the following Twitter API for details. This API lists the id of the followers:
`https://api.twitter.com/1.1/followers/ids.json?screen_name=<your-screen-name>`

Let's suppose the default page size of this API is 15. The response of first invocation will be as below:

```
{
  "ids": [
    2552855054,
    4345418177,
    3803100858,
    56422577,
    3326965752,
    3075258528,
    3302261082,
    297834835,
    2927402418,
    56053134,
    78849029,
    70703605,
    2850513554,
    161289980,
    548960923
  ],
  "next_cursor": 1434098452051477000,
  "next_cursor_str": "1434098452051476935",
  "previous_cursor": 0,
  "previous_cursor_str": "0"
}
```

Define an operation "getFollowers" with URL: `https://api.twitter.com/1.1/followers/ids.json?screen_name=<your-screen-name>`

In the second operation give a query parameter cursor=`{odi.PAGE_TOKEN}`

REST tool Parameters

```
PRE_OPERATION=getFollowers
OPERATION=getFollowers
```

```
REQUEST_QUERY.cursor={odi.PAGE_TOKEN} // This will be added to main
operation only
NEXT_REQUEST_RESOLVER=$.next_cursor_str
RESPONSE_DATA_CONTAINER_RESOLVER=$.ids
```

The JSONPath expression `$.next_cursor_str` will be resolved in to a value "1434098452051476935". This will be used as the value to the parameter cursor in the second REST invocation.

```
https://api.twitter.com/1.1/followers/ids.json?screen_name=<your-screen-
name>&cursor=1434098452051476935
```

```
{
  "ids": [
    548960923,
    435520948,
    338402626,
    80845228
  ],
  "next_cursor": 0,
  "next_cursor_str": "0",
  "previous_cursor": -1434098452051477000,
  "previous_cursor_str": "-1434098452051476935"
}
```

In the response, the value of next cursor is 0. It indicates there are no more pages left and the tool can stop execution after writing the response. The parameter `RESPONSE_DATA_CONTAINER_RESOLVER` will be used to fetch the required data from the response. The final response will be as follows:

```
[
    2552855054,
    4345418177,
    3803100858,
    56422577,
    3326965752,
    3075258528,
    3302261082,
    297834835,
    2927402418,
    56053134,
    78849029,
    70703605,
    2850513554,
    161289980,
    548960923,
    548960923,
    435520948,
    338402626,
    80845228
  ]
```

Google Drive API

This is similar to Twitter REST API pagination which uses the page pointers in the response to get the value for page token query parameter for next request.

Let's consider the following API URL: `https://developers.google.com/apis-explorer/#s/drive/v2/drive.files.list?_h=1`

Sample Response

```
{
"kind": "drive#fileList",
 "etag": "\"rCKCAyesbPCaBxGt0eDJcEBQNUI/HNdpkEyt-3gaIlW8i4TRzGJXk-w\"",
 "selfLink": "https://www.googleapis.com/drive/v2/files?maxResults=3",
"nextPageToken": "EAIaqgELEgBSoQEKjwEKaPjz",
 "nextLink": "https://www.googleapis.com/drive/v2/files?
maxResults=3&pageToken=EAIaqgELEgBSoQEKjwEKaPjz",
 "items": [
  ...
 ]
}
```

Define an operation "getFiles" with URL: `https://developers.google.com/apis-explorer/#s/drive/v2/drive.files.list?_h=1`

REST tool Parameters

```
PRE_OPERATION=getFiles
OPERATION=getFiles
REQUEST_QUERY.pageToken={odi.PAGE_TOKEN} // This will be added to main
operation only
NEXT_REQUEST_RESOLVER=$.nextPageToken
RESPONSE_DATA_CONTAINER_RESOLVER=$.items
```

JSONPath expression `$.nextPageToken` will be resolved to a value `EAIaqgELEgBSoQEKjwEKaPjz`

Using the above parameters the REST tool will construct the URL for second invocation as below:

```
https://developers.google.com/apis-explorer/#s/drive/v2/drive.files.list?
_h=1&pageToken=EAIaqgELEgBSoQEKjwEKaPjz
```

Since Google drive API also provides nextLink in the response which is the complete URL pointing to next page, there is an alternate way of accumulating the paginated result.

Set `NEXT_REQUEST_RESOLVER` as below:

REST tool Parameters

```
PRE_OPERATION=getFiles
OPERATION=getFiles
REQUEST_QUERY.pageToken={odi.PAGE_TOKEN} // This will be added to main
operation only
```

```
NEXT_REQUEST_RESOLVER=$.nextLink
RESPONSE_DATA_CONTAINER_RESOLVER=$.items
```

Salesforce REST API (Chatter REST API)

This is also similar to Twitter REST API pagination except that the next page pointer in the REST response will be only an actual URL. But in Google drive API and Twitter API, the next page pointer was a token which would be passed as the value of the query parameter <PageTokenParam> in the second request and so on.

Reference Link: `http://help.salesforce.com/HTViewSolution?id=000175552&language=en_US`

Salesforce Chatter API URL suffix will similar to — `/services/data/v25.0/chatter/feeds/news/me/feed-items?pageSize=25`

Here `pageSize` in the query parameter indicating page size. Providing `pageSize`is optional. The response will contain a field `nextPageUrl`. The value of the will be the complete REST URL to get the next page of the result.

LinkedIn Get Company Updates (Offset Based Pagination)

Pagination support in Linked REST APIs is different when compared to above mentioned REST APIs.

Reference Link : `https://developer.linkedin.com/docs/rest-api`

Let's consider the following REST API — `https://api.linkedin.com/v1/companies/1337/updates`

Define an operation `getCompanyUpdates` for the above URL.

REST tool Parameters

```
PRE_OPERATION=getCompanyUpdates
OPERATION=getCompanyUpdates
NEXT_REQUEST_RESOLVER=10 //Page size. Specified in the below    URL
using count parameter
RESPONSE_DATA_CONTAINER_RESOLVER=$.values
TOTAL_COUNT_FIELD_RESOLVER=$._total
REQUEST_QUERY.start={odi.PAGE_TOKEN} // Will be used by main operation
only
```

The first REST invocation will happen with below URL provided user is adding the query parameter start into the query string either using `REQUEST_QUERY` or `ENCODED_REQUEST_QUERY_STRING`. Since first invocation is of `PRE_OPERATION`, it will not consider `REQUEST_QUERY` parameter specified in the tool: `https://api.linkedin.com/v1/companies/1337/updates?start=0&count=10&format=json`

1. Response for first invocation

```
{
  "_count": 10,
  "_start": 0,
  "_total": 26,
  "values":  [
```

```
        …
  ]
}
```

When each response is processed, tool will increment the value of `odi.PAGE_TOKEN` runtime templates by the numeric value specified in `NEXT_REQUEST_RESOLVER` starting from 0 and the new value for start is 0+10=10. This is less than total record count =26 (extracted using JSONPath expression specified in `TOTAL_RECORDS`). So there is a next page.

The second REST invocation URL will be — `https://api.linkedin.com/v1/companies/1337/updates?start=10&count=10&format=json`

2. Response for second invocation

```
{
  "_count": 10,
  "_start": 10,
  "_total": 28,
  "values":  [
      …
  ]
}
```

New value for start = 10+10=20. This is less than total record count 28 (Earlier it was 26. Two more records are added in the REST service end). So there is a next page.

The third REST invocation URL will be — `https://api.linkedin.com/v1/companies/1337/updates?start=20&count=10&format=json`

3. Response for third invocation

```
{
  "_count": 10,
  "_start": 20,
  "_total": 24,
  "values":  [
      …
  ]
}
```

New value for start is 20+10=30. This is greater than total record count 24 (It was 28 in last request, but changed to 24 because 4 records are deleted from REST service end in the mean time). So this is the last page.

Total records value will be extracted from each response and will be used to determine the last page.

Note:

1. If you are not specifying the page size using count query parameter, the REST service will return pages with default page size. In such case, you are expected to provide this default page size as the value for `NEXT_REQUEST_RESOLVER` parameter because the tool will not be able to identify the default page size for a service.
2. The tool solves this pagination using an internal resolver. For this resolver to be used you have to provide values for `NEXT_REQUEST_RESOLVER`and `TOTAL_RECORDS` and the value given for `NEXT_REQUEST_RESOLVER` should be numeric (it indicates page size).

Writing own Implementations using `RESOLVER_OVERWRITE_CLASS` REST tool Parameters

```
RESOLVER_OVERWRITE_CLASS=myCom.LinkedInPaginationResolver

public class OffsetBasedPaginationResolver implements
INextRequestResolver {
    private int mStartIndex = 0; // Default value of the start index
    public OffsetBasedPaginationResolver(){
    }
    public OffsetBasedPaginationResolver(int startIndex){
        mStartIndex = startIndex;
    }
    @Override
    public boolean process(IRestToolProvider restToolProvider,
            IOdiToolRestRequest request,
            InputStream responseData, MultivaluedMap<String,Object>
responseHeaders)
            throws Exception {
        int pageSize =
Integer.parseInt(restToolProvider.getToolParameterValue(IRestToolProvid
er.PARAM_NEXT_REQUEST_RESOLVER).toString());
        String totalCountResolverExpression =
restToolProvider.getToolParameterValue(IRestToolProvider.PARAM_TOTAL_CO
UNT_FIELD_RESOLVER).toString();
        long totalRecords =
InvokeRESTfulServiceSupport.getPathLongValue(responseData,
totalCountResolverExpression);
        mStartIndex+= pageSize;
        if(mStartIndex <= totalRecords){

restToolProvider.getRuntimeTemplates().put(IRestToolProvider.RUNTIME_TE
MPLATE_PAGE_TOKEN, mStartIndex);
            return true;
        }
        return false;
    }
}
```

Oracle Storage Cloud Service API – List Containers

Pagination support in Oracle Storage Cloud Service REST APIs is different when compared to above mentioned REST APIs, considering how the next page token is extracted from the response.

Reference Link : `http://docs.oracle.com/cloud/latest/storagecs_common/SSAPI/op-v1-%7Baccount%7D-get.html#request`

Let's consider the following REST API. Here the maximum limit is MAX_LIMIT=10000 (This is both default and maximum value). Let's assume the default page size or MAX_LIMIT is 5.

First invocation URL is — `https://<your_domain>.storage.oraclecloud.com/v1/<your_account>?format=json`

Define an operation `listContainers`with above URL:

Rest tool Parameters

```
PRE_OPERATION=listContainers
OPERATION=listContainers
REQUEST_QUERY.marker={odi.PAGE_TOKEN}
NEXT_REQUEST_RESOLVER =$[-1:].name   //This will fetch the name field from
last record. To fetch the name field from first record use JSONPath
expression "$[:1]"
```

Sample response for first invocation

```
[{"name":"container-10","count":0,"bytes":0,"accountId":
{"id":15935},"deleteTimestamp":0.0,"containerId":{"id":223537}},
{"name":"container1","count":0,"bytes":0,"accountId":
{"id":15935},"deleteTimestamp":0.0,"containerId":{"id":223427}},
{"name":"container10","count":0,"bytes":0,"accountId":
{"id":15935},"deleteTimestamp":1.45984151022644E9,"containerId":
{"id":223433}},
{"name":"container2","count":0,"bytes":0,"accountId":
{"id":15935},"deleteTimestamp":0.0,"containerId":{"id":223379}},
{"name":"container3","count":0,"bytes":0,"accountId":
{"id":15935},"deleteTimestamp":0.0,"containerId":{"id":223380}}]
```

Next page resolver will be resolved to a JSON array.

```
["container3"]
```

REST tool will extract the value for `container3`from this JSON array.

New REST URL for second page will be — `https://<your_domain>.storage.oraclecloud.com/v1/<your_account>?marker=container3`

Resolved value for JSONPath expression `$..name`on the response data

```
[
   "container-10",
   "container1",
   "container10",
```

```
"container2",
"container3"]
```

Now let's consider the scenario where response format is XML — https://<your_domain>.storage.oraclecloud.com/v1/<your_account>?format=xml

Define an operation `listContainersXml` with above URL.

Rest tool Parameters

```
PRE_OPERATION=listContainersXml
OPERATION=listContainersXml
REQUEST_QUERY.marker={odi.PAGE_TOKEN}
NEXT_REQUEST_RESOLVER=/account/container[last()]/name/text()  //This
will fetch the name field from last record. To fetch the name field
from first record use Xpath expression "/account/container[1]/name/
text()"
RESPONSE_DATA_CONTAINER_RESOLVER=/account   //This will be root
element in the accumulated response
```

Sample response of first invocation

```
<?xml version="1.0" encoding="UTF-8"?>
<account><container><accountId><id>15935</id></accountId><bytes>0</
bytes><containerId><id>223537</id></containerId><count>0</
count><deleteTimestamp>0.0</deleteTimestamp><name>container-10</name></
container>
<container><accountId><id>15935</id></accountId><bytes>0</
bytes><containerId><id>223427</id></containerId><count>0</
count><deleteTimestamp>0.0</deleteTimestamp><name>container1</name></
container>
<container><accountId><id>15935</id></accountId><bytes>0</
bytes><containerId><id>223433</id></containerId><count>0</
count><deleteTimestamp>1.45984154E9</
deleteTimestamp><name>container10</name></container>
<container><accountId><id>15935</id></accountId><bytes>0</
bytes><containerId><id>223379</id></containerId><count>0</
count><deleteTimestamp>0.0</deleteTimestamp><name>container2</name></
container>
<container><accountId><id>15935</id></accountId><bytes>0</
bytes><containerId><id>223380</id></containerId><count>0</
count><deleteTimestamp>0.0</deleteTimestamp><name>container3</name></
container></account>
```

Next page resolver will be resolved to a value `container3`, which will be used as the next value of `marker` query parameter.

Oracle Storage Cloud Service API – Get Object Content (Download object)

This API supports retrieval of objects stored in the containers. Object can be retrieved in chunks using Range header. This API does not have a max limit for the number of bytes that can be downloaded. But using the pagination support in the REST tool, you can download this object as chunks.

If you want to download an object named `mydata` which is stored in container `container1`, the first invocation URL is: `https://<your_domain>.storage.oraclecloud.com/v1/<your_account>/container1/mydata`

You need to set the initial Range header using tool parameter HEADER.

Rest tool Parameters

```
HEADER.Range=bytes=1-99
RESOLVER_OVERWRITE_CLASS=com.StorageCSPaginationResolver
```

You need to provide implementation to parse the response header Content-Range to get the current range and the total byte size. A sample value will be “bytes 1-99/2677”. 1-99 is the range and 2677 is the total bytes available. Response of process() in `StorageCSPaginationResolver` should be a string value "bytes=100-199". This value will be passed as the new Range header value in the second REST call. You should also handle termination condition in their implementation. When a new range is calculated, if the lower range is less than the total bytes value, it indicates that no more pages are left. The process() should return with a false value, as the method process()'s return type is a boolean.

GitHub API

This API is an example for Link header based pagination support.

First REST invocation will happen with the URL — `https://api.github.com/search/code?q=addClass+user:mozilla&page=1`

Rest tool Parameters

```
NEXT_REQUEST_RESOLVER=next // It is a link header relation
RESPONSE_DATA_CONTAINER_RESOLVER=$.items
```

The response header "Link "will contain the following value —

```
https://api.github.com/search/code?q=addClass+user%3Amozilla&page=2;
rel="next",
https://api.github.com/search/code?q=addClass+user%3Amozilla&page=34;
rel="last"
```

The REST tool will parse this link header and extract the link that corresponds to the relation "next".

The URL for next invocation will be — `https://api.github.com/search/code?q=addClass+user%3Amozilla&page=2`

This will keep on going until the link header value in the response does not has a relation represented by the relation " next".

This can also be achieved by you own implementation of resolver class as described below:

Rest tool Parameters

```
NEXT_REQUEST_RESOLVER=next
RESOLVER_OVERWRITE_CLASS=com.GitHubPaginationResolver
RESPONSE_DATA_CONTAINER_RESOLVER=$.items
```

Twitter timeline API

> **Note:**
>
> Pagination support in Twitter Tweets API is the same as the above Oracle Storage Cloud Service API .

Consider the REST service initial URL — `https://api.twitter.com/1.1/search/tweets.json?q=<screen_name>`

Define an operation "getTweets "using above URL.

REST tool Parameters

```
PRE_OPERATION=getTweets
OPERATION=getTweets
REQUEST_QUERY.since_id={odi.PAGE_TOKEN}
NEXT_REQUEST_RESOLVER=$[:1].id_str   //This will give the id_str from
the first record
RESPONSE_DATA_CONTAINER_RESOLVER=$.statuses
```

Second invocation URL will be — `https://api.twitter.com/1.1/search/tweets.json?q=<screen_name>&since_id=<resolved_next_page_value>`

Examples for Chunk Upload

Twitter Media Upload

Reference link: `https://dev.twitter.com/rest/reference/post/media/upload-init`

It has three types of operations. They are:

1. POST media/upload (INIT) – This request will start an upload session

   Parameters:

   ```
   command=INIT
   total_bytes=total file size
   ```

2. POST media/upload (APPEND) - multipart/form-data format

   Parameters:

   ```
   command=APPEND
   media_id= the media_id returned from the INIT command
   media= The raw binary file content being uploaded. <=5MB // media
   is multipart body part field name.
   segment_index= An ordered index of file chunk.It must be between
   0-999 inclusive. The first segment has index 0, second segment has
   index 1, and so on.
   ```

3. POST media/upload (FINALIZE)

Parameters:

```
command=FINALIZE
media_id= the media_id returned from the INIT command
```

For Example — Let us assume you have to upload 1 GB (1073741824 bytes) media.

You need to create 2 rest operations in the physical schema. Define the below operations:

- Initialize(POST) = `https://upload.twitter.com/1.1/media/upload.json?command=INIT&total_bytes=1073741824`
- Upload (POST)=`https://upload.twitter.com/1.1/media/upload.json?command=APPEND&media_id={odi.UPLOAD_ID}&segment_index={odi.CHUNK_INDEX}`
- Finalize(POST)=`https://upload.twitter.com/1.1/media/upload.json?command=FINALIZE&media_id={odi.UPLOAD_ID}`

The response of first invocation can be:

```
{
  "media_id": 710511363345354753,
  "media_id_string": "710511363345354753",
  "size": 11065,
  "expires_after_secs": 86400,
}
```

REST tool Parameters

```
CONTEXT=<Context>

LSCHEMA=<Logical Schema>

INPUT_FILE=<input file path>

CHUNK_SIZE=5000000 (Nearly 5MB).This will break input file into 215 chunks

REQUEST_BODY_PART_NAME=media

REQUEST_BODY_PART_VALUE={odi.CHUNK_PATH}

REQUEST_BODY_PART_CONTENT_TYPE=application/json

PRE_OPERATION=Initialize

NEXT_REQUEST_RESOLVER=$.media_id

OPERATION=Upload    //This operation uploads all chunks.

POST_OPERATION=Finalize

TRACE_FILE=<Trace File Name>
```

Google Drive Upload Files API

Reference link : `https://developers.google.com/drive/v3/web/manage-uploads#resumable`

It has two types of operations. They are:

1. `POST files?uploadType=resumable (initial request)`

   Use the following HTTP headers with the initial request:

   a. `X-Upload-Content-Type`. Set to the media MIME type of the upload data to be transferred in subsequent requests.

   b. `X-Upload-Content-Length`. Set to the number of bytes of upload data to be transferred in subsequent requests. If the length is unknown at the time of this request, you can omit this header.

   c. If providing metadata: `Content-Type`. Set according to the metadata's data type.

   d. `Content-Length`. Set to the number of bytes provided in the body of this initial request. Not required if you are using chunk transfer encoding.

   The response will contain a response header location which contains the `session_uri`.

2. `PUT session_uri (uploading chunks)`

   Content-Length header must be set when a chunk is being uploaded. Suppose you need to upload 1 GB (1073741824 bytes) of media . You need to create 2 rest operations in the physical schema.

   Define below operations:

   a. InitialRequest (POST)= `https://www.googleapis.com/upload/drive/v3/files?uploadType=resumable`

      Set the necessary headers such as `X-Upload-Content-Type`, `X-Upload-Content-Length`and so on, while defining the operation in the physical schema.

   b. Upload (PUT)=`https://www.googleapis.com/upload/drive/v3/files?uploadType=resumable (This url will be ignored and the upload`

      URL obtained in the first operation response will be used. Set the `Content-Type header`. Set `Content-Length={odi.CHUNK_SIZE}`

      The response of first operation can be:

      ```
      HTTP/1.1 200 OK
      Location: https://www.googleapis.com/upload/drive/v3/files?
      uploadType=resumable&upload_id=xa298sd_sdlkj2
      Content-Length: 0
      ```

REST tool Parameters

```
CONTEXT=<Context>

LSCHEMA=<Logical Schema>
```

```
INPUT_FILE=<input file path>

CHUNK_SIZE=5000000 (Nearly 5MB).This will break input file into 215 chunks

PRE_OPERATION=InitialRequest

NEXT_REQUEST_RESOLVER=Location   (Location is expected to be a response
header)

OPERATION=Upload

TRACE_FILE=<Trace File Name>
```

COMMVAULT API (Upload a File in Chunks)

Reference link: `https://documentation.commvault.com/commvault/v10/article?p=features/rest_api/operations/post_contentstore_share_chunk_upload.htm`

It has two types of operations:

1. `POST upload?uploadType=chunkedFile (initial request)`

   Set all necessary headers in the operation. The response will be an element `DM2ContentIndexing_UploadFileResp` which contains the upload id. This will be the session id for upload requests.

2. `POST upload?uploadType=chunkedFile`

For Example — If you have to upload 1 GB (1073741824 bytes) media, you need to create three Rest operations in the physical schema.

Define the below operations:

- `InitUpload (POST)= SearchSvc/CVWebService.svc/contentstore/share/{shareId}/file/action/upload?uploadType=chunkedFile`

  Set the necessary headers such as `FileName`, `FileSize`, `ParentFolderPath` and so on while defining the operation in the physical schema.

- `Upload (POST)= SearchSvc/CVWebService.svc/contentstore/share/{shareId}/file/action/upload?uploadType=chunkedFile&requestId={odi.UPLOAD_ID}`

  Set all necessary headers for the upload. Ignore `FileEOF` header . This operation will be used to upload all chunks except the last one.

- `EndUpload (POST)= SearchSvc/CVWebService.svc/contentstore/share/{shareId}/file/action/upload?uploadType=chunkedFile&requestId={odi.UPLOAD_ID}`

  Set header, `FileEOF=1`

The response of first operation may look, as mentioned below. The field `requestId` contains the upload id.

```
<DM2ContentIndexing_UploadFileResp
requestId="132130220882341981601081252141832305861341 82" chunkOffset="780830"
errorCode="409" />
```

Rest tool Parameters

```
CONTEXT=<Context>
```

```
LSCHEMA=<Logical Schema>

INPUT_FILE=<input file path>

CHUNK_SIZE=5000000 (Nearly 5MB).This will break input file into 215
chunks

PRE_OPERATION=InitUpload

PRE_OPERATION_BODY={odi.CHUNK_PATH} //Sets the first chunk as body of
this operation

NEXT_REQUEST_RESOLVER=string(/DM2ContentIndexing_UploadFileResp/
@requestId)    //Xpath expression to get the requestId

OPERATION=Upload

POST_OPERATION=EndUpload

PRE_OPERATION_BODY={odi.CHUNK_PATH} //Sets the last chunk as body of
this operation

TRACE_FILE=<Trace File Name>
```

Oracle Storage Cloud Service

Reference link: `https://docs.oracle.com/cloud/latest/storagecs_common/CSSTO/GUID-CA3E7F7B-4B33-4C18-8CEB-652813D9ADFB.htm`

It has two types of operations:

1. `PUT accountURL/containerName/{odi.CHUNK_NAME} /`
2. `PUT accountURL/containerName/manifestFile`

For Example — If you need to upload 1 GB (1073741824 bytes) media. You need to create two Rest operations in the physical schema.

Define below operations

- `Upload (POST)=<accountURL>/{containerName}/{odi.CHUNK_NAME}`
- `UploadManifest (POST)=<accountURL>/{containerName}/manifestFile`

  This operation will be used to upload the 0 byte manifest object.

  Set the following headers for this operation :

  ```
  X-Object-Manifest= {containerName}/{odi.CHUNK_PREFIX}
  Content-Length=0
  ```

REST tool Parameters

```
CONTEXT=<Context>

LSCHEMA=<Logical Schema>
```

```
INPUT_FILE=<input file path>

CHUNK_SIZE=5000000 (Nearly 5MB).This will break input file into 215 chunks

OPERATION=Upload

END_OPERATION=UploadManifest

TRACE_FILE=<Trace File Name>
```

OdiInvokeWebService

> **Note:**
>
> This tool replaces the OdiExecuteWebService tool.

Use this command to invoke a web service over HTTP/HTTPS and write the response to an XML file.

This tool invokes a specific *operation* on a *port* of a web service whose description file (WSDL) *URL* is provided.

If the `LOGICAL_SCHEMA` parameter is specified, this tool will use the configuration from topology objects. The syntax for the existing mode will be supported for backward compatibility.

If this operation requires a web service request, it is provided either in a request file, or directly written out in the tool call (`<XML Request>`). This request file can have two different formats (`XML`, which corresponds to the XML body only, or `SOAP`, which corresponds to the full-formed SOAP envelope including a SOAP header and body) specified in the `-RESPONSE_FILE_FORMAT` parameter. The response of the web service request is written to an XML file that can be processed afterwards in Oracle Data Integrator. If the web service operation is one-way and does not return any response, no response file is generated.

> **Note:**
>
> This tool cannot be executed in a command line with `startcmd`.

Usage

Syntax for new topology

```
OdiInvokeWebService -LOGICAL_SCHEMA=<WS Logical Schema> -OPERATION=<operation> -
CONTEXT=<ODI Context> (Optional)
[<XML Request>][-REQUEST_FILE=<xml_request_file>] [-RESPONSE_MODE=<NO_FILE|NEW_FILE|
FILE_APPEND>]
[-RESPONSE_FILE=<xml_response_file>] [-RESPONSE_XML_ENCODING=<charset>]
[-RESPONSE_FILE_CHARSET=<charset>]
[-RESPONSE_FILE_FORMAT=<XML|SOAP>] [-TIMEOUT=<timeout>]
```

Syntax for existing mode

```
OdiInvokeWebService -URL=<url> -PORT=<port> -OPERATION=<operation>
[<XML Request>] [-REQUEST_FILE=<xml_request_file>]
[-RESPONSE_MODE=<NO_FILE|NEW_FILE|FILE_APPEND>]
[-RESPONSE_FILE=<xml_response_file>] [-RESPONSE_XML_ENCODING=<charset>]
[-RESPONSE_FILE_CHARSET=<charset>] [-RESPONSE_FILE_FORMAT=<XML|SOAP>]
[-HTTP_USER=<user>]
[-HTTP_PASS=<password>] [-TIMEOUT=<timeout>]
```

Parameters

Parameters	Mandatory	Description
-LSCHEMA=<logical_schema>	No	Logical schema containing the journalized tables (optional parameter). If LSCHEMA is specified, then OdiInvokeWebService will use URL, PORT, HTTP_USER, and HTTP_PASS configured at mapped SOAP WS Physical Schema and/or SOAP WS Data Server.
-CONTEXT=<Odi context>	No	Context in which the logical schema will be resolved. If no context is specified, the execution context is used (optional parameter).
-URL=<url>	No	URL of the Web Service Description File (WSDL) describing the web service.
-PORT_TYPE=<port_type>	No	Name of the WSDL port type to invoke.
-OPERATION=<operation>	Yes	Name of the web service operation to invoke.
<XML Request>	No	Request message in SOAP (Simple Object Access Protocol) format. This message should be provided on the line immediately following the OdiInvokeWebService call. The request can alternately be passed through a file whose location is provided with the -REQUEST_FILE parameter.
-REQUEST_FILE=<xml_request_file>	No	Location of the XML file containing the request message in SOAP format. The request can alternately be directly written out in the tool call (<xmlRequest>).

Parameters	Mandatory	Description
-RESPONSE_MODE=<NO_FILE|NEW_FILE|FILE_APPEND>	No	Generation mode for the response file. This parameter takes the following values: • NO_FILE (default): No response file is generated. • NEW_FILE: A new response file is generated. If the file already exists, it is overwritten. • FILE_APPEND: The response is appended to the file. If the file does not exist, it is created.
-RESPONSE_FILE=<file>	Depends	The name of the result file to write. Mandatory if -RESPONSE_MODE is NEW_FILE or APPEND.
-RESPONSE_FILE_CHARSET=<charset>	Depends	Response file character encoding. See the following table. Mandatory if -RESPONSE_MODE is NEW_FILE or APPEND.
-RESPONSE_XML_ENCODING=<charset>	Depends	Character encoding that will be indicated in the XML declaration header of the response file. See the following table. Mandatory if -RESPONSE_MODE is not NO_FILE.
-RESPONSE_FILE_FORMAT=<XML|SOAP>	No	Format of the request and response file. • If XML is selected (default), the request is processed as a SOAP body. The tool adds a default SOAP header and envelope content to this body before sending the request. The response is stripped from its SOAP envelope and headers and only the response's body is written to the response file. • If SOAP is selected, the request is processed as a full-formed SOAP envelope and is sent as is. The response is also written to the response file with no processing.
-HTTP_USER=<user>	No	User account authenticating on the HTTP server.
-HTTP_PASS=<password>	No	Password of the HTTP user. **Note**: When using an ODI variable as the password, the variable content must be encrypted using the encode script.

Parameters	Mandatory	Description
-TIMEOUT=<timeout>	No	The web service request waits for a reply for this amount of time before considering that the server will not provide a response and an error is produced. The default value is 15 seconds.

The following table lists the most common XML/Java character encoding schemes. For a more complete list, see:

https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html

XML Charset	Java Charset
US-ASCII	ASCII
UTF-8	UTF8
UTF-16	UTF-16
ISO-8859-1	ISO8859_1

Examples

The following web service call returns the capital city for a given country (the ISO country code is sent in the request). Note that the request and response format, as well as the port and operations available, are defined in the WSDL passed in the URL parameter.

```
OdiInvokeWebService -
-URL=http://www.oorsprong.org/websamples.countryinfo/CountryInfoService.wso
?WSDL -PORT_TYPE=CountryInfoServiceSoapType -OPERATION=CapitalCity
-RESPONSE_MODE=NEW_FILE -RESPONSE_XML_ENCODING=ISO-8859-1
"-RESPONSE_FILE=/temp/result.xml" -RESPONSE_FILE_CHARSET=ISO8859_1 -
RESPONSE_FILE_FORMAT=XML
<CapitalCityRequest>
<sCountryISOCode>US</sCountryISOCode>
</CapitalCityRequest>
```

The generated `/temp/result.xml` file contains the following:

```
<CapitalCityResponse>
<m:CapitalCityResponse>
<m:CapitalCityResult>Washington</m:CapitalCityResult>
</m:CapitalCityResponse>
</CapitalCityResponse>
```

Packages

Oracle Data Integrator provides a special graphical interface for calling OdiInvokeWebService in packages. See the chapter Using Web Services in *Developing Integration Projects with Oracle Data Integrator* for more information.

OdiKillAgent

Use this command to stop a standalone agent.

Java EE Agents deployed in an application server cannot be stopped using this tool and must be stopped using the application server utilities.

Usage

```
OdiKillAgent (-PORT=<TCP/IP Port>|-NAME=<physical_agent_name>)
[-IMMEDIATE=<yes|no>] [-MAX_WAIT=<timeout>]
```

Parameter

Parameters	Mandatory	Description
`-PORT=<TCP/IP Port>`	No	If this parameter is specified, the agent running on the local machine with the specified port is stopped.
`-NAME=<physical_agent_name>`	Yes	If this parameter is specified, the physical agent whose name is provided is stopped. This agent may be a local or remote agent. It must be declared in the master repository.
`-IMMEDIATE=<yes|no>`	No	If this parameter is set to Yes, the agent is stopped without waiting for its running sessions to complete. If this parameter is set to No, the agent is stopped after its running sessions reach completion or after the `-MAX_WAIT` timeout is reached. The default value is No.
`-MAX_WAIT=<timeout>`	No	This parameter can be used when `-IMMEDIATE` is set to No. The parameter defines a timeout in milliseconds after which the agent is stopped regardless of the running sessions. The default value is 0, which means no timeout and the agent is stopped after its running sessions complete.

Examples

Stop the `ODI_AGT_001` physical agent immediately.

```
OdiKillAgent -NAME=ODI_AGT_001 -IMMEDIATE=yes
```

OdiManageOggProcess

Use this command to start and stop Oracle GoldenGate processes.

The `-NB_PROCESS` parameter specifies the number of processes on which to perform the operation and applies only to Oracle GoldenGate Delivery processes.

If `-NB_PROCESS` is not specified, the name of the physical process is derived from the logical process. For example, if logical schema `R1_LS` maps to physical process `R1`, an Oracle GoldenGate process named `R1` is started or stopped.

If `-NB_PROCESS` is specified with a positive value, sequence numbers are appended to the process and all processes are started or stopped with the new name. For example, if the

value is set to `3`, and logical schema `R2_LS` maps to physical process `R2`, processes `R21`, `R22` and `R23` are started or stopped.

If Start Journal is used to start the CDC (Changed Data Capture) process with Oracle GoldenGate JKMs (Journalizing Knowledge Modules), Oracle Data Integrator generates the Oracle GoldenGate Delivery process with the additional sequence number in the process name. For example, if Delivery process `RP` is used for the Start Journal action, Start Journal generates an Oracle GoldenGate Delivery process named `RP1`. To stop and start the process using the OdiManageOggProcess tool, set `-NB_PROCESS` to `1`. The maximum value of `-NB_PROCESS` is the value of the `-NB_APPLY_PROCESS` parameter of the JKM within the model.

Usage

```
OdiManageOggProcess -OPERATION=<start|stop>
-PROCESS_LSCHEMA=<OGG logical schema> [-NB_PROCESS=<number of processes>]
```

Parameters

Parameters	Mandatory	Description
`-OPERATION=<start|stop>`	Yes	Operation to perform on the process.
`-PROCESS_LSCHEMA=<OGG logical schema>`	Yes	Logical schema of the process.
`-NB_PROCESS=<number of processes>`	No	Number of processes on which to perform the operation.

Examples

Start Oracle GoldenGate process `R1`, which maps to logical schema `R1_LS`.

```
OdiManageOggProcess "-OPERATION=START" "-PROCESS_LSCHEMA=R1_LS
```

OdiMkDir

Use this command to create a directory structure.

If the parent directory does not exist, this command recursively creates the parent directories.

Usage

```
OdiMkDir -DIR=<directory>
```

Parameters

Parameters	Mandatory	Description
`-DIR=<directory>`	Yes	Directory (or folder) to create.
`-TO_HDFS=<yes|no>`	No	Indicates if the target is HDFS

Examples

Create the directory `odi` in `C:\temp`. If `C:\temp` does not exist, it is created.

```
OdiMkDir "-DIR=C:\temp\odi"
```

OdiObjectStorageDelete

Use this command to delete single or multiple files or an entire directory present in Oracle Object Storage.

> **NOT_SUPPORTED:**
>
> This tool is applicable only for Data Integration Platform Cloud.

Usage

```
OdiObjectStorageDelete
-TRG_LOGICAL_SCHEMA=<trg_logical_schema_name>
-FILE_NAMES_FILTER =<file_names_filter>
-RETRY_ON_ERROR=<retry_number>
-RETRY_INTERVAL_SECONDS =<retry_interval_seconds>
```

Parameters

Parameter	Mandatory	Description
TRG_LOGICAL_SCHEMA	Yes	Target logical schema has the details of Oracle Object Storage Data Server which contains the files and directories that are to be deleted.
FILE_NAMES_FILTER	Yes	Field to specify one or more files or directories to be deleted from Oracle Object Storage recursively. It also supports delimiter | for separated files list. The pattern followed is: • *.txt - should delete all files ending with .txt • test* - Deletes all the files and directories that matches with prefix “test” • *test* - Deletes all the files and directories having substring “test” • test.xml | test1.xml | test2.xml - Deletes all the files specified • test* | test1* - Deletes all the files and directories matching pattern test* and test1* • test.xml - Only one file is deleted.
RETRY_ON_ERROR	No	It represents the number of times the retry attempt should occur when a failure or error happens during delete.
RETRY_INTERVAL_SECONDS	No	Retry interval indicates after how many seconds a retry attempt should happen.

Example

The following command is used to delete file(s) and directories from Oracle Object Storage

```
OdiObjectStorageDelete -TRG_LOGICAL_SCHEMA=OBJ_LS_OBJDELETE -
FILE_NAMES_FILTER=*SE* -RETRY_ON_ERROR=2 -RETRY_INTERVAL_SECONDS=3
```

OdiObjectStorageDownload

Use this command to download single or multiple files or an entire directory to HDFS or a local file system from Oracle Object Storage.

> **NOT_SUPPORTED:**
>
> This tool is applicable only for Data Integration Platform Cloud.

Usage

```
OdiObjectStorageDownload
SRC_LOGICAL_SCHEMA=<src_logical_schema_name>
-TRG_LOGICAL_SCHEMA=<trg_logical_schema_name>
-FILE_NAMES_FILTER = <file_names_filter>
-OVERWRITE= Yes|No
-RETRY_ON_ERROR=<retry_number>
-RETRY_INTERVAL_SECONDS = <retry_interval_seconds>
```

Parameters

Parameter	Mandatory	Description
SRC_LOGICAL_SCHEMA	Yes	Source Logical schema name configured for Oracle Object Storage Data Server.
TRG_LOGICAL_SCHEMA	Yes	Generally the download operation downloads the file from Oracle Object Storage to Local or HDFS file system. The Target logical schema specifies whether the files are downloaded to Local or HDFS file system.

Parameter	Mandatory	Description
FILE_NAMES_FILTER	Yes	Field to specify one or more files to be downloaded from Oracle Object Storage recursively. It also supports delimiter | for separated files list. The pattern followed is: • *.txt - should download all files ending with .txt • test* - Downloads all the files and directories that matches with prefix “test” • *test* - Downloads all the files and directories having substring “test” • test.xml | test1.xml | test2.xml - Downloads all the files specified • test* | test1* - Downloads all the files matching pattern test* and test1* • test.xml - Only one file is downloaded.
OVERWRITE	No	This parameter indicates, if download operation should overwrite an existing file or not. The default value for this parameter is No.
RETRY_ON_ERROR	No	It represents the number of times the retry attempt should occur when a failure or error happens during download.
RETRY_INTERVAL_SECONDS	No	Retry interval indicates after how many seconds a retry attempt should happen.

Example

The following command is used to download file(s) and directories from Oracle Object Storage.

```
OdiObjectStorageDownload -SRC_LOGICAL_SCHEMA=OBJ_LS_OBJDOWNLOAD -
TRG_LOGICAL_SCHEMA=FILE_LS_OBJDOWNLOAD -FILE_NAMES_FILTER=*SE* -
OVERWRITE=yes -RETRY_ON_ERROR=2 -RETRY_INTERVAL_SECONDS=3
```

OdiObjectStorageUpload

Use this tool to upload single or multiple files or an entire directory from HDFS or a local file system on to Oracle Object Storage.

> **NOT_SUPPORTED:**
>
> This tool is applicable only for Data Integration Platform Cloud.

Usage

```
OdiObjectStorageUpload
-TRG_LOGICAL_SCHEMA=<trg_logical_schema_name>
```

```
-SRC_LOGICAL_SCHEMA = <src_logical_schema_name>
-FILE_NAMES_FILTER =<file_names_filter>
-OVERWRITE = Yes|No
-RETRY_ON_ERROR =<retry_number>
-RETRY_INTERVAL_SECONDS =<retry_interval_seconds>
```

Parameters

Parameters	Mandatory	Description
TRG_LOGICAL_SCHEMA	Yes	Target Logical schema name configured for Oracle Object Storage Data Server.
SRC_LOGICAL_SCHEMA	Yes	Source Logical schema name configured for File or HDFS Data Server for upload of Local or HDFS Files to Oracle Object Storage.
FILE_NAMES_FILTER	Yes	Field to specify one or more files to be uploaded to Oracle Object Storage recursively. It also supports the list of files separated by | as a delimiter. The pattern followed is: • *.txt - should upload all the files ending with .txt • test* - uploads all the files and directories that matches with prefix “test” • *test* - uploads all the files and directories having substring “test”. • test.xml | test1.xml | test2.xml - Uploads all the files specified. • test* | test1* - Uploads all the files matching pattern test* and test1* • test.xml - Only one file is uploaded.
OVERWRITE	No	This parameter indicates if upload operation should overwrite an existing file or not. Default value for this parameter is No.
RETRY_ON_ERROR	No	It represents the number of times the retry attempt should occur when a failure or error happens during upload.
RETRY_INTERVAL_SECONDS	No	Retry interval indicates after how many seconds a retry attempt should happen.

> **Note:**
>
> The upload operation fails if the selected bucket does not exist.

Example

The following command is used to upload file(s) and directories to Oracle Object Storage:

```
OdiObjectStorageUpload -SRC_LOGICAL_SCHEMA=FILE_LS_OBJUPLOAD -
TRG_LOGICAL_SCHEMA=OBJ_LS_OBJUPLOAD -FILE_NAMES_FILTER=*SE* -OVERWRITE=yes -
RETRY_ON_ERROR=2 -RETRY_INTERVAL_SECONDS=3
```

OdiOggCommand

Use this command to integrate the Oracle GoldenGate based CDC mechanism by performing specific tasks at runtime to interact with the GoldenGate process.

Usage

```
OdiOggCommand -OPERATION="ADDEXTRACT" -LSCHEMA="%EXTRACT_LSCHEMA%"
OdiOggCommand -OPERATION="ADDREPLICAT" -LSCHEMA="%REPLICAT_LSCHEMA%"
OdiOggCommand -OPERATION="DROPPROCESS" -LSCHEMA="%PROCESS_LSCHEMA%"
OdiOggCommand -OPERATION="STARTPROCESS" -LSCHEMA="%PROCESS_LSCHEMA%"
OdiOggCommand -OPERATION="STOPPROCESS" -LSCHEMA="%PROCESS_LSCHEMA%"
OdiOggCommand -OPERATION="SAVEPARAM" -LSCHEMA="%PROCESS_LSCHEMA%" -FILEPATH="%TMP/
PRMFILE%"
OdiOggCommand -OPERATION="ADDPUMP" -LSCHEMA="%EXTRACT_LSCHEMA%" -NAME="%PUMPNAME%"
OdiOggCommand -OPERATION="ADDCHECKPOINTTABLE" -LSCHEMA="%REPLICAT_LSCHEMA%" -
TABLE="TABLE_NAME"
OdiOggCommand -OPERATION="DEFGEN” -LSCHEMA="%EXTRACT_LSCHEMA%" -
TGT_LSCHEMA="%REPLICAT_LSCHEMA%"
OdiOggCommand -OPERATION="ADDTRANDATA" -LSCHEMA="%EXTRACT_LSCHEMA%" -
TABLE_NAME="%TABLENAME%" -COLLIST="%[col1,col2]%"
OdiOggCommand -OPERATION="DBLOGIN" -LSCHEMA="%PROCESS_LSCHEMA%" -
MODEL_LSCHEMA_NAME="%EXTR_MODEL_DB_LSCHEMA%"
OdiOggCommand -OPERATION="DBLOGIN" -LSCHEMA="%PROCESS_LSCHEMA%" -
PSCHEMA_NAME="%REPLICAT_TGT_PSCHEMA%"
```

Operations

Operation	Description	Required Parameter	Supported Custom Parameters	Remarks
ADDEXTRACT	Adds the Extract process to GoldenGate through JAgent.	-LSCHEMA="%EXTRACT_LSCHEMA%"	-BEGINNOW="TRUE|FALSE"	Retrieves the Extract and JAgent host details.
ADDREPLICAT	Adds the Replicat process to GoldenGate through JAgent.	-LSCHEMA="%REPLICAT_LSCHEMA%"	N/A	Retrieves the Replicat and JAgent host details.
DROPPROCESS	Deletes the process associated with the logical schema.	-LSCHEMA="%PROCESS_LSCHEMA%"	N/A	Retrieves the host details based on the type of the process and logical schema.

Operation	Description	Required Parameter	Supported Custom Parameters	Remarks
STARTPROCESS	Starts the process associated with the logical schema.	-LSCHEMA="%PROCESS_LSCHEMA%"	N/A	Retrieves the host details based on the type of the process and logical schema.
STOPPROCESS	Stops the process associated with the logical schema.	-LSCHEMA="%PROCESS_LSCHEMA%"	N/A	Retrieves the host details based on the type of the process and logical schema.
SAVEPARAM	Uploads the param file.	-LSCHEMA="%PROCESS_LSCHEMA%"	-FILEPATH="%TMP/PRMFILE%"	Saves the param file associated to the process in the JAgent host associated with the logical schema.
ADDPUMP	Adds a pump process in the JAgent host associated with the Extract process.	-LSCHEMA="%EXTRACT_LSCHEMA%"	-NAME="%PUMPNAME%"	The name of the pump is considered to be REPLICAT_NAME#P to associate a pump process to the Replicat process.
ADDCHECKPOINTTABLE	Adds the checkpoint table.	-LSCHEMA="%REPLICAT_LSCHEMA%"	-TABLE="TABLE_NAME"	The table name is obtained from the JKM option.
DEFGEN	Loads and runs defgen.	-LSCHEMA="%EXTRACT_LSCHEMA%"	-TGT_LSCHEMA="%REPLICAT_LSCHEMA%"	The defgen run on the Extract source will be saved to the target Replicat host.
ADDTRANDATA	Enables tran data in Extract source.	-LSCHEMA="%EXTRACT_LSCHEMA%"	-TABLE_NAME="%TABLENAME%" -COLLIST="%[col1,col2]%"	This has to be run for the Extract JAgent host. The details of the tables and columns have to be provided
DBLOGIN	Database login to enable GoldenGate operations.	-LSCHEMA="%PROCESS_LSCHEMA%"	-MODEL_LSCHEMA_NAME="%EXTR_MODEL_DB_LSCHEMA%"	The user name and password required to log in to the database can be retrieved from journalized model logical schema and current context.
DBLOGIN	Database login to enable GoldenGate operations.	-LSCHEMA="%PROCESS_LSCHEMA%"	-PSCHEMA_NAME="%REPLICAT_TGT_PSCHEMA%"	The user name and password required to log in to the database can be retrieved from model physical schema assigned to Replicat process and current context.

Examples

Add the Extract process to GoldenGate.

```
OdiOggCommand -OPERATION="ADDEXTRACT" -
LSCHEMA="<%=odiRef.getOggProcessLschemaName("EXTRACT")%>"
```

Log in to the database by retrieving the user name and password from the physical schema assigned to the Replicat process.

```
OdiOggCommand -OPERATION="DBLOGIN" -
LSCHEMA="<%=odiRef.getOggProcessLschemaName("REPLICAT")%>"
-
PSCHEMA_NAME="<%=odiRef.getOggProcessInfo("<%=odiRef.getOggProcessLschemaName("REPLICAT
")%>"),"DB_PSCHEMA")"
```

Add a pump process in the JAgent host associated with the Extract process.

```
OdiOggCommand -OPERATION="ADDPUMP" -
LSCHEMA=""<%=odiRef.getOggProcessLschemaName("EXTRACT")%>""
-NAME="<%=odiRef.getProcessInfo("<%=odiRef.getOggProcessLschemaName("REPLICAT")
%>"),"NAME")"
```

OdiOSCommand

Use this command to invoke an operating system command shell to carry out a command, and redirect the output result to files.

The following operating systems are supported:

- Windows operating systems, using `cmd`
- POSIX-compliant operating systems, using `sh`

The following operating system is not supported:

- Mac OS

Usage

```
OdiOSCommand [-OUT_FILE=<stdout_file>] [-ERR_FILE=<stderr_file>]
[-FILE_APPEND=<yes|no>] [-WORKING_DIR=<workingdir>] [-SYNCHRONOUS=<yes|no>]
[CR/LF <command> | -COMMAND=<command>]
```

Parameters

Parameters	Mandatory	Description
`-COMMAND=<command>`	Yes	Command to execute. For a multiline command, pass the whole command as raw text after the OdiOSCommand line without the `-COMMAND` parameter.
`-OUT_FILE=<stdout_file>`	No	Absolute name of the file to redirect standard output to.
`-ERR_FILE=<stderr_file>`	No	Absolute name of the file to redirect standard error to.
`-FILE_APPEND=<yes|no>`	No	Whether to append to the output files, rather than overwriting them. The default value is Yes.
`-WORKING_DIR=<workingdir>`	No	Directory in which the command is executed.

Parameters	Mandatory	Description
-SYNCHRONOUS=<yes|no>	No	If set to Yes (default), the session waits for the command to terminate. If set to No, the session continues immediately with error code 0. The default is synchronous mode.
-CAPTURE_OUT_STREAM=[ON_ERROR[,]][ALL|NONE|[NSTART][,NEND]]	No	Use to capture some of the content that is written to the output stream and display in the Task Execution details in Operator. If set to **ON_ERROR**, the content will be captured only if the task fails. If set to **ALL** or **NONE**, either all or none of the output stream will be captured. Use **NSTART** and **NEND** to specify the number of lines to be captured (from the start and end).
-CAPTURE_ERR_STREAM=[ON_ERROR[,]][ALL|NONE|[NSTART][,NEND]]	No	Use to capture some of the content that is written to the error stream and display in the Task Error Message in Operator. If set to **ON_ERROR**, the content will be captured only if the task fails. If set to **ALL** or **NONE**, either all or none of the output stream will be captured. Use **NSTART** and **NEND** to specify the number of lines to be captured (from the start and end).

Examples

Execute the file `c:\work\load.bat` (on a Windows machine) and append the output streams to files.

```
OdiOSCommand "-OUT_FILE=c:\work\load-out.txt"
"-ERR_FILE=c:\work\load-err.txt" "-FILE_APPEND=YES"
"-WORKING_DIR=c:\work" c:\work\load.bat
```

OdiOutFile

Use this command to write or append content to a text file.

Usage

```
OdiOutFile -FILE=<file_name> [-APPEND] [-CHARSET_ENCODING=<encoding>]
[-XROW_SEP=<hexadecimal_line_break>] [CR/LF <text> | -TEXT=<text>]
```

Parameters

Parameters	Mandatory	Description
-FILE=<file_name>	Yes	Target file. The file location is always relative to the data schema directory of its logical schema.

Parameters	Mandatory	Description
-APPEND	No	Indicates whether <text> must be appended at the end of the file. If this parameter is not specified, the file is overwritten if it exists.
-CHARSET_ENCODING=<encoding>	No	Target file encoding. The default value is ISO-8859-1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html
-XROW_SEP=<hexadecimal_line_break>	No	Hexadecimal code of the character used as a line separator (line break). The default value is 0A (UNIX line break). For a Windows line break, the value is 0D0A.
CR/LF <text> or -TEXT=<text>	No	Text to write in the file. This text can be typed on the line following the OdiOutFile command (a carriage return - CR/LF - indicates the beginning of the text), or can be defined with the -TEXT parameter. The -TEXT parameter should be used when calling this Oracle Data Integrator command from an OS command line. The text can contain variables or substitution methods.
-TO_HDFS=<yes|no>	No	Indicates if the output file is created in HDFS
-TGT_LSCHEMA	No	Indicates if the file is located on a data server resolved based on the Logical Schema value.

Examples

Generate the file /var/tmp/my_file.txt on the UNIX system of the agent that executed it.

```
OdiOutFile -FILE=/var/tmp/my_file.txt
Welcome to Oracle Data Integrator
This file has been overwritten by <%=odiRef.getSession("SESS_NAME")%>
```

Add the entry PLUTON into the file hosts of the Windows system of the agent that executed it.

```
OdiOutFile -FILE=C:\winnt\system32\drivers\etc\hosts -APPEND
192.0.2.1 PLUTON pluton
```

OdiPingAgent

Use this command to perform a test on a given agent. If the agent is not started, this command raises an error.

Usage

```
OdiPingAgent -AGENT_NAME=<physical_agent_name>
```

Parameters

Parameters	Mandatory	Description
-AGENT_NAME=<physical_agent_name>	Yes	Name of the physical agent to test.

Examples

Test the physical agent `AGENT_SOLARIS_DEV`.

```
OdiPingAgent -AGENT_NAME=AGENT_SOLARIS_DEV
```

OdiPurgeLog

Use this command to purge the execution logs.

The OdiPurgeLog tool purges all session logs and/or Load Plan runs that match the filter criteria.

The `-PURGE_TYPE` parameter defines the objects to purge:

- Select `SESSION` to purge all session logs matching the criteria. Child sessions and grandchild sessions are purged if the parent session matches the criteria. Note that sessions launched by a Load Plan execution, including the child sessions, are *not* purged.
- Select `LOAD_PLAN_RUN` to purge all load plan logs matching the criteria. Note that all sessions launched from the Load Plan run are purged even if the sessions attached to the Load Plan runs themselves do not match the criteria.
- Select `ALL` to purge both session logs and Load Plan runs matching the criteria.

The `-COUNT` parameter defines the number of sessions and/or Load Plan runs (after filter) to preserve in the log. The `-ARCHIVE` parameter enables automatic archiving of the purged sessions and/or Load Plan runs.

> **Note:**
>
> Load Plans and sessions in running, waiting, or queued status are not purged.

Usage

```
OdiPurgeLog
[-PURGE_TYPE=<SESSION|LOAD_PLAN_RUN|ALL>]
[-COUNT=<session_number>] [-FROMDATE=<from_date>] [TODATE=<to_date>]
[-CONTEXT_CODE=<context_code>] [-USER_NAME=<user_name>]
[-AGENT_NAME=<agent_name>] [-PURGE_REPORTS=<Yes|No>] [-STATUS=<D|E|M>]
[-NAME=<session_or_load_plan_name>] [-ARCHIVE=<Yes|No>] [-EXPORT_KEY=<key>]
[-TODIR=<directory>]
[-ZIPFILE_NAME=<zipfile_name>] [-XML_CHARSET=<charset>] [-
JAVA_CHARSET=<charset>]
[-REMOVE_TEMPORARY_OBJECTS=<yes|no>] [ARCHIVE_WITHOUT_CIPHER_DATA=<yes|no>]
```

Parameters

Parameters	Mandatory	Description
-PURGE_TYPE=<SESSION|LOAD_PLAN_RUN|ALL>	No	Purges only session logs, Load Plan logs, or both. The default is session.
-COUNT=<session_number>	No	Retains the most recent count number of sessions and/or Load Plan runs that match the specified filter criteria and purges the rest. If this parameter is not specified or equals 0, purges all sessions and/or Load Plan runs that match the filter criteria.
-FROMDATE=<from_date>	No	Starting date for the purge, using the format yyyy/MM/dd hh:mm:ss. If -FROMDATE is omitted, the purge is done starting with the oldest session and/or Load Plan run.
-TODATE=<to_date>	No	Ending date for the purge, using the format yyyy/MM/dd hh:mm:ss. If -TODATE is omitted, the purge is done up to the most recent session and/or Load Plan run.
-CONTEXT_CODE=<context_code>	No	Purges only sessions and/or Load Plan runs executed in <context_code>. If -CONTEXT_CODE is omitted, the purge is done on all contexts.
-USER_NAME=<user_name>	No	Purges only sessions and/or Load Plan runs launched by <user_name>.
-AGENT_NAME=<agent_name>	No	Purges only sessions and/or Load Plan runs executed by <agent_name>.
-PURGE_REPORTS=<0|1>	No	If set to 1, scenario reports (appearing under the execution node of each scenario) are also purged.
-STATUS=<D|E|M>	No	Purges only the sessions and/or Load Plan runs with the specified state: • D: Done • E: Error • M: Warning If this parameter is not specified, sessions and/or Load Plan runs in all of these states are purged.
-NAME=<session_or_load_plan_name>	No	Session name or Load Plan name.

Parameters	Mandatory	Description
-ARCHIVE=<Yes|No>	No	If set to Yes, exports the sessions and/or Load Plan runs before they are purged.
-EXPORT_KEY=<key>	No[1]	Specifies a cryptographic private key used to encrypt sensitive cipher data. You must specify this key again when importing the exported object in order to import the cipher data.
-ARCHIVE_WITHOUT_CIPHER_DATA=<yes|no>	No[2]	When set to Yes, specifies that sensitive (cipher) values should be set to null in the object when it is archived. When set to No or when this parameter is omitted, you must include the -EXPORT_KEY parameter and specify a valid key. The default value is No.
-TODIR=<directory>	No	Target directory for the export. This parameter is required if -ARCHIVE is set to Yes.
-ZIPFILE_NAME=<zipfile_name>	No	Name of the compressed file. Target directory for the export. This parameter is required if -ARCHIVE is set to Yes.
-XML_CHARSET=<charset>	No	XML encoding of the export files. The default value is ISO-8859-1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html
-JAVA_CHARSET=<charset>	No	Export file encoding. The default value is ISO8859_1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html
-REMOVE_TEMPORARY_OBJECTS=<yes|no>	No	If set to Yes (default), cleanup tasks are performed before sessions are purged so that any temporary objects are removed.

1 If the -EXPORT_KEY parameter is not specified, the -ARCHIVE_WITHOUT_CIPHER_DATA parameter must be specified, and must be set to Yes.
2 If -ARCHIVE_WITHOUT_CIPHER_DATA is not specified, or if it is specified and set to No, you must specify the -EXPORT_KEY parameter with a valid key value.

Examples

Purge all sessions executed between 2001/03/25 00:00:00 and 2001/08/31 21:59:00.

```
OdiPurgeLog "-FROMDATE=2001/03/25 00:00:00" "-TODATE=2001/08/31 21:59:00"
```

Purge all Load Plan runs that were executed in the `GLOBAL` context by the `Internal` agent and that are in Error status.

```
OdiPurgeLog "-PURGE_TYPE=LOAD_PLAN_RUN" "-CONTEXT_CODE=GLOBAL"
"-AGENT_NAME=Internal" "-STATUS=E"
```

OdiReadMail

Use this command to read emails and attachments from a POP or IMAP account.

This command connects the mail server `-MAILHOST` using the connection parameters specified by `-USER` and `-PASS`. The execution agent reads messages from the mailbox until `-MAX_MSG` messages are received or the maximum waiting time specified by `-TIMEOUT` is reached. The extracted messages must match the filters such as those specified by the parameters `-SUBJECT` and `-SENDER`. When a message satisfies these criteria, its content and its attachments are extracted in a directory specified by the parameter `-FOLDER`. If the parameter `-KEEP` is set to No, the retrieved message is suppressed from the mailbox.

Usage

```
OdiReadMail -MAILHOST=<mail_host> -USER=<mail_user>
-PASS=<mail_user_password> -FOLDER=<folder_path>
[-PROTOCOL=<pop3|imap>] [-FOLDER_OPT=<none|sender|subject>]
[-KEEP=<no|yes>] [-EXTRACT_MSG=<yes|no>] [-EXTRACT_ATT=<yes|no>]
[-MSG_PRF=<my_prefix>] [-ATT_PRF=<my_prefix>] [-USE_UCASE=<no|yes>]
[-NOMAIL_ERROR=<no|yes>] [-TIMEOUT=<timeout>] [-POLLINT=<pollint>]
[-MAX_MSG=<max_msg>] [-SUBJECT=<subject_filter>] [-SENDER=<sender_filter>]
[-TO=<to_filter>] [-CC=<cc_filter>]
```

Parameters

Parameters	Mandatory	Description
`-MAILHOST=<mail_host>`	Yes	IP address of the POP or IMAP mail server.
`-USER=<mail_user>`	Yes	Valid mail server account.
`-PASS=<mail_user_password>`	Yes	Password of the mail server account.
`-FOLDER=<folder_path>`	Yes	Full path of the storage folder for attachments and messages.
`-PROTOCOL=<pop3|imap>`	No	Type of mail accessed (POP3 or IMAP). The default is POP3.

Parameters	Mandatory	Description
-FOLDER_OPT=<none|sender|subject>	No	Allows the creation of a subdirectory in the directory -FOLDER according to the following parameters: • none (default): No action. • sender: A subdirectory is created with the external name of the sender. • subject: A subdirectory is created with the subject of the message. For the sender and subject folder options, the spaces and nonalphanumeric characters (such as @) are replaced by underscores in the generated folder's name.
-KEEP=<no|yes>	No	If set to Yes, keeps the messages that match the filters in the mailbox after reading them. If set to No (default), deletes the messages that match the filters of the mailbox after reading them.
-EXTRACT_MSG=<yes|no>	No	If set to Yes (default), extracts the body of the message into a file. If set to No, does not extract the body of the message into a file.
-EXTRACT_ATT=<yes|no>	No	If set to Yes (default), extracts the attachments into files. If set to No, does not extract attachments.
-MSG_PRF=<my_prefix>	No	Prefix of the file that contains the body of the message. The default is MSG.
-ATT_PRF=<my_prefix>	No	Prefix of the files that contain the attachments. The original file names are kept.
-USE_UCASE=<no|yes>	No	If set to Yes, forces the file names to uppercase. If set to No (default), keeps the original letter case.
-NOMAIL_ERROR=<no|yes>	No	If set to Yes, generates an error when no mail matches the specified criteria. If set to No (default), does not generate an error when no mail corresponds to the specified criteria.

Parameters	Mandatory	Description
-TIMEOUT=<timeout>	No	Maximum waiting time in milliseconds. If this waiting time is reached, the command ends. The default value is 0, which means an infinite waiting time (as long as needed for the maximum number of messages specified with -MAX_MSG to be received).
-POLLINT=<pollint>	No	Searching interval in milliseconds to scan for new messages. The default value is 1000 (1 second).
-MAX_MSG=<max_msg>	No	Maximum number of messages to extract. If this number is reached, the command ends. The default value is 1.
-SUBJECT=<subject_filter>	No	Parameter used to filter the messages according to their subjects.
-SENDER=<sender_filter>	No	Parameter used to filter messages according to their sender.
-TO=<to_filter>	No	Parameter used to filter messages according to their addresses. This option can be repeated to create multiple filters.
-CC=<cc_filter>	No	Parameter used to filter messages according to their addresses in copy. This option can be repeated to create multiple filters.

Examples

Automatic reception of the mails of support with attachments detached in the folder C:\support on the system of the agent. Wait for all messages with a maximum waiting time of 10 seconds.

```
OdiReadMail -MAILHOST=mail.example.com -USER=myaccount -PASS=mypass
-KEEP=no -FOLDER=c:\support -TIMEOUT=0 -MAX_MSG=0
-SENDER=support@example.com -EXTRACT_MSG=yes -MSG_PRF=TXT
-EXTRACT_ATT=yes
```

Wait indefinitely for 10 messages and check for new messages every minute.

```
OdiReadMail -MAILHOST=mail.example.com -USER=myaccount -PASS=mypass
-KEEP=no -FOLDER=c:\support -TIMEOUT=0 -MAX_MSG=10 -POLLINT=60000
-SENDER=support@example.com -EXTRACT_MSG=yes -MSG_PRF=TXT
-EXTRACT_ATT=yes
```

OdiRefreshJournalCount

Use this command to refresh for a given journalizing subscriber the number of rows to consume for the given table list or CDC set. This refresh is performed on a logical schema and a given context, and may be limited.

> **Note:**
>
> This command is suitable for journalized tables in simple or consistent mode and cannot be executed in a command line with `startcmd`.

Usage

```
OdiRefreshJournalCount -LSCHEMA=<logical_schema>
-SUBSCRIBER_NAME=<subscriber_name>
(-TABLE_NAME=<table_name> | -CDC_SET_NAME=<cdc set name>)
[-CONTEXT=<context>] [-MAX_JRN_DATE=<to_date>]
```

Parameters

Parameters	Mandatory	Description
`-LSCHEMA=<logical_schema>`	Yes	Logical schema containing the journalized tables.
`-TABLE_NAME=<table_name>`	Yes for working with simple CDC	Journalized table name, mask, or list to check. This parameter accepts three formats: • Table Name • Table Name Mask: This mask selects the tables to poll. The mask is specified using the SQL LIKE syntax: the `%` symbol replaces an unspecified number of characters and the _ symbol acts as a wildcard. • Table Names List: List of table names separated by commas. Masks as defined above are **not** allowed. Note that this option works only for tables in a model journalized in simple mode. This parameter cannot be used with `-CDC_SET_NAME`. It is mandatory if `-CDC_SET_NAME` is not set.
`-CDC_SET_NAME=<cdcSetName>`	Yes for working with consistent set CDC	Name of the CDC set to check. Note that this option works only for tables in a model journalized in consistent mode. This parameter cannot be used with `-TABLE_NAME`. It is mandatory if `-TABLE_NAME` is not set.
`-SUBSCRIBER_NAME=<subscriber_name>`	Yes	Name of the subscriber for which the count is refreshed.
`-CONTEXT=<context>`	No	Context in which the logical schema will be resolved. If no context is specified, the execution context is used.

Parameters	Mandatory	Description
-MAX_JRN_DATE=<to_date>	No	Date (and time) until which the journalizing events are taken into account.

Examples

Refresh for the CUSTOMERS table in the SALES_APPLICATION schema the count of modifications recorded for the SALES_SYNC subscriber. This datastore is journalized in simple mode.

```
OdiRefreshJournalCount -LSCHEMA=SALES_APPLICATION
-TABLE_NAME=CUSTOMERS -SUBSCRIBER_NAME=SALES_SYNC
```

Refresh for all tables from the SALES CDC set in the SALES_APPLICATION schema the count of modifications recorded for the SALES_SYNC subscriber. These datastores are journalized with consistent set CDC.

```
OdiRefreshJournalCount -LSCHEMA=SALES_APPLICATION
-SUBSCRIBER_NAME=SALES_SYNC -CDC_SET_NAME=SALES
```

OdiReinitializeSeq

Use this command to reinitialize an Oracle Data Integrator sequence.

Usage

```
OdiReinitializeSeq -SEQ_NAME=<sequence_name> -CONTEXT=<context>
-STD_POS=<position>
```

Parameters

Parameters	Mandatory	Description
-SEQ_NAME=<sequence_name>	Yes	Name of the sequence to reinitialize. It must be prefixed with GLOBAL. for a global sequence, or by <project code>. for a project sequence.
-CONTEXT=<context>	Yes	Context in which the sequence must be reinitialized.
-STD_POS=<position>	Yes	Position to which the sequence must be reinitialized.

Examples

Reset the global sequence SEQ_I to 0 for the GLOBAL context.

```
OdiReinitializeSeq -SEQ_NAME=GLOBAL.SEQ_I -CONTEXT=GLOBAL
-STD_POS=0
```

OdiRemoveTemporaryObjects

Use this command to remove temporary objects that could remain between executions. This is performed by executing the cleanup tasks for the sessions identified by the parameters specified in the tool parameters.

Usage

```
OdiRemoveTemporaryObjects [-COUNT=<session_number>] [-FROMDATE=<from_date>]
[-TODATE=<to_date>] [-CONTEXT_CODE=<context_code>]
[-AGENT_NAME=<agent_name>] [-USER_NAME=<user_name>]
[-NAME=<session_name>] [-ERRORS_ALLOWED=<number_of_errors_allowed>]
```

Parameter

Parameters	Mandatory	Description
-COUNT=<session_number>	No	Number of sessions to skip cleanup for. The most recent number of sessions (<session_number>) is kept and the rest are cleaned up.
-FROMDATE=<from_date>	No	Start date for the cleanup, using the format yyyy/MM/dd hh:mm:ss. All sessions started after this date are cleaned up. If -FROMDATE is omitted, the cleanup starts with the oldest session.
-TODATE=<to_date>	No	End date for the cleanup, using the format yyyy/MM/dd hh:mm:ss. All sessions started before this date are cleaned up. If -TODATE is omitted, the cleanup starts with the most recent session.
-CONTEXT_CODE=<context_code>	No	Cleans up only those sessions executed in this context (<context_code>). If -CONTEXT_CODE is omitted, cleanup is performed on all contexts.
-AGENT_NAME=<agent_name>	No	Cleans up only those sessions executed by this agent (<agent_name>).
-USER_NAME=<user_name>	No	Cleans up only those sessions launched by this user (<user_name>).
-NAME=<session_name>	No	Session name.
-ERRORS_ALLOWED=<number_of_errors_allowed>	No	Number of errors allowed before the step ends with OK. If set to 0, the step ends with OK regardless of the number of errors encountered during the cleanup phase.

Examples

Remove the temporary objects by performing the cleanup tasks of all sessions executed between 2013/03/25 00:00:00 and 2013/08/31 21:59:00.

```
OdiRemoveTemporaryObjects "-FROMDATE=2013/03/25 00:00:00" "-TODATE=2013/08/31
21:59:00"
```

Remove the temporary objects by performing the cleanup tasks of all sessions executed in the GLOBAL context by the Internal agent.

```
OdiRemoveTemporaryObjects "-CONTEXT_CODE=GLOBAL" "-AGENT_NAME=Internal"
```

OdiRetrieveHadoopLog

Use this command to retrieve log information from executions in an Oozie execution agent.

Usage

```
OdiRetrieveHadoopLog [-SESSION_LIST=<session-ids>] -POLLINT=<poll> -TIMEOUT=<timeout>
```

Parameters

Parameters	Mandatory	Description
`-SESSION_LIST=<session-ids>`	No	A comma separated list of sessions IDs to be retrieved. If blank, all Oozie sessions currently running will be retrieved.
`-POLLINT=<poll>`	No	The length of time between each instance when the log data is retrieved. Can be in secs (s), mins (m), hours (h), days (d), or years (y). If zero, the log data will be retrieved once and then the tool will end.
`-TIMEOUT=<timeout>`	No	The maximum period of time that the tool will execute for. Can be in secs(s), mins(m), hours(h), days(d) or years(h). If zero, the log will be polled an retrieved according to the poll interval and will end when no sessions are candidates for retrieval

Examples

Perform a one time retrieval of the Hadoop Log for the current session if it is being executed in an Oozie execution engine.

```
OdiRetrieveHadopLog -SESSION_LIST=<?=odiRef.getSession("SESS_NO")?>
```

OdiRetrieveJournalData

Use this command to retrieve the journalized events for a given journalizing subscriber, a given table list or CDC set. The retrieval is performed specifically for the technology containing the tables. This retrieval is performed on a logical schema and a given context.

> **Note:**
>
> This tool works for tables journalized using simple or consistent set modes and cannot be executed in a command line with `startcmd`.

Usage

```
OdiRetrieveJournalData -LSCHEMA=<logical_schema>
-SUBSCRIBER_NAME=<subscriber_name>
(-TABLE_NAME=<table_name> | -CDC_SET_NAME=<cdc_set_name>)
[-CONTEXT=<context>] [-MAX_JRN_DATE=<to_date>]
```

Parameters

Parameters	Mandatory	Description
-LSCHEMA=<logical_schema>	Yes	Logical schema containing the journalized tables.
-TABLE_NAME=<table_name>	No	Journalized table name, mask, or list to check. This parameter accepts three formats: • Table Name • Table Name Mask: This mask selects the tables to poll. The mask is specified using the SQL LIKE syntax: the % symbol replaces an unspecified number of characters and the _ symbol acts as a wildcard. • Table Names List: List of table names separated by commas. Masks as defined above are **not** allowed. Note that this option works only for tables in a model journalized in simple mode. This parameter cannot be used with -CDC_SET_NAME. It is mandatory if -CDC_SET_NAME is not set.
-CDC_SET_NAME=<cdc_set_name>	No	Name of the CDC set to update. Note that this option works only for tables in a model journalized in consistent mode. This parameter cannot be used with -TABLE_NAME. It is mandatory if -TABLE_NAME is not set.
-SUBSCRIBER_NAME=<subscriber_name>	Yes	Name of the subscriber for which the data is retrieved.
-CONTEXT=<context>	No	Context in which the logical schema will be resolved. If no context is specified, the execution context is used.
-MAX_JRN_DATE=<to_date>	No	Date (and time) until which the journalizing events are taken into account.

Examples

Retrieve for the `CUSTOMERS` table in the `SALES_APPLICATION` schema the journalizing events for the `SALES_SYNC` subscriber.

```
OdiRetrieveJournalData -LSCHEMA=SALES_APPLICATION
-TABLE_NAME=CUSTOMERS -SUBSCRIBER_NAME=SALES_SYNC
```

OdiReverseGetMetaData

Use this command to reverse-engineer metadata for the given model in the reverse tables using the JDBC driver capabilities. This command is typically preceded by OdiReverseResetTable and followed by OdiReverseSetMetaData.

> **Note:**
>
> - This command uses the same technique as the standard reverse-engineering, and depends on the capabilities of the JDBC driver used.
> - The use of this command is restricted to DEVELOPMENT type Repositories because the metadata is not available on EXECUTION type Repositories.

Usage

```
OdiReverseGetMetaData -MODEL=<model_id>
```

Parameters

Parameters	Mandatory	Description
`-MODEL=<model_id>`	Yes	Model to reverse-engineer.

Examples

Reverse the RKM's current model.

```
OdiReverseGetMetaData -MODEL=<%=odiRef.getModel("ID")%>
```

OdiReverseManageShortcut

Use this command to define how to handle shortcuts when they are reverse-engineered in a model.

Usage

```
OdiReverseManageShortcut "-MODEL=<model_id>" "-MODE=MATERIALIZING_MODE"
```

Parameters

Parameters	Mandatory	Description
`-MODEL=<model_id>`	Yes	Global identifier of the model to be reversed.

Parameters	Mandatory	Description
-MODE=ALWAYS_MATERIALIZE|ALWAYS_SKIP|PROMPT	Yes	This parameter is supported only when a package or scenario is run in ODI Studio. This parameter accepts the following values: • ALWAYS_MATERIALIZE: Conflicted shortcuts are always materialized and datastores are reversed (default). • ALWAYS_SKIP: Conflicted shortcuts are always skipped and not reversed. • PROMPT: The **Shortcut Conflict Detected** dialog is displayed. You can define how to handle conflicted shortcuts. Select **Materialize**, to materialize and reverse-engineer the conflicted datastore shortcut. Leave **Materialize** unselected, to skip the conflicted shortcuts. Unselected datastores are not reversed and the shortcut remains.

Examples

Reverse model 44fa5543-a378-4442-ac64-3dabab65ef98 in ALWAYS_MATERIALIZE mode.

```
OdiReverseManageShortcut -MODEL=44fa5543-a378-4442-ac64-3dabab65ef98 -
MODE=ALWAYS_MATERIALIZE
```

OdiReverseResetTable

Use this command to reset the content of reverse tables for a given model. This command is typically used at the beginning of a customized reverse-engineering process.

Usage

```
OdiReverseResetTable -MODEL=<model_id>
```

Parameters

Parameters	Mandatory	Description
-MODEL=<model_id>	Yes	Global identifier of the model to be reversed.

Examples

```
OdiReverseResetTable -MODEL=44fa5543-a378-4442-ac64-3dabab65ef98
```

OdiReverseSetMetaData

Use this command to integrate metadata from the reverse tables into the Repository for a given data model.

Usage

```
OdiReverseSetMetaData -MODEL=<model_id> [-USE_TABLE_NAME_FOR_UPDATE=<true|false>]
```

Parameters

Parameters	Mandatory	Description
`-MODEL=<model_id>`	Yes	Global identifier of the model to be reversed.
`-USE_TABLE_NAME_FOR_UPDATE=<true|false>`	No	• If `true`, the `TABLE_NAME` is used as an update key on the target tables. • If `false` (default), the `RES_NAME` is used as the update key on the target tables.

Example

Reverse model `125880`, using the `TABLE_NAME` as an update key on the target tables.

```
OdiReverseSetMetaData -MODEL=44fa5543-a378-4442-ac64-3dabab65ef98 -
USE_TABLE_NAME_FOR_UPDATE=true
```

OdiRollbackDeploymentArchive

Use this command to rollback a Patch Deployment Archive (DA) from an ODI repository.

Usage

```
OdiRollbackDeploymentArchive -ROLLBACK_FILE_NAME=<rollback_file_name>
[-APPLY_WITHOUT_CIPHER_DATA=<yes|no>] [-EXPORT_KEY=<Export_Key>]
```

Parameters

Parameters	Mandatory	Description
`ROLLBACK_FILE_NAME=<rollback_file_name>`	Yes	Complete file name of the rollback deployment archive.
`APPLY_WITHOUT_CIPHER_DATA=<yes|no>`	No[1]	If set to Yes, any cipher data present in the deployment archive will be made null. If set to No, the export key will be used to migrate the cipher data. The default value is No.
`EXPORT_KEY=<Export_Key>`	No	Specifies a cryptographic private key used to migrate cipher data in the deployment archive objects.

[1] If the APPLY_WITHOUT_CIPHER_DATA parameter is set to No, the EXPORT_KEY parameter must be specified.

Examples

Rollback the last applied patch deployment archive with export key.

```
OdiRollbackDeploymentArchive -ROLLBACK_FILE_NAME=rollback_file_name
-APPLY_WITHOUT_CIPHER_DATA=no -EXPORT_KEY=Export_Key
```

OdiSAPALEClient and OdiSAPALEClient3

Use this command to generate SAP Internal Documents (IDoc) from XML source files and transfer these IDocs using ALE (Application Link Enabling) to a remote tRFC server (SAP R/3 server).

> **Note:**
>
> The OdiSAPALEClient tool supports SAP Java Connector 2.x. To use the SAP Java Connectors 3.x, use the OdiSAPALEClient3 tool.

Usage

```
OdiSAPALEClient -USER=<sap_logon> -ENCODED_PASSWORD=<password>
-GATEWAYHOST=<gateway_host> -SYSTEMNR=<system_number>
-MESSAGESERVERHOST=<message_server> -R3NAME=<system_name>
-APPLICATIONSERVERSGROUP=<group_name>
[-DIR=<directory>] [-FILE=<file>] [-CASESENS=<yes|no>]
[-MOVEDIR=<target_directory>] [-DELETE=<yes|no>] [-POOL_KEY=<pool_key>]
[-LANGUAGE=<language>] [-CLIENT=<client>] [-MAX_CONNECTIONS=<n>]
[-TRACE=<no|yes>]
```

Usage for OdiSAPALEClient3

```
OdiSAPALEClient3 -USER=<sap_logon> -ENCODED_PASSWORD=<password>
-GATEWAYHOST=<gateway_host> -SYSTEMNR=<system_number>
-MESSAGESERVERHOST=<message_server> -R3NAME=<system_name>
-APPLICATIONSERVERSGROUP=<group_name>
[-DIR=<directory>] [-FILE=<file>] [-CASESENS=<yes|no>]
[-MOVEDIR=<target_directory>] [-DELETE=<yes|no>] [-POOL_KEY=<pool_key>]
[-LANGUAGE=<language>] [-CLIENT=<client>] [-MAX_CONNECTIONS=<n>]
[-TRACE=<no|yes>]
```

Parameter

Parameters	Mandatory	Description
-USER=<sap_logon>	Yes	SAP logon. This user may be a system user.
-PASSWORD=<password>	Deprecated	SAP logon password. This command is deprecated. Use -ENCODED_PASSWORD instead.

Parameters	Mandatory	Description
-ENCODED_PASSWORD=<password>	Yes	SAP logon password, encrypted. The OS command encode <password> can be used to encrypt this password.
-GATEWAYHOST=<gateway_host>	No	Gateway host, mandatory if -MESSAGESERVERHOST is not specified.
-SYSTEMNR=<system_number>	No	SAP system number, mandatory if -GATEWAYHOST is used. The SAP system number enables the SAP load balancing feature.
-MESSAGESERVERHOST=<message_server>	No	Message server host name, mandatory if -GATEWAYHOST is not specified. If -GATEWAYHOST and -MESSAGESERVERHOST are both specified, -MESSAGESERVERHOST is used.
-R3NAME=<system_name>	No	Name of the SAP system (r3name), mandatory if -MESSAGESERVERHOST is used.
-APPLICATIONSERVERSGROUP=<group_name>	No	Application servers group name, mandatory if -MESSAGESERVERHOST is used.
-DIR=<directory>	No	XML source file directory. This parameter is taken into account if -FILE is not specified. At least one of the -DIR or -FILE parameters must be specified.
-FILE=<file>	No	Name of the source XML file. If this parameter is omitted, all files in -DIR are processed. At least one of the -DIR or -FILE parameters must be specified.
-CASESENS=<yes|no>	No	Indicates if the source file names are case-sensitive. The default value is No.
-MOVEDIR=<target_directory>	No	If this parameter is specified, the source files are moved to this directory after being processed.
-DELETE=<yes|no>	No	Deletes the source files after their processing. The default value is Yes.
-POOL_KEY=<pool_key>	No	Name of the connection pool. The default value is ODI.
-LANGUAGE=<language>	No	Language code used for error messages. The default value is EN.
-CLIENT=<client>	No	Client identifier. The default value is 001.
-MAX_CONNECTIONS=<n>	No	Maximum number of connections in the pool. The default value is 3.

Parameters	Mandatory	Description
-TRACE=<no|yes>	No	The generated IDoc files are archived in the source file directory. If the source files are moved (-MOVEDIR parameter), the generated IDocs are also moved. The default value is No.

Examples

Process all files in the /sap directory and send them as IDocs to the SAP server. The original XML and generated files are stored in the /log directory after processing.

```
OdiSAPALEClient -USER=ODI -ENCODED_PASSWORD=xxx -SYSTEMNR=002
-GATEWAYHOST=GW001 -DIR=/sap -MOVEDIR=/log -TRACE=yes
```

OdiSAPALEServer and OdiSAPALEServer3

Use this command to start a tRFC listener to receive SAP IDocs transferred using ALE (Application Link Enabling). This listener transforms incoming IDocs into XML files in a given directory.

> **Note:**
>
> The OdiSAPALEServer tool supports SAP Java Connector 2.x. To use the SAP Java Connectors 3.x, use the OdiSAPALEServer3 tool.

Usage

```
OdiSAPALEServer -USER=<sap_logon> -ENCODED_PASSWORD=<password>
-GATEWAYHOST=<gateway_host> -SYSTEMNR=<system_number>
-GATEWAYNAME=<gateway_name> -PROGRAMID=<program_id> -DIR=<target_directory>
[-TIMEOUT=<n>] [-POOL_KEY=<pool_key>] [-LANGUAGE=<Language>]
[-CLIENT=<client>] [-MAX_CONNECTIONS=<n>]
[-INTERREQUESTTIMEOUT=<n>] [-MAXREQUEST=<n>] [-TRACE=<no|yes>]
```

Usage of OdiSAPALEServer3

```
OdiSAPALEServer3 -USER=<sap_logon> -ENCODED_PASSWORD=<password>
-GATEWAYHOST=<gateway_host> -SYSTEMNR=<system_number>
-GATEWAYNAME=<gateway_name> -PROGRAMID=<program_id> -DIR=<target_directory>
[-TIMEOUT=<n>] [-POOL_KEY=<pool_key>] [-LANGUAGE=<Language>]
[-CLIENT=<client>] [-MAX_CONNECTIONS=<n>]
[-INTERREQUESTTIMEOUT=<n>] [-MAXREQUEST=<n>] [-TRACE=<no|yes>]
```

Parameters

Parameters	Mandatory	Description
-USER=<UserName>	Yes	SAP logon. This user may be a system user.

Parameters	Mandatory	Description
-ENCODED_PASSWORD=<password>	Yes	SAP logon password, encrypted. The system command encode <password> can be used to encrypt this password.
-GATEWAYHOST=<gateway_host>	Yes	Gateway host.
-SYSTEMNR=<system_number>	Yes	SAP system number.
-GATEWAYNAME=<gateway_name>	Yes	Gateway name.
-PROGRAMID=<program_id>	Yes	The program ID. External name used by the tRFC server.
-DIR=<target_directory>	Yes	Directory in which the target XML files are stored. These files are named <IDOC Number>.xml, and are located in subdirectories named after the IDoc type. The default is ./FromSAP.
-POOL_KEY=<pool_key>	Yes	Name of the connection pool. The default value is ODI.
-LANG=<language>	Yes	Language code used for error messages. The default value is EN.
-CLIENT=<client>	Yes	SAP client identifier. The default value is 001.
-TIMEOUT=<n>	No	Life span in milliseconds for the server. At the end of this period, the server stops automatically. If this timeout is set to 0, the server life span is infinite. The default value is 0.
-MAX_CONNECTIONS=<n>	Yes	Maximum number of connections allowed for the pool of connections. The default value is 3.
-INTERREQUESTTIMEOUT=<n>	No	If no IDOC is received during an interval of n milliseconds, the listener stops. If this timeout is set to 0, the timeout is infinite. The default value is 0.
-MAXREQUEST=<n>	No	Maximum number of requests after which the listener stops. If this parameter is set to 0, the server expects an infinite number of requests. The default value is 0. Note: If -TIMEOUT, -INTERREQUESTTIMEOUT, and -MAXREQUEST are set to 0 or left empty, then -MAXREQUEST automatically takes the value 1.
-TRACE=<no|yes>	No	Activate the debug trace. The default value is No.

Parameters	Mandatory	Description
-	No	Must match the RFC destination in SAP. Verify that the Unicode setting in SAP transaction SM59 matches this parameter. Note: Applies to OdiSAPALEServer3 only.

Examples

Wait for 2 IDoc files and generate the target XML files in the `/temp` directory.

```
OdiSAPALEServer -POOL_KEY=ODI -MAX_CONNECTIONS=3 -CLIENT=001
-USER=ODI -ENCODED_PASSWORD=xxx -LANGUAGE=EN
-GATEWAYHOST=SAP001 -SYSTEMNR=002 -GATEWAYNAME=GW001
-PROGRAMID=ODI01 -DIR=/tmp -MAXREQUEST=2
```

OdiScpGet

Use this command to download a file from an SSH server.

Usage

```
OdiScpGet -HOST=<ssh server host name> -USER=<ssh user>
[-PASSWORD=<ssh user password>] -REMOTE_DIR=<remote dir on ssh host>
[-REMOTE_FILE=<file name under the REMOTE_DIR>] -LOCAL_DIR=<local dir>
[-LOCAL_FILE=<file name under the LOCAL_DIR>]
[-TIMEOUT=<time in seconds>]
[-IDENTITY_FILE=<full path to the private key file of the user>]
[-KNOWNHOSTS_FILE=<full path to known hosts file>] [COMPRESSION=<yes|no>]
[-STRICT_HOSTKEY_CHECKING=<yes|no>] [-PROXY_HOST=<proxy server host name>]
[-PROXY_PORT=<proxy server port>] [-PROXY_TYPE=<HTTP|SOCKS5>]
```

Parameters

Parameters	Mandatory	Description
`-HOST=<ssh server host name>`	Yes	Host name of the SSH server.
`-USER=<ssh user>`	Yes	User on the SSH server.
`-PASSWORD=<ssh user password>`	No	The password of the SSH user or the passphrase of the password-protected identity file. If the `-IDENTITY_FILE` argument is provided, this value is used as the passphrase for the password-protected private key file. If public key authentication fails, it falls back to the normal user password authentication.
`-REMOTE_DIR=<dir on remote SSH>`	Yes	Directory path on the remote SSH host.

Parameters	Mandatory	Description
-REMOTE_FILE=<file name under -REMOTE DIR>	No	File name under the directory specified in the -REMOTE_DIR argument. Note that all subdirectories matching the remote file name will also be transferred to the local folder. If this argument is missing, the file is copied with the -LOCAL_FILE file name. If -LOCAL_FILE is also missing, the -LOCAL_DIR is copied recursively to the -REMOTE_DIR.
-LOCAL_DIR=<local dir path>	Yes	Directory path on the local machine.
-LOCAL_FILE=<local file>	No	File name under the directory specified in the -LOCAL_DIR argument. If this argument is missing, all files and directories under -LOCAL_DIR are copied recursively to the -REMOTE_DIR. To filter the files to be copied, use * to specify the generic characters. Examples: • *.log (all files with the log extension) • arch_*.lst (all files starting with arch_ and with the extension lst)
-IDENTITY_FILE=<full path to the private key file of the user>	No	Private key file of the local user. If this argument is specified, public key authentication is performed. The -PASSWORD argument is used as the password for the password-protected private key file. If authentication fails, it falls back to normal user password authentication.
-KNOWNHOSTS_FILE=<full path to the known hosts file on the local machine>	No	Full path to the known hosts file on the local machine. The known hosts file contains the host keys of all remote machines that the user trusts. If this argument is missing, the <user home dir>/.ssh/known_hosts file is used as the known hosts file if it exists.
-COMPRESSION=<yes|no>	No	If set to Yes, data compression is used. The default value is No.
-STRICT_HOSTKEY_CHECKING=<yes|no>	No	If set to Yes (default), strict host key checking is performed and authentication fails if the remote SSH host key is not present in the known hosts file specified in -KNOWNHOSTS_FILE.

Parameters	Mandatory	Description
-PROXY_HOST=<proxy server host name>	No	Host name of the proxy server to be used for the connection.
-PROXY_PORT=<proxy server port>	No	Port number of the proxy server.
-PROXY_TYPE=<HTTP|SOCKS5>	No	Type of proxy server you are connecting to, HTTP or SOCKS5.
-TIMEOUT=<time in seconds>	No	Time in seconds after which the socket connection times out.

Examples

Copy the remote directory `/test_copy555` on the SSH server recursively to the local directory `C:\temp\test_copy`.

```
OdiScpGet -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password>
 -LOCAL_DIR=C:\temp\test_copy -REMOTE_DIR=/test_copy555
```

Copy all files matching the `Sales*.txt` pattern under the remote directory `/` on the SSH server to the local directory `C:\temp\`.

```
OdiScpGet -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password>
 -LOCAL_DIR=C:\temp -REMOTE_FILE=Sales*.txt -REMOTE_DIR=/
```

Copy the `Sales1.txt` file under the remote directory `/` on the SSH server to the local directory `C:\temp\` as a `Sample1.txt` file.

```
OdiScpGet -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password>
 -REMOTE_DIR=/ REMOTE_FILE=Sales1.txt -LOCAL_DIR=C:\temp
-LOCAL_FILE=Sample1.txt
```

Copy the `Sales1.txt` file under the remote directory `/` on the SSH server to the local directory `C:\temp\` as a `Sample1.txt` file. Public key authentication is performed by providing the path to the identity file and the path to the known hosts file.

```
OdiScpGet -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password>
-REMOTE_DIR=/ -REMOTE_FILE=Sales1.txt -LOCAL_DIR=C:\temp
-LOCAL_FILE=Sample1.txt -IDENTITY_FILE=C:\Documents and
Settings\username\.ssh\id_dsa -KNOWNHOSTS_FILE=C:\Documents and
Settings\username\.ssh\known_hosts
```

Copy the `Sales1.txt` file under the remote directory `/` on the SSH server to the local directory `C:\temp\` as a `Sample1.txt` file. Public key authentication is performed by providing the path to the identity file. All hosts are trusted by passing the No value to the `-STRICT_HOSTKEY_CHECKING` parameter.

```
OdiScpGet -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password>
-REMOTE_DIR=/ -REMOTE_FILE=Sales1.txt -LOCAL_DIR=C:\temp -LOCAL_FILE=Sample1.txt
-IDENTITY_FILE=C:\Documents and Settings\username\.ssh\id_dsa
-STRICT_HOSTKEY_CHECKING=NO
```

OdiScpPut

Use this command to upload a file to an SSH server.

Usage

```
OdiScpPut -HOST=<SSH server host name> -USER=<SSH user>
[-PASSWORD=<SSH user password>] -LOCAL_DIR=<local dir>
[-LOCAL_FILE=<file name under the LOCAL_DIR>] -REMOTE_DIR=<remote dir on ssh host>
[-REMOTE_FILE=<file name under the REMOTE_DIR>]
[-TIMEOUT=<time in seconds>]
[-IDENTITY_FILE=<full path to the private key file of the user>]
[-KNOWNHOSTS_FILE=<full path to known hosts file>] [-COMPRESSION=<yes|no>]
[-STRICT_HOSTKEY_CHECKING=<yes|no>] [<-PROXY_HOST=<proxy server host name>]
[-PROXY_PORT=<proxy server port>] [-PROXY_TYPE=<HTTP|SOCKS5>]
```

Parameters

Parameters	Mandatory	Description
-HOST=<host name of the SSH server>	Yes	Host name of the SSH server.
-USER=<host name of the SSH user>	Yes	User on the SSH server.
-PASSWORD=<password of the SSH user>	No	Password of the SSH user or the passphrase of the password-protected identity file. If the -IDENTITY_FILE argument is provided, this value is used as the passphrase for the password-protected private key file. If public key authentication fails, it falls back to the normal user password authentication.
-REMOTE_DIR=<dir on remote SSH	Yes	Directory path on the remote SSH host.
-REMOTE_FILE=<file name under -REMOTE DIR>	No	File name under the directory specified in the -REMOTE_DIR argument. If this argument is missing, the file is copied with the -LOCAL_FILE file name. If the -LOCAL_FILE argument is also missing, the -LOCAL_DIR is copied recursively to the -REMOTE_DIR.
-LOCAL_DIR=<local dir path>	Yes	Directory path on the local machine.

Parameters	Mandatory	Description
-LOCAL_FILE=<local file>	No	File name under the directory specified in the -LOCAL_DIR argument. If this argument is missing, all files and directories under the -LOCAL_DIR are copied recursively to the -REMOTE_DIR. To filter the files to be copied, use * to specify the generic characters. Examples: • *.log (all files with the log extension) • arch_*.lst (all files starting with arch_ and with the extension lst)
-IDENTITY_FILE=<full path to the private key file of the user>	No	Private key file of the local user. If this argument is specified, public key authentication is performed. The -PASSWORD argument is used as the password for the password-protected private key file. If authentication fails, it falls back to normal user password authentication.
-KNOWNHOSTS_FILE=<full path to the known hosts file on the local machine>	No	Full path to the known hosts file on the local machine. The known hosts file contains the host keys of all remote machines the user trusts. If this argument is missing, the <user home dir>/.ssh/known_hosts file is used as the known hosts file if it exists.
-COMPRESSION=<yes|no>	No	If set to Yes, data compression is used. The default value is No.
-STRICT_HOSTKEY_CHECKING=<yes|no>	No	If set to Yes (default), strict host key checking is performed and authentication fails if the remote SSH host key is not present in the known hosts file specified in -KNOWNHOSTS_FILE.
-PROXY_HOST=<proxy server host name>	No	Host name of the proxy server to be used for the connection.
-PROXY_PORT=<proxy server port>	No	Port number of the proxy server.
-PROXY_TYPE=<HTTP|SOCKS5>	No	Type of proxy server you are connecting to, HTTP or SOCKS5.
-TIMEOUT=<timeout value>	No	Time in seconds after which the socket connection times out.

Examples

Copy the local directory C:\temp\test_copy recursively to the remote directory /test_copy555 on the SSH server.

```
OdiScpPut -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password> -
LOCAL_DIR=C:\temp\test_copy -REMOTE_DIR=/test_copy555
```

Copy all files matching the `Sales*.txt` pattern under the local directory `C:\temp\` to the remote directory `/` on the SSH server.

```
OdiScpPut -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password> -
LOCAL_DIR=C:\temp -LOCAL_FILE=Sales*.txt -REMOTE_DIR=/
```

Copy the `Sales1.txt` file under the local directory `C:\temp\` to the remote directory `/` on the SSH server as a `Sample1.txt` file.

```
OdiScpPut -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password>
-LOCAL_DIR=C:\temp -LOCAL_FILE=Sales1.txt -REMOTE_DIR=/ -REMOTE_FILE=Sample1.txt
```

Copy the `Sales1.txt` file under the local directory `C:\temp\` to the remote directory `/` on the SSH server as a `Sample1.txt` file. Public key authentication is performed by providing the path to the identity file and the path to the known hosts file.

```
OdiScpPut -HOST=machine.example.com -USER=test_ftp
-PASSWORD=<password> -LOCAL_DIR=C:\temp -LOCAL_FILE=Sales1.txt
-REMOTE_DIR=/ -REMOTE_FILE=Sample1.txt
-IDENTITY_FILE=C:\Documents and Settings\username\.ssh\id_dsa
-KNOWNHOSTS_FILE=C:\Documents and Settings\username\.ssh\known_hosts
```

Copy the `Sales1.txt` file under the local directory `C:\temp\` to the remote directory `/` on the SSH server as a `Sample1.txt` file. Public key authentication is performed by providing the path to the identity file. All hosts are trusted by passing the No value to the `-STRICT_HOSTKEY_CHECKING` parameter.

```
OdiScpPut -HOST=machine.example.com -USER=test_ftp
-PASSWORD=<password> -LOCAL_DIR=C:\temp -LOCAL_FILE=Sales1.txt
-REMOTE_DIR=/ -REMOTE_FILE=Sample1.txt
-IDENTITY_FILE=C:\Documents and Settings\username\.ssh\id_dsa
-STRICT_HOSTKEY_CHECKING=NO
```

OdiSendMail

Use this command to send an email to an SMTP server.

Usage

```
OdiSendMail -MAILHOST=<mail_host> -FROM=<from_user> -TO=<address_list>
[-CC=<address_list>] [-BCC=<address_list>] [-SUBJECT=<subject>]
[-ATTACH=<file_path>]* [-PORT=<PortNumber>] [-PROTOCOL=<MailProtocol>] [-AUTH=<Yes|
No>] [-AUTHMECHANISM=<MailAuthMechanism] [-USER=<Username>] [-PASS=<Password>] [-
MSGBODY=<message_body> | CR/LF<message_body>]
```

Parameters

Parameters	Mandatory	Description
`-MAILHOST=<mail_host>`	Yes	IP address of the SMTP server.

Parameters	Mandatory	Description
-FROM=<from_user>	Yes	Address of the sender of the message. Example: support@example.com To send the external name of the sender, the following notation can be used: "-FROM=Support center <support@example.com>"
-TO=<address_list>	Yes	List of email addresses of the recipients, separated by commas. Example: "-TO=sales@example.com, support@example.com"
-CC=<address_list>	No	List of e-mail addresses of the CC-ed recipients, separated by commas. Example: "-CC=info@example.com"
-BCC=<address_list>	No	List of email-addresses of the BCC-ed recipients, separated by commas. Example: "-BCC=manager@example.com"
-SUBJECT=<subject>	No	Object (subject) of the message.
-ATTACH=<file_path>	No	Path of the file to join to the message, relative to the execution agent. To join several files, repeat -ATTACH. Example: Attach the files .profile and .cshrc to the mail: -ATTACH=/home/usr/.profile -ATTACH=/home/usr/.cshrc
CR/LF <message_body> or -MSGBODY=<message_body>	No	Message body (text). This text can be typed on the line following the OdiSendMail command (a carriage return - CR/LF - indicates the beginning of the mail body), or can be defined with the -MSGBODY parameter. The -MSGBODY parameter should be used when calling this Oracle Data Integrator command from an OS command line.
-PORT	No	The Port number of the mail server. Default is @ default port used by javax.mail.
-PROTOCOL	No	E-mail protocol. It can be SMTP or POP3. Default is SMTP.
-AUTH	No	If authentication is to be used. The values are YES or NO. Default is NO.

Parameters	Mandatory	Description
-AUTHMECHANISMS	No	The authentication mechanism supported by the mail server. The values are PLAIN, LOGIN or DIGEST-MD5.
-USER	No	User for authentication. Only if authentication is used.
-PASS	No	Password for authentication. Only if authentication is used.

Examples

```
OdiSendMail -MAILHOST=mail.example.com "-FROM=Application Oracle Data
Integrator<odi@example.com>" -TO=admin@example.com "-SUBJECT=Execution OK"
-ATTACH=C:\log\job.log -ATTACH=C:\log\job.bad
Hello Administrator !
Your process finished successfully. Attached are your files.
Have a nice day!
Oracle Data Integrator.
```

OdiSftp

Use this command to connect to an SSH server with an enabled SFTP subsystem and perform standard FTP commands on the remote system. Trace from the script is recorded against the Execution Details of the task representing the OdiSftp step in Operator Navigator.

Usage

```
OdiSftp -HOST=<ssh server host name> -USER=<ssh user>
[-PASSWORD=<ssh user password>] -LOCAL_DIR=<local dir>
-REMOTE_DIR=<remote dir on ssh host>
[-TIMEOUT=<time in seconds>] [-IDENTITY_FILE=<full path to private key file of user>]
[-KNOWNHOSTS_FILE=<full path to known hosts file on local machine>]
[-COMPRESSION=<yes|no>] [-STRICT_HOSTKEY_CHECKING=<yes|no>]
[-PROXY_HOST=<proxy server host name>] [-PROXY_PORT=<proxy server port>]
 [-PROXY_TYPE=<HTTP|SOCKS5>] [STOP_ON_FTP_ERROR=<yes|no>]
-COMMAND=<command>
```

Parameters

Parameters	Mandatory	Description
-HOST=<ssh server host name>	Yes	Host name of the SSH server.
-USER=<ssh user>	Yes	User on the SSH server.
-PASSWORD=<ssh user password>	No	Password of the SSH user.
-LOCAL_DIR=<local dir>	Yes	Directory path on the local machine.
-REMOTE_DIR=<remote dir on ssh host>	Yes	Directory path on the remote SSH host.
-TIMEOUT=<time in seconds>	No	Time in seconds after which the socket connection times out.

Parameters	Mandatory	Description
-IDENTITY_FILE=<full path to private key file of user>	No	Private key file of the local user. If specified, public key authentication is performed. The -PASSWORD argument is used as the password for the password-protected private key file. If authentication fails, normal user password authentication is performed.
-KNOWNHOSTS_FILE=<full path to known hosts file on local machine>	No	Full path to the known hosts file on the local machine. The known hosts file contains host keys for all remote machines trusted by the user. If this argument is missing, the <user home dir>/.ssh/known_hosts file is used as the known hosts file if it exists.
-COMPRESSION=<yes|no>	No	If set to Yes, data compression is used. The default value is No.
-STRICT_HOSTKEY_CHECKING=<yes|no>	No	If set to Yes (default), strict host key checking is performed and authentication fails if the remote SSH host key is not present in the known hosts file specified in -KNOWNHOSTS_FILE.
-PROXY_HOST=<proxy server host name>	No	Host name of the proxy server to be used for the connection.
-PROXY_PORT=<proxy server port>	No	Port number of the proxy server.
-PROXY_TYPE<HTTP|SOCKS5>	No	Type of proxy server you are connecting to, HTTP or SOCKS5.
STOP_ON_FTP_ERROR=<yes|no>	No	If set to Yes (default), the step stops with an Error status if an error occurs rather than running to completion.
-COMMAND=<command>	Yes	Raw FTP command to execute. For a multiline command, pass the whole command as raw text after the OdiSftp line without the -COMMAND parameter. Supported commands: APPE, CDUP, CWD, DELE, LIST, MKD, NLST, PWD, QUIT, RETR, RMD, RNFR, RNTO, SIZE, STOR

Examples

Execute a script on a remote host that changes directory into a directory, deletes a file from the directory, changes directory into the parent directory, and removes the directory.

```
OdiSftp -HOST=machine.example.com -USER=odiftpuser -PASSWORD=<password>
-LOCAL_DIR=/tmp -REMOTE_DIR=/tmp -STOP_ON_FTP_ERROR=No
CWD /tmp/ftpToolDir1
```

```
DELE ftpToolFile
CDUP
RMD ftpToolDir1
```

OdiSftpGet

Use this command to download a file from an SSH server with an enabled SFTP subsystem.

Usage

```
OdiSftpGet -HOST=<ssh server host name> -USER=<ssh user>
[-PASSWORD=<ssh user password>] -REMOTE_DIR=<remote dir on ssh host>
[-REMOTE_FILE=<file name under REMOTE_DIR>] -LOCAL_DIR=<local dir>
[-LOCAL_FILE=<file name under LOCAL_DIR>]
[-TIMEOUT=<time in seconds>]
[-IDENTITY_FILE=<full path to private key file of user>]
[-KNOWNHOSTS_FILE=<full path to known hosts file on local machine>]
[-COMPRESSION=<yes|no>] [-STRICT_HOSTKEY_CHECKING=<yes|no>]
[-PROXY_HOST=<proxy server host name>] [-PROXY_PORT=<proxy server port>]
[-PROXY_TYPE=<HTTP|SOCKS5>]
```

> **Note:**
>
> If a Local or Remote file name needs to have % as part of its name, %25 needs to be passed instead of just %.
>
> %25 will resolve automatically to %.
>
> For example, if file name needs to be `temp%result`, it should be passed as `REMOTE_FILE=temp%25result` or `-LOCAL_FILE=temp%25result`.

Parameters

Parameters	Mandatory	Description
`-HOST=<ssh server host name>`	Yes	Host name of the SSH server. You can add the port number to the host name by prefixing it with a colon (`:`). For example: `machine.example.com:25` If no port is specified, port 22 is used by default.
`-USER=<ssh user>`	Yes	User on the SSH server.
`-PASSWORD=<ssh user password>`	No	Password of the SSH user.
`-REMOTE_DIR=<remote dir on ssh host>`	Yes	Directory path on the remote SSH host.

Parameters	Mandatory	Description
-REMOTE_FILE=<file name under -REMOTE DIR>	No	File name under the directory specified in the -REMOTE_DIR argument. If this argument is missing, the file is copied with the -LOCAL_FILE file name. If the -LOCAL_FILE argument is also missing, the -LOCAL_DIR is copied recursively to the -REMOTE_DIR.
-LOCAL_DIR=<local dir>	Yes	Directory path on the local machine.
-LOCAL_FILE=<file name under LOCAL_DIR>	No	File name under the directory specified in the -LOCAL_DIR argument. If this argument is missing, all files and directories under the -LOCAL_DIR are copied recursively to the -REMOTE_DIR. To filter the files to be copied, use * to specify the generic characters. Examples: • *.log (all files with the log extension) • arch_*.lst (all files starting with arch_ and with the extension lst)
-IDENTITY_FILE=<full path to private key file of user>	No	Private key file of the local user. If this argument is specified, public key authentication is performed. The -PASSWORD argument is used as the password for the password-protected private key file. If authentication fails, it falls back to normal user password authentication.
-KNOWNHOSTS_FILE=<full path to known hosts file on local machine>	No	The full path to the known hosts file on the local machine. The known hosts file contains the host keys of all remote machines the user trusts. If this argument is missing, the <user home dir>/.ssh/known_hosts file is used as the known hosts file if it exists.
-COMPRESSION=<yes|no>	No	If set to Yes, data compression is used. The default value is No.
-STRICT_HOSTKEY_CHECKING=<yes|no>	No	If set to Yes (default), strict host key checking is performed and authentication fails if the remote SSH host key is not present in the known hosts file specified in -KNOWNHOSTS_FILE.
-PROXY_HOST=<proxy server host name>	No	Host name of the proxy server to be used for the connection.
-PROXY_PORT=<proxy server port>	No	Port number of the proxy server.

Parameters	Mandatory	Description
-PROXY_TYPE=<HTTP|SOCKS5>	No	Type of proxy server you are connecting to, HTTP or SOCKS5.
-TIMEOUT=<time in seconds>	No	Time in seconds after which the socket connection times out.

Examples

Copy the remote directory /test_copy555 on the SSH server recursively to the local directory C:\temp\test_copy.

```
OdiSftpGet -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password> -
LOCAL_DIR=C:\temp\test_copy -REMOTE_DIR=/test_copy555
```

Copy all files matching the Sales*.txt pattern under the remote directory / on the SSH server to the local directory C:\temp\.

```
OdiSftpGet -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password> -
LOCAL_DIR=C:\temp -REMOTE_FILE=Sales*.txt -REMOTE_DIR=/
```

Copy the Sales1.txt file under the remote directory / on the SSH server to the local directory C:\temp\ as a Sample1.txt file.

```
OdiSftpGet -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password> -REMOTE_DIR=/
-LOCAL_FILE=Sales1.txt -LOCAL_DIR=C:\temp -LOCAL_FILE=Sample1.txt
```

Copy the Sales1.txt file under the remote directory / on the SSH server to the local directory C:\temp\ as a Sample1.txt file. Public key authentication is performed by providing the path to the identity file and the path to the known hosts file.

```
OdiSftpGet -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password>
-REMOTE_DIR=/ -REMOTE_FILE=Sales1.txt -LOCAL_DIR=C:\temp -LOCAL_FILE=Sample1.txt
-IDENTITY_FILE=C:\Documents and Settings\username\.ssh\id_dsa
-KNOWNHOSTS_FILE=C:\Documents and Settings\username\.ssh\known_hosts
```

Copy the Sales1.txt file under the remote directory / on the SSH server to the local directory C:\temp\ as a Sample1.txt file. Public key authentication is performed by providing the path to the identity file. All hosts are trusted by passing the No value to the -STRICT_HOSTKEY_CHECKING parameter.

```
OdiSftpGet -HOST=dev3 -USER=test_ftp -PASSWORD=<password>
-REMOTE_DIR=/ -REMOTE_FILE=Sales1.txt -LOCAL_DIR=C:\temp -LOCAL_FILE=Sample1.txt
-IDENTITY_FILE=C:\Documents and Settings\username\.ssh\id_dsa
-STRICT_HOSTKEY_CHECKING=NO
```

OdiSftpPut

Use this command to upload a file to an SSH server with the SFTP subsystem enabled.

Usage

```
OdiSftpPut -HOST=<ssh server host name> -USER=<ssh user>
[-PASSWORD=<ssh user password>] -LOCAL_DIR=<local dir>
[-LOCAL_FILE=<file name under LOCAL_DIR>] -REMOTE_DIR=<remote dir on ssh host>
[-REMOTE_FILE=<file name under REMOTE_DIR>]
[-TIMEOUT=<time in seconds>]
```

```
[-IDENTITY_FILE=<full path to private key file of user>]
[-KNOWNHOSTS_FILE=<full path to known hosts file on local machine>]
[-COMPRESSION=<yes|no>] [-STRICT_HOSTKEY_CHECKING=<yes|no>]
[-PROXY_HOST=<proxy server host name>] [-PROXY_PORT=<proxy server port>]
[-PROXY_TYPE=<HTTP|SOCKS5>]
```

> **Note:**
>
> If a Local or Remote file name needs to have % as part of its name, %25 needs to be passed instead of just %.
>
> %25 will resolve automatically to %.
>
> For example, if file name needs to be `temp%result`, it should be passed as `REMOTE_FILE=temp%25result` or `-LOCAL_FILE=temp%25result`.

Parameter

Parameters	Mandatory	Description
`-HOST=<ssh server host name>`	Yes	Host name of the SSH server. You can add the port number to the host name by prefixing it with a colon (`:`). For example: `machine.example.com:25` If no port is specified, port 22 is used by default.
`-USER=<ssh user>`	Yes	User on the SSH server.
`-PASSWORD=<ssh user password>`	No	Password of the SSH user or the passphrase of the password-protected identity file. If the `-IDENTITY_FILE` argument is provided, this value is used as the passphrase for the password-protected private key file. If public key authentication fails, it falls back to normal user password authentication.
`-REMOTE_DIR=<remote dir on ssh host`	Yes	Absolute directory path on the remote SSH host.
`-REMOTE_FILE=<file name under -REMOTE DIR>`	No	File name under the directory specified in the `-REMOTE_DIR` argument. If this argument is missing, the file is copied with the `-LOCAL_FILE` file name. If the `-LOCAL_FILE` argument is also missing, the `-LOCAL_DIR` is copied recursively to the `-REMOTE_DIR`.
`-LOCAL_DIR=<local dir>`	Yes	Directory path on the local machine.

Parameters	Mandatory	Description
-LOCAL_FILE=<file name under LOCAL_DIR>	No	File name under the directory specified in the -LOCAL_DIR argument. If this argument is missing, all files and directories under the -LOCAL_DIR are copied recursively to the -REMOTE_DIR. To filter the files to be copied, use * to specify the generic characters. Examples: • *.log (all files with the log extension) • arch_*.lst (all files starting with arch_ and with the extension lst)
-IDENTITY_FILE=<full path to private key file of user>	No	Private key file of the local user. If this argument is specified, public key authentication is performed. The -PASSWORD argument is used as the password for the password-protected private key file. If authentication fails, it falls back to normal user password authentication.
-KNOWNHOSTS_FILE=<full path to known hosts file on local machine>	No	Full path to the known hosts file on the local machine. The known hosts file contains the host keys of all remote machines the user trusts. If this argument is missing, the <user home dir>/.ssh/known_hosts file is used as the known hosts file if it exists.
-COMPRESSION=<yes|no>	No	If set to Yes, data compression is used. The default value is No.
-STRICT_HOSTKEY_CHECKING=<yes|no>	No	If set to Yes (default), strict host key checking is performed and authentication fails if the remote SSH host key is not present in the known hosts file specified in -KNOWNHOSTS_FILE.
-PROXY_HOST=<proxy server host name>	No	Host name of the proxy server to be used for the connection.
-PROXY_PORT=<proxy server port>	No	Port number of the proxy server.
-PROXY_TYPE=<HTTP|SOCKS5>	No	Type of proxy server you are connecting to, HTTP or SOCKS5.
-TIMEOUT=<time in seconds>	No	Time in seconds after which the socket connection times out.

Examples

Copy the local directory C:\temp\test_copy recursively to the remote directory /test_copy555 on the SSH server.

```
OdiSftpPut -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password> -
LOCAL_DIR=C:\temp\test_copy -REMOTE_DIR=/test_copy555
```

Copy all files matching the `Sales*.txt` pattern under the local directory `C:\temp\` to the remote directory `/` on the SSH server.

```
OdiSftpPut -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password> -
LOCAL_DIR=C:\temp -LOCAL_FILE=Sales*.txt -REMOTE_DIR=/
```

Copy the `Sales1.txt` file under the local directory `C:\temp\` to the remote directory `/` on the SSH server as a `Sample1.txt` file.

```
OdiSftpPut -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password> -
LOCAL_DIR=C:\temp -LOCAL_FILE=Sales1.txt -REMOTE_DIR=/Sample1.txt
```

Copy the `Sales1.txt` file under the local directory `C:\temp\` to the remote directory `/` on the SSH server as a `Sample1.txt` file. Public key authentication is performed by providing the path to the identity file and the path to the known hosts file.

```
OdiSftpPut -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password>
-LOCAL_DIR=C:\temp -LOCAL_FILE=Sales1.txt -REMOTE_DIR=/Sample1.txt
-IDENTITY_FILE=C:\Documents and Settings\username\.ssh\id_dsa
-KNOWNHOSTS_FILE=C:\Documents and Settings\username\.ssh\known_hosts
```

Copy the `Sales1.txt` file under the local directory `C:\temp\` to the remote directory `/` on the SSH server as a `Sample1.txt` file. Public key authentication is performed by providing the path to the identity file. All hosts are trusted by passing the No value to the `-STRICT_HOSTKEY_CHECKING` parameter.

```
OdiSftpPut -HOST=machine.example.com -USER=test_ftp -PASSWORD=<password>
-LOCAL_DIR=C:\temp -LOCAL_FILE=Sales1.txt -REMOTE_DIR=/Sample1.txt
-IDENTITY_FILE=C:\Documents and Settings\username\.ssh\id_dsa
-STRICT_HOSTKEY_CHECKING=NO
```

OdiSleep

Use this command to wait for `<delay>` milliseconds.

Usage

```
OdiSleep -DELAY=<delay>
```

Parameters

Parameters	Mandatory	Description
`-DELAY=<delay>`	Yes	Number of milliseconds to wait.

Examples

```
OdiSleep -DELAY=5000
```

OdiSqlUnload

Use this command to write the result of a SQL query to a file.

This command executes the SQL query <sql_query> on the data server whose connection parameters are provided by <driver>, <url>, <user>, and <encoded_pass>. The resulting resultset is written to <file_name>.

Usage

```
OdiSqlUnload -FILE=<file_name> -DRIVER=<driver> -URL=<url> -USER=<user>
-PASS=<password> [-FILE_FORMAT=<file_format>] [-FIELD_SEP=<field_sep> |
-XFIELD_SEP=<field_sep>] [-ROW_SEP=<row_sep> | -XROW_SEP=<row_sep>]
[-DATE_FORMAT=<date_format>] [-TIME_FORMAT=<time_format>][-CHARSET_ENCODING=<encoding>]
[-XML_CHARSET_ENCODING=<encoding>] [-FETCH_SIZE=<array_fetch_size>]
( CR/LF <sql_query> | -QUERY=<sql_query> | -QUERY_FILE=<sql_query_file> )
```

Parameters

Parameters	Mandatory	Description
-FILE=<file_name>	Yes	Full path to the output file, relative to the execution agent.
-DRIVER=<driver>	Yes	Name of the JDBC driver used to connect to the data server.
-URL=<url>	Yes	JDBC URL to the data server.
-USER=<user>	Yes	Login of the user on the data server that will be used to run the SQL query.
-PASS=<password>	Yes	Encrypted password for the login to the data server. This password can be encrypted with the system command encode <clear_text_password>. Note that agent(.bat or .sh) is located in the /bin subdirectory of your Oracle Data Integrator installation directory.

Parameters	Mandatory	Description
-FILE_FORMAT=<file_format>	No	Specifies the file format with one of the following three values: • fixed: Fixed size recording • variable: Variable size recording • xml: XML file If <file_format> is not specified, the format defaults to variable. If <file_format> is xml, the XML nodes generated have the following structure: <TABLE> <ROW> <column_name>![CDATA[VALUE]]</column_name> <column_name>![CDATA[VALUE]]</column_name> ... </ROW> .... </TABLE>
-FIELD_SEP=<field_sep>	No	Field separator character in ASCII format if -FILE_FORMAT=variable. The default <field_sep> is a tab character.
-XFIELD_SEP=<field_sep>	No	Field separator character in hexadecimal format if -FILE_FORMAT=variable. The default <field_sep> is a tab character.
-ROW_SEP=<row_sep>	No	Record separator character in ASCII format. The default <row_sep> is a Windows carriage return. For instance, the following values can be used: • UNIX: -ROW_SEP=\n • Windows: -ROW_SEP=\r\n
-XROW_SEP=<row_sep>	No	Record separator character in hexadecimal format. Example: 0A.
-DATE_FORMAT=<date_format>	No	Output format used for date datatypes. This date format is specified using the Java date and time format patterns.

Parameters	Mandatory	Description
-TIME_FORMAT=<time_format>	No	Output format used for time datatypes. For Example — OdiSqlUnload -FILE=C:\temp\clients.csv -DRIVER=sun.jdbc.odbc.JdbcOdbcDriver -URL=jdbc:odbc:NORTHWIND_ODBC -USER=sa -PASS=NFNEKKNGGJHAHBHDHEHJDBGBGFDGGH -FIELD_SEP=; "-DATE_FORMAT=dd/MM/yyyy" "-TIME_FORMAT=hh:mm:ss"
-CHARSET_ENCODING=<encoding>	No	Target file encoding. The default value is ISO-8859-1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html
-XML_CHARSET_ENCODING=<encoding>	No	Encoding specified in the XML file, in the tag <?xml version="1.0" encoding="ISO-8859-1"?>. The default value is ISO-8859-1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html
-FETCH_SIZE=<array_fetch_size>	No	Number of rows (records read) requested by Oracle Data Integrator in each communication with the data server.
-CR/LF=<sql_query>|-QUERY=<sql_query>|-QUERY_FILE=<sql_query_file>	Yes	SQL query to execute on the data server. The query must be a SELECT statement or a call to a stored procedure returning a valid recordset. This query can be entered on the line following the OdiSqlUnload command (a carriage return - CR/LF - indicates the beginning of the query). The query can be provided within the -QUERY parameter, or stored in a file specified with the -QUERY_FILE parameter. The -QUERY or -QUERY_FILE parameters must be used when calling this command from an OS command line.

Examples

Generate the file C:\temp\clients.csv separated by ; containing the result of the query on the Customers table.

```
OdiSqlUnload -FILE=C:\temp\clients.csv -DRIVER=sun.jdbc.odbc.JdbcOdbcDriver
-URL=jdbc:odbc:NORTHWIND_ODBC -USER=sa
-PASS=NFNEKKNGGJHAHBHDHEHJDBGBGFDGGH -FIELD_SEP=;
"-DATE_FORMAT=dd/MM/yyyy"
"-TIME_FORMAT=hh:mm:ss"
select cust_id, cust_name, cust_creation_date from Northwind.dbo.Customers
```

OdiStartLoadPlan

Use this command to start a Load Plan.

The -SYNC parameter starts a load plan in synchronous or asynchronous mode. In synchronous mode, the tool ends with the same status as the completed load plan run.

Usage

```
OdiStartLoadPlan -LOAD_PLAN_NAME=<load_plan_name> [-LOG_LEVEL=<log_level>]
[-CONTEXT=<context_code>] [-AGENT_URL=<agent_url>]
[-AGENT_CODE=<logical_agent_code>] [-ODI_USER=<ODI User>]
[-ODI_PASS=<ODI Password>] [-KEYWORDS=<Keywords>]
[-<PROJECT_CODE>.<VARIABLE>=<var_value> ...] [-SYNC=<yes|no>] [-POLLINT=<msec>]
```

Parameters

Parameters	Mandatory	Description
-LOAD_PLAN_NAME=<load_plan_name>	Yes	Name of the load plan to start.
-LOG_LEVEL=<log_level>	No	Level of logging information to retain. All sessions with a defined log level lower than or equal to this value are kept in the session log when the session completes. However, if object execution ends abnormally, all tasks are kept, regardless of this setting. Note that log level 6 has the same behavior as log level 5, but with the addition of variable and sequence tracking. See Tracking Variables and Sequences in *Developing Integration Projects with Oracle Data Integrator* for more information.
[-CONTEXT=<context_code>]	Yes	Code of the execution context. If this parameter is omitted, the load plan starts in the execution context of the calling session, if any.
[-AGENT_URL=<agent_url>]	No	URL of the remote agent that starts the load plan.
[-AGENT_CODE=<logical_agent_code>]	No	Code of the logical agent responsible for starting this load plan. If this parameter and -AGENT_URL are omitted, the current agent starts this load plan. This parameter is ignored if -AGENT_URL is specified.

Parameters	Mandatory	Description
[-ODI_USER=<ODI user>]	No	Oracle Data Integrator user to be used to start the load plan. The privileges of this user are used. If this parameter is omitted, the load plan is started with the privileges of the user launching the parent session.
[-ODI_PASS=<ODI Password>]	No	Password of the Oracle Data Integrator user. This password must be encoded. This parameter is required if -ODI_USER is specified.
-KEYWORDS=<keywords>	No	Comma-separated list of keywords attached to this load plan. These keywords make load plan execution identification easier.
-<VARIABLE>=<value>	No	List of project or global variables whose value is set as the default for the execution of the load plan. Project variables should be named <project_code>.<variable_name> and global variables should be named GLOBAL.<variable_name>. This list is of the form -<variable>=<value>.
-SYNC=<yes|no>	No	Specifies whether the load plan should be executed synchronously or asynchronously. If set to Yes (synchronous mode), the load plan is started and runs to completion with a status of Done or Error before control is returned. If set to No (asynchronous mode), the load plan is started and control is returned before the load plan runs to completion. The default value is No.
-POLLINT=<msec>	No	The time in milliseconds to wait between polling the load plan run status for completion state. The -SYNC parameter must be set to Yes. The default value is 1000 (1 second). The value must be greater than 0.

Examples

Start load plan LOAD_DWH in the GLOBAL context on the same agent.

```
OdiStartLoadPlan -LOAD_PLAN_NAME=LOAD_DWH -CONTEXT=GLOBAL
```

OdiStartOwbJob

Use this command to execute Oracle Warehouse Builder (OWB) objects from within Oracle Data Integrator and to retrieve the execution audit data into Oracle Data Integrator.

This command uses an Oracle Warehouse Builder runtime repository data server that can be created in Topology Navigator. This data server must connect as an Oracle Warehouse Builder user who can access an Oracle Warehouse Builder workspace. The physical schemas under this data server represent the Oracle Warehouse Builder workspaces that this user can access. For information about the Oracle Data Integrator topology, see Setting Up a Topology in *Administering Oracle Data Integrator*

Usage

```
OdiStartOwbJob -WORKSPACE=<logical_owb_repository> -LOCATION=<owb_location>
-OBJECT_NAME=<owb_object> -OBJECT_TYPE=<owb_object_type>
[-EXEC_PARAMS=<exec_params>] [-CONTEXT=<context_code>] [-LOG_LEVEL=<log_level>]
[-SYNC_MODE=<1|2>] [-POLLINT=<n>] [-SESSION_NAME=<session_name>]
[-KEYWORDS=<keywords>] [<OWB parameters>]
```

Parameters

Parameters	Mandatory	Description
-WORKSPACE=<logical_owb_repository>	Yes	Logical schema of the OWB Runtime Repository technology. This resolves to a physical schema that represents the Oracle Warehouse Builder workspace that contains the Oracle Warehouse Builder object to be executed. The Oracle Warehouse Builder workspace was chosen when you added a Physical Schema under the OWB Runtime Repository DataServer in Topology Navigator. The context for this mapping can also be specified using the -CONTEXT parameter.
-LOCATION=<owb_location>	Yes	Name of the Oracle Warehouse Builder location that contains the Oracle Warehouse Builder object to be executed. This location must exist in the physical workspace that resolves from -WORKSPACE.
-OBJECT_NAME=<owb_object>	Yes	Name of the Oracle Warehouse Builder object. This object must exist in -LOCATION.
-OBJECT_TYPE=<owb_object_type>	Yes	Type of Oracle Warehouse Builder object, for example: PLSQLMAP, PROCESSFLOW, SQLLOADERCONTROLFILE, MAPPING, DATAAUDITOR, ABAPFILE
-EXEC_PARAMS=<exec_params>	No	Custom and/or system parameters for the Oracle Warehouse Builder execution.

Parameters	Mandatory	Description
-CONTEXT=<context_code>	No	Execution context of the Oracle Warehouse Builder object. This is the context in which the logical workspace will be resolved. Studio editors use this value or the Default Context. Execution uses this value or the Parent Session context.
-LOG_LEVEL=<log_level>	No	Log level (0-5). The default value is 5, which means that maximum details are captured in the log.
-SYNC_MODE=<1|2>	No	Synchronization mode of the Oracle Warehouse Builder job: 1 - Synchronous (default). Execution of the session waits until the Oracle Warehouse Builder job terminates. 2 - Asynchronous. Execution of the session continues without waiting for the Oracle Warehouse Builder job to terminate.
-POLLINT=<n>	No	The period of time in milliseconds to wait between each transfer of Oracle Warehouse Builder audit data to Oracle Data Integrator log tables. The default value is 0, which means that audit data is transferred at the end of the execution.
-SESSION_NAME=<session_name>	No	Name of the Oracle Warehouse Builder session as it appears in the log.
-KEYWORDS=<keywords>	No	Comma-separated list of keywords attached to the session.
<OWB parameters>	No	List of values for the Oracle Warehouse Builder parameters relevant to the object. This list is of the form -PARAM_NAME=value. Oracle Warehouse Builder system parameters should be prefixed by OWB_SYSTEM, for example, OWB_SYSTEM.AUDIT_LEVEL.

Examples

Execute the Oracle Warehouse Builder process flow LOAD_USERS that has been deployed to the Oracle Workflow DEV_OWF.

```
OdiStartOwbJob -WORKSPACE=OWB_WS1 -CONTEXT=QA
-LOCATION=DEV_OWF -OBJECT_NAME=LOAD_USERS -OBJECT_TYPE=PROCESSFLOW
```

Execute the Oracle Warehouse Builder PL/SQL map STAGE_USERS that has been deployed to the database location DEV_STAGE. Poll and transfer the Oracle Warehouse Builder audit data every 5 seconds. Pass the input parameter AGE_LIMIT whose value is obtained from an Oracle Data Integrator variable, and specify an Oracle Warehouse Builder system parameter relevant to a PL/SQL map.

```
OdiStartOwbJob -WORKSPACE=OWB_WS1 -CONTEXT=QA
-LOCATION=DEV_STAGE -OBJECT_NAME=STAGE_USERS -OBJECT_TYPE=PLSQLMAP
-POLLINT=5000 -OWB_SYSTEM.MAX_NO_OF_ERRORS=25 -AGE_LIMIT=#VAR_MINAGE
```

OdiStartScen

Use this command to start a scenario.

The optional parameter `-AGENT_CODE` is used to dedicate this scenario to another agent other than the current agent.

The parameter `-SYNC_MODE` starts a scenario in synchronous or asynchronous mode.

> **Note:**
>
> The scenario that is started should be present in the repository into which the command is launched. If you go to production with a scenario, make sure to also take all scenarios called by your scenario using this command. The Solutions can help you group scenarios for this purpose.

Usage

```
OdiStartScen -SCEN_NAME=<scenario> -SCEN_VERSION=<version>
[-CONTEXT=<context>] [-ODI_USER=<odi user> -ODI_PASS=<odi password>]
[-SESSION_NAME=<session_name>] [-LOG_LEVEL=<log_level>]
[-AGENT_CODE=<logical_agent_name>] [-SYNC_MODE=<1|2>]
[-KEYWORDS=<keywords>] [-<VARIABLE>=<value>]*
```

Parameters

Parameters	Mandatory	Description
`-SCEN_NAME=<scenario>`	Yes	Name of the scenario to start.
`-SCEN_VERSION=<version>`	Yes	Version of the scenario to start. If the version specified is -1, the last version of the scenario is executed.
`-CONTEXT=<context>`	No	Code of the execution context. If this parameter is omitted, the scenario is executed in the execution context of the calling session.
`-ODI_USER=<odi user>`	No	Oracle Data Integrator user to be used to run the scenario. The privileges of this user are used. If this parameter is omitted, the scenario is executed with privileges of the user launching the parent session.
`-ODI_PASS=<odi password>`	No	Password of the Oracle Data Integrator user. This password should be encoded. This parameter is required if the user is specified.
`-SESSION_NAME=<session_name>`	No	Name of the session that will appear in the execution log.

Parameters	Mandatory	Description
-LOG_LEVEL=<log_level>	No	Trace level (0 .. 5) to keep in the execution log. The default value is 5.
-AGENT_CODE=<logical_agent_name>	No	Name of the logical agent responsible for executing this scenario. If this parameter is omitted, the current agent executes this scenario.
-SYNC_MODE=<1|2>	No	Synchronization mode of the scenario: 1 - Synchronous mode (default). The execution of the calling session is blocked until the scenario finishes its execution. 2 - Asynchronous mode. The execution of the calling session continues independently from the return of the called scenario.
-KEYWORDS=<keywords>	No	Comma-separated list of keywords attached to this session. These keywords make session identification easier.
-<VARIABLE>=<value>	No	List of variables whose value is set for the execution of the scenario. This list is of the form PROJECT.VARIABLE=value or GLOBAL.VARIABLE=value.

Examples

Start the scenario LOAD_DWH in version 2 in the production context (synchronous mode).

```
OdiStartScen -SCEN_NAME=LOAD_DWH -SCEN_VERSION=2
-CONTEXT=CTX_PRODUCTION
```

Start the scenario LOAD_DWH in version 2 in the current context in asynchronous mode on the agent UNIX Agent while passing the values of the variables START_DATE (local) and COMPANY_CODE (global).

```
OdiStartScen -SCEN_NAME=LOAD_DWH -SCEN_VERSION=2 -SYNC_MODE=2
"-AGENT_CODE=UNIX Agent" -MY_PROJECT.START_DATE=10-APR-2002
-GLOBAL.COMPANY_CODE=SP4356
```

OdiStorageCSDownload

Use this command to download single or multiple files or an entire directory to HDFS or a local file system from Oracle Storage Cloud Service. For HDFS files, the files are first copied

to the local directory (as specified by user in Directory — Work Schema) for Oracle Storage Cloud Service Physical Schema and then downloaded to the actual directory.

> **NOT_SUPPORTED:**
>
> This tool is applicable only for Data Integration Platform Cloud.

Usage

```
OdiStorageCSDownload
-SRC_LOGICAL_SCHEMA= <src_logical_schema>
-TRG_ LOGICAL_SCHEMA = <trg_logical_schema>
-FILE_NAMES_FILTER= <file_names_filter>
-OVERWRITE= Yes|No
-RETRY ON ERROR= <retry_number>
-RETRY_INTERVAL_SECONDS = <retry_interval_seconds>
-DECRYPT_KEY = <decrypt_key>
```

Table 2-1 Parameters

Parameter	Mandatory	Description
SRC_LOGICAL_SCHEMA	Yes	Name of the Source Logical schema name configured for Oracle Storage Cloud Service Data Server, which has information on Storage Cloud Service instance. Container information is obtained from Logical schema through configured physical schema.
TRG_LOGICAL_SCHEMA	Yes	Logical schema name configured for File of HDFS Data server for downloading files from Oracle Storage Cloud Service to Local file system of HDFS.
FILE_NAMES_FILTER	Yes	Field to specify one or more files to be downloaded from Oracle Storage CS recursively. It also supports delimiter | for separated files list. The pattern followed is: • *.txt - should download all files ending with .txt • test* - Downloads all the files and directories that matches with prefix “test” • *test* - Downloads all the files and directories having substring “test” • test.xml | test1.xml | test2.xml - Downloads all the files specified • test* | test1* - Downloads all the files matching pattern test* and test1* • test.xml - Only one file is downloaded.

Table 2-1 (Cont.) Parameters

Parameter	Mandatory	Description
OVERWRITE	No	This parameter indicates, if download operation should overwrite an existing file or not. The default value for this parameter is No.
RETRY_ON_ERROR	No	It represents the number of times the retry attempt should occur when a failure or error happens during download.
RETRY_INTERVAL_SECONDS	No	Retry interval indicates after how many seconds a retry attempt should happen.
DECRYPT_KEY	No	This is user provided key used for decrypting files/objects while downloading from Oracle Storage Cloud Service.

Note:

This parameter cannot be null, if you want to download objects which are encrypted using encrypt key.

Example

The following command is used to download files and directories from Oracle Storage Cloud Service:

```
OdiStorageCSDownload -SRC_LOGICAL_SCHEMA=src_logical_schema -
TRG_LOGICAL_SCHEMA=trg_logical_schema -FILE_NAMES_FILTER=file names filter
```

For Example:

```
OdiStorageCSDownload "-TRG_LOGICAL_SCHEMA=File_LS_Download" "-
SRC_LOGICAL_SCHEMA=StorageCS_LS" "-FILE_NAMES_FILTER=myfile.txt" "-
OVERWRITE=YES"
```

OdiStorageCSUpload

Use this tool to upload single or multiple files or an entire directory from HDFS or a local file system onto Oracle Storage Cloud Service.

> **NOT_SUPPORTED:**
>
> This tool is applicable only for Data Integration Platform Cloud.

Usage

```
OdiStorageCSUpload
-TRG_LOGICAL_SCHEMA = <trg_logical_schema>
-SRC_LOGICAL_SCHEMA = <src_logical_schema>
-FILE_NAMES_FILTER = <file_names_filter>
-OVERWRITE = Yes|No
-RETRY_ON_ERROR = <retry_number>
-RETRY_INTERVAL_SECONDS = <retry_interval_seconds>
-ENCRYPT_KEY = <encrypt_key>
```

Parameters

Table 2-2 Parameters

Parameters	Mandatory	Description
TRG_LOGICAL_SCHEMA	Yes	Target Logical schema name configured for Oracle Storage Cloud Service Data Server, which has information of Storage Cloud Service instance. Container information is obtained from Logical schema through configured physical schema.
SRC_LOGICAL_SCHEMA	Yes	Name of the Source Logical Schema configured for File of HDFS Data Server for uploading files from Local or HDFS to Oracle Storage Cloud Service.

Table 2-2 (Cont.) Parameters

Parameters	Mandatory	Description
FILE_NAMES_FILTER	Yes	Field to specify one or more files or directories to be uploaded to Oracle Storage Cloud Service recursively. It also supports the list of files separated by | as a delimiter. The pattern followed is: • *.txt - should upload all the files ending with .txt • test* - uploads all the files and directories that matches with prefix “test” • *test* - uploads all the files and directories having substring “test”. • test.xml | test1.xml | test2.xml - Uploads all the files specified. • test* | test1* - Uploads all the files matching pattern test* and test1* • test.xml - Only one file is uploaded.
OVERWRITE	No	This parameter indicates if upload operation should overwrite an existing file or not. Default value for this parameter is No.
RETRY_ON_ERROR	No	It represents the number of times the retry attempt should occur when a failure or error happens during upload.
RETRY_INTERVAL_SECONDS	No	Retry interval indicates after how many seconds a retry attempt should happen.

Table 2-2 (Cont.) Parameters

Parameters	Mandatory	Description
ENCRYPT_KEY	No	This is the user provided key used for encrypting files/objects while uploading files or directories to Oracle Storage Cloud Service. **Note:** This parameter cannot be null, if you want to encrypt objects while upload.

Example

The following command is used to upload file(s) and directories to Oracle Storage Cloud Service:

```
OdiStorageCSUpload -TRG_LOGICAL_SCHEMA =<trg_logical_schema> -
SRC_LOGICAL_SCHEMA = <src_logical_schema> -FILE_NAMES_FILTER=file name
filter
```

For Example-

```
OdiStorageCSUpload "-TRG_LOGICAL_SCHEMA=StorageCS_LS_Archive" "-
SRC_LOGICAL_SCHEMA=File_LS_Upload" "-FILE_NAMES_FILTER=myfile.txt" "-
OVERWRITE=YES"
```

OdiUnZip

Use this command to extract an archive file to a directory.

Usage

```
OdiUnZip -FILE=<file> -TODIR=<target_directory> [-OVERWRITE=<yes|no>]
[-ENCODING=<file_name_encoding>]
```

Parameters

Parameters	Mandatory	Description
-FILE=<file>	Yes	Full path to the ZIP file to extract.
-TODIR=<target_file>	Yes	Destination directory or folder.

Parameters	Mandatory	Description
-OVERWRITE=<yes|no>	No	Indicates if the files that already exist in the target directory must be overwritten. The default value is No.
-ENCODING=<file_name_encoding>	No	Character encoding used for file names inside the archive file. For a list of possible values, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html Defaults to the platform's default character encoding.

Examples

Extract the file `archive_001.zip` from directory `C:\archive\` into directory `C:\TEMP`.

```
OdiUnZip "-FILE=C:\archive\archive_001.zip" -TODIR=C:\TEMP\
```

OdiLockUnlockVCSRepository

Use this command to lock/unlock the remote VCS repository.

This command supports all the authentication types used in the VCS system.

Usage

```
OdiLockUnlockVCSRepository

[-VCS_LOCK_TYPE = <lock_type>]
[-VCS_TYPE=<vcs_type>]
[-VCS_AUTH_TYPE=<vcs_auth_type>]
[-VCS_URL=<vcs_url>]
[-VCS_USER=<vcs_user>]
[-VCS_PASS=<vcs_pass>]
[-VCS_PROXY_HOST=<vcs_proxy_host>]
[-VCS_PROXY_PORT=<vcs_proxy_port>]
[-VCS_PROXY_USER=<vcs_proxy_user>]
[-VCS_PROXY_PASS=<vcs_proxy_pass>]
[-VCS_SSH_PRIVATE_KEY_PATH=<vcs_ssh_private_key_path>]
[-VCS_SSH_PASS_PHRASE=<vcs_ssh_pass_phrase>]
[-VCS_SSH_PORT=<vcs_ssh_port>]
[-VCS_SSL_CERT_PATH=<vcs_ssl_ssl_cert_path>]
[-VCS_SSL_PASS_PHRASE=<vcs_ssl_pass_phrase>]
```

Parameters

Parameters	Mandatory	Description
VCS_LOCK_TYPE	Yes	Lock type for the VCS repository. The lock type can be: • LOCK to lock the VCS repository. • UNLOCK to unlock the VCS repository.
VCS_TYPE	Yes	Type of VCS. Can be SVN or Git.

Parameters	Mandatory	Description
VCS_AUTH_TYPE	Yes	Authentication type of the VCS used. The value can be: • BASIC for HTTP authentication. • PROXY for proxy authentication. • SSH for SSH authentication. • SSL for SSL authentication. • FILE for file authentication. • SVNBASIC for SVN Basic authentication. • GITBASIC for GIT Basic authentication.
VCS_URL	Yes	VCS Repository URL.
VCS_USER	No	VCS Username.
VCS_PASS	No	VCS Password.
VCS_PROXY_HOST	No	VCS Proxy Host. (Required if the VCS_AUTH_TYPE parameter is set to **PROXY**.)
VCS_PROXY_PORT	No	VCS Proxy Port. (Required if the VCS_AUTH_TYPE parameter is set to **PROXY**.)
VCS_PROXY_USER	No	VCS Proxy User. (Required if the VCS_AUTH_TYPE parameter is set to **PROXY**.)
VCS_PROXY_PASS	No	VCS Proxy Password. (Required if the VCS_AUTH_TYPE parameter is set to **PROXY**.)
VCS_SSH_PRIVATE_KEY_PATH	No	SSH private key file path if SSH Authentication is used.
VCS_SSH_PASS_PHRASE	No	SSH pass phrase for SSH authentication.
VCS_SSH_PORT	No	SSH port for SSH authentication.
VCS_SSL_CERT_PATH	No	SSL certificate path.
VCS_SSL_PASS_PHRASE	No	SSL pass phrase for SSL authentication.

Examples

```
./startcmd.sh -INSTANCE=OracleDIAgent1 OdiLockUnlockVCSRepository -
VCS_LOCK_TYPE=<LOCK|UNLOCK> -VCS_TYPE=<GIT|SVN> -VCS_AUTH_TYPE=<BASIC |
PROXY | SSH | SSL |FILE | SVNBASIC | GITBASIC> -VCS_URL=<vcs url> -
VCS_USER= -VCS_PASS=<Encoded password> -VCS_SSH_PORT=22
```

OdiUpdateAgentSchedule

Use this command to force an agent to recalculate its schedule of tasks.

Usage

```
OdiUpdateAgentSchedule -AGENT_NAME=<physical_agent_name>
```

Parameters

Parameters	Mandatory	Description
-AGENT_NAME=<physical_agent_name>	Yes	Name of the physical agent to update.

Examples

Cause the physical agent `agt_s1` to update its schedule.

```
OdiUpdateAgentSchedule -AGENT_NAME=agt_s1
```

OdiWaitForChildSession

Use this command to wait for the child session (started using the OdiStartScen tool) of the current session to complete.

This command checks every `<polling_interval>` to determine if the sessions launched from `<parent_sess_number>` are finished. If all child sessions (possibly filtered by their name and keywords) are finished (status of Done, Warning, or Error), this command terminates.

Usage

```
OdiWaitForChildSession [-PARENT_SESS_NO=<parent_sess_number>]
[-POLL_INT=<polling_interval>]
[-SESSION_NAME_FILTER=<session_name_filter>]
[-SESSION_KEYWORDS=<session_keywords>]
[-MAX_CHILD_ERROR=ALL|<error_number>]
```

Parameters

Parameters	Mandatory	Description
-PARENT_SESS_NO=<parent_sess_number>	No	ID of the parent session. If this parameter is not specified, the current session ID is used.
-POLL_INT=<polling_interval>	No	Interval in seconds between each sequence of termination tests for the child sessions. The default value is 1.
-SESSION_NAME_FILTER=<session_name_filter>	No	Only child sessions whose names match this filter are tested. This filter can be a SQL LIKE-formatted pattern.
-SESSION_KEYWORDS=<session_keywords>	No	Only child sessions for which ALL keywords have a match in this comma-separated list are tested. Each element of the list can be a SQL LIKE-formatted pattern.

Parameters	Mandatory	Description
-MAX_CHILD_ERROR= ALL|<error_number>	No	This parameter enables OdiWaitForChildSession to terminate in error if a number of child sessions have terminated in error: • ALL: Error if all child sessions terminate in error. • <error_number>: Error if <error_number> or more child sessions terminate in error. If this parameter is equal to 0, negative, or not specified, OdiWaitForChildSession never terminates in an error status, regardless of the number of failing child sessions.

Examples

Wait and poll every 5 seconds for all child sessions of the current session with a name filter of LOAD% and keywords MANDATORY and CRITICAL to finish.

```
OdiWaitForChildSession -PARENT_SESS_NO=<%=odiRef.getSession("SESS_NO")%>
-POLL_INT=5 -SESSION_NAME_FILTER=LOAD%
-SESSION_KEYWORDS=MANDATORY,CRITICAL
```

OdiWaitForData

Use this command to wait for a number of rows in a table or set of tables. This can also be applied to a number of objects containing data, such as views.

The OdiWaitForData command tests that a table, or a set of tables, has been populated with a number of records. This test is repeated at regular intervals (-POLLINT) until one of the following conditions is met: the desired number of rows for one of the tables has been detected (-UNIT_ROWCOUNT), the desired, cumulated number of rows for all of the tables has been detected (-GLOBAL_ROWCOUNT), or a timeout (-TIMEOUT) has been reached.

Filters may be applied to the set of counted rows. They are specified by an explicit SQL where clause (-SQLFILTER) and/or the -RESUME_KEY_xxx parameters to determine field-value-operator clause. These two methods are cumulative (AND).

The row count may be considered either in absolute terms (with respect to the total number of rows in the table) or in differential terms (the difference between a stored reference value and the current row count value).

When dealing with multiple tables:

- The -SQLFILTER and -RESUME_KEY_xxx parameters apply to **ALL** tables concerned.
- The -UNIT_ROWCOUNT parameter determines the row count to be expected for each table. The -GLOBAL_ROWCOUNT parameter determines the SUM of the row count number cumulated over the set of tables. When only one table is concerned, the -UNIT_ROWCOUNT and -GLOBAL_ROWCOUNT parameters are equivalent.

Usage

```
OdiWaitForData -LSCHEMA=<logical_schema> -TABLE_NAME=<table_name>
[-OBJECT_TYPE=<list of object types>] [-CONTEXT=<context>]
[-RESUME_KEY_VARIABLE=<resumeKeyVariable>
-RESUME_KEY_COL=<resumeKeyCol>
[-RESUME_KEY_OPERATOR=<resumeKeyOperator>]|-SQLFILTER=<SQLFilter>]
[-TIMEOUT=<timeout>] [-POLLINT=<pollInt>]
[-GLOBAL_ROWCOUNT=<globalRowCount>]
[-UNIT_ROWCOUNT=<unitRowCount>] [-TIMEOUT_WITH_ROWS_OK=<yes|no>]
[-INCREMENT_DETECTION=<no|yes> [-INCREMENT_MODE=<M|P|I>]
[-INCREMENT_SEQUENCE_NAME=<incrementSequenceName>]]
```

Parameters

Parameters	Mandatory	Description
`-LSCHEMA=<logical_schema>`	Yes	Logical schema containing the tables.
`-TABLE_NAME=<table_name>`	Yes	Table name, mask, or list of table names to check. This parameter accepts three formats: • Table Name • Table Name Mask: This mask selects the tables to poll. The mask is specified using the SQL LIKE syntax: the `%` symbol replaces an unspecified number of characters and the _ symbol is a single character wildcard. • Table Names List: Comma-separated list of table names. Masks as defined above are allowed.
`-OBJECT_TYPE=<list of object types>`	No	Type of objects to check. By default, only tables are checked. To take into account other objects, specify a comma-separated list of object types. Supported object types are: • `T`: Table • `V`: View
`-CONTEXT=<context>`	No	Context in which the logical schema will be resolved. If no context is specified, the execution context is used.
`-SQLFILTER=<SQLFilter>`	No	Explicit SQL filter to be applied to the table(s). This statement must be valid for the technology containing the checked tables. Note that this statement must not include the `WHERE` keyword.

Parameters	Mandatory	Description
-RESUME_KEY_VARIABLE=<resumeKeyVariable> -RESUME_KEY_COL=<resumeKeyCol> [-RESUME_KEY_OPERATOR=<resumeKeyOperator>]	No	The RESUME_KEY_xxx parameters enable filtering of the set of counted rows in the polled tables. • <key_column>: Name of a column in the checked table. • <operator>: Valid comparison operator for the technology containing the checked tables. If this parameter is omitted, the value > is used by default. • <variable_name>: Variable name whose value has been previously set. The variable name must be prefixed with : (bind) or # (substitution). The variable scope should be explicitly stated in the Oracle Data Integrator syntax; GLOBAL.<variable name> for global variables or <project code>.<variable name> for project variables.
-TIMEOUT=<timeout>	No	Maximum period of time in milliseconds over which data is polled. If this value is equal to 0, the timeout is infinite. The default value is 0.
-POLLINT=<pollInt>	No	The period of time in milliseconds to wait between data polls. The default value is 1000.
-UNIT_ROWCOUNT=<unitRowCount>	No	Number of rows expected in a polled table to terminate the command. The default value is 1.
-GLOBAL_ROWCOUNT=<globalRowCount>	No	Total number of rows expected cumulatively, over the set of tables, to terminate the command. If not specified, the default value 1 is used.

Parameters	Mandatory	Description
-INCREMENT_DETECTION=<no|yes>	No	Defines the mode in which the command considers row count: either in absolute terms (with respect to the total number of rows in the table) or in differential terms (the difference between a stored reference value and the current row count value). • If set to Yes, the row count is performed in differential mode. The number of additional rows in the table is compared to a stored **reference value**. The reference value depends on the -INCREMENT_MODE parameter. • If set to No, the count is performed in absolute row count mode. The default value is No.

Parameters	Mandatory	Description
-INCREMENT_MODE=<M|P|I>	No	This parameter specifies the persistence mode of the **reference value** between successive OdiWaitForData calls. Possible values are: • M: Memory. The **reference value** is nonpersistent. When OdiWaitForData is called, the **reference value** takes a value equal to the number of rows in the polled table. When OdiWaitForData ends, the value is lost. A following call in this mode sets a new reference value. • P: Persistent. The **reference value** is persistent. It is read from the **increment sequence** when OdiWaitForData starts and it is saved in the increment sequence when OdiWaitForData ends. If the **increment sequence** is not set (at initial call time), the current table row count is used. • I: Initial. The **reference value** is initialized and is persistent. When OdiWaitForData starts, the **reference value** takes a value equal to the number of rows in the polled table. When OdiWaitForData ends, it is saved in the increment sequence as in the persistent mode. The default value is M. Note that using the Persistent or Initial modes is not supported when a mask or list of tables is polled.
-INCREMENT_SEQUENCE_NAME=<incrementSequenceName>	No	This parameter specifies the name of an automatically allocated storage space used for reference value persistence. This increment sequence is stored in the Repository. If this name is not specified, it takes the name of the table. Note that this Increment Sequence is not an Oracle Data Integrator Sequence and cannot be used as such outside a call to OdiWaitForData.

Parameters	Mandatory	Description
-TIMEOUT_WITH_ROWS_OK=<yes|no>	No	If this value is set to Yes, at least one row was detected, and the timeout occurs before the expected number of rows has been inserted, the API exits with a return code of 0. Otherwise, it signals an error. The default value is Yes.

Examples

Wait for the DE1P1 table in the ORA_WAITFORDATA schema to contain 200 records matching the filter.

```
OdiWaitForData -LSCHEMA=ORA_WAITFORDATA -TABLE_NAME=DE1P1
-GLOBAL_ROWCOUNT=200 "-SQLFILTER=DATMAJ >
to_date('#MAX_DE1_DATMAJ_ORACLE_CHAR', 'DD/MM/YYYY HH24:MI:SS')"
```

Wait for a maximum of 4 hours for new data to appear in either the CITY_SRC or the CITY_TRG table in the logical schema SQLSRV_SALES.

```
OdiWaitForData -LSCHEMA=SQLSRV_SALES -TABLE_NAME=CITY%
-TIMEOUT=14400000 -INCREMENT_DETECTION=yes
```

OdiWaitForLoadPlans

Use this command to wait for load plan runs to complete.

Usage

```
OdiWaitForLoadPlans [-PARENT_SESS_NO=<parent_sess_guid>]
[-LP_NAME_FILTER=<load_plan_name_filter>] [-LP_KEYWORDS=<load_plan_keywords>]
[-MAX_LP_ERROR=ALL|<number_of_lp_errors>] [-POLLINT=<polling_interval_msec>]
```

Parameters

Parameters	Mandatory	Description
-PARENT_SESS_NO=<parent_sess_guid>	No	Global ID of the parent session that started the load plan. If this parameter is not specified, the global ID of the current session is used.
-LP_NAME_FILTER=<load_plan_name_filter>	No	Only load plan runs whose name matches this filter are tested for completion status. This filter can be a SQL LIKE-formatted pattern.
-LP_KEYWORDS=<load_plan_keywords>	No	Only load plan runs whose keywords contain all entries in this comma-separated list are tested for completion status. Each element in the list can be a SQL LIKE-formatted pattern.

Parameters	Mandatory	Description
-MAX_LP_ERROR=ALL|<number_of_lp_errors>	No	OdiWaitForLoadPlans terminates in error if a number of load plan runs are in Error status: • ALL: Error if all load plan runs complete in Error status. • <number_of_lp_errors>: Error if the number of load plan runs in Error status is at or above this value <number_of_lp_errors> when all load plan runs are complete. If this parameter is not specified or its value is less than 1, OdiWaitForLoadPlans never terminates in error, regardless of the number of load plan runs in Error status.
-POLLINT=<polling_interval_msec>	No	The time in milliseconds to wait between polling load plan runs status for completion state. The default value is 1000 (1 second). The value must be greater than 0.

Examples

Wait and poll every 5 seconds for all load plan runs started by the current session with a name filter of POPULATE% and keywords MANDATORY and CRITICAL to finish in a Done or Error status. If 2 or more load plan runs are in Error status when execution is complete for all selected load plan runs, OdiWaitForLoadPlans ends in error.

```
OdiWaitForLoadPlans -PARENT_SESS_NO=<%=odiRef.getSession("SESS_GUID")%>
-LP_NAME_FILTER=POPULATE% -LP_KEYWORDS=MANDATORY,CRITICAL
-POLLINT=5000 -MAX_LP_ERROR=2
```

OdiWaitForLogData

Use this command to wait for a number of modifications to occur on a journalized table or a list of journalized tables.

The OdiWaitForLogData command determines whether rows have been modified on a table or a group of tables. These changes are detected using the Oracle Data Integrator changed data capture (CDC) in simple mode (using the -TABLE_NAME parameter) or in consistent mode (using the -CDC_SET_NAME parameter). The test is repeated every -POLLINT milliseconds until one of the following conditions is met: the desired number of row modifications for one of the tables has been detected (-UNIT_ROWCOUNT), the desired cumulative number of row modifications for all of the tables has been detected (-GLOBAL_ROWCOUNT), or a timeout (-TIMEOUT) has been reached.

> **Note:**
>
> This command takes into account all journalized operations (inserts, updates, and deletes).
>
> The command is suitable for journalized tables only in simple or consistent mode.

Usage

```
OdiWaitForLogData -LSCHEMA=<logical_schema>  -SUBSCRIBER_NAME=<subscriber_name>
(-TABLE_NAME=<table_name> | -CDC_SET_NAME=<cdcSetName>)
[-CONTEXT=<context>] [-TIMEOUT=<timeout>] [-POLLINT=<pollInt>]
[-GLOBAL_ROWCOUNT=<globalRowCount>]
[-UNIT_ROWCOUNT=<unitRowCount> [-OPTIMIZED_WAIT=<yes|no|AUTO>]
[-TIMEOUT_WITH_ROWS_OK=<yes|no>]
```

Parameters

Parameters	Mandatory	Description
-CONTEXT=<context>	No	Context in which the logical schema will be resolved. If no context is specified, the execution context is used.
-GLOBAL_ROWCOUNT=<globalRowCount>	No	Total number of changes expected in the tables or the CDC set to end the command. The default value is 1.
-LSCHEMA=<logical_schema>	Yes	Logical schema containing the journalized tables.
-OPTIMIZED_WAIT=<yes|no|AUTO>	No	Method used to access the journals. • yes: Optimized method. This method works for later versions of journalizing. It runs faster than the nonoptimized mode. • no: Nonoptimized method. A count is performed on the journalizing table. This method is of lower performance but is compatible with earlier versions of the journalizing feature. • AUTO: If more than one table is checked, the optimized method is used. Otherwise, the nonoptimized method is used. The default value is AUTO.
-POLLINT=<pollInt>	No	The period of time in milliseconds to wait between polls. The default value is 2000.
-SUBSCRIBER_NAME=<subscriber_name>	Yes	Name of the subscriber used to get the journalizing information.

Parameters	Mandatory	Description
-TABLE_NAME=<table_name>	Yes	Journalized table name, mask, or list to check. This parameter accepts three formats: • Table Name • Table Name Mask: This mask selects the tables to poll. The mask is specified using the SQL LIKE syntax: the % symbol replaces an unspecified number of characters and the _ symbol acts as a wildcard. • Table Names List: List of table names separated by commas. Masks as defined above are **not** allowed. Note that this option works only for tables in a model journalized in simple mode. This parameter cannot be used with -CDC_SET_NAME. It is mandatory if -CDC_SET_NAME. is not set.
-CDC_SET_NAME=<cdcSetName>	Yes	Name of the CDC set to check. This CDC set name is the fully qualified model code, typically PHYSICAL_SCHEMA_NAME.MODEL_CODE. It can be obtained in the current context using a substitution method API call, as shown below: <%=odiRef.getObjectName("L", "model_code", "logical_schema", "D")%>. Note that this option works only for tables in a model journalized in consistent mode. This parameter cannot be used with -TABLE_NAME. It is mandatory if -TABLE_NAME is not set.
-TIMEOUT=<timeout>	No	Maximum period of time in milliseconds over which changes are polled. If this value is equal to 0, the timeout is infinite. The default value is 0.
-TIMEOUT_WITH_ROWS_OK=<yes|no>	No	If this parameter is set to Yes, at least one row was detected, and the timeout occurs before the predefined number of rows has been polled, the API exits with a return code of 0. Otherwise, it signals an error. The default value is Yes.

Parameters	Mandatory	Description
-UNIT_ROWCOUNT=<unitRowCount>	No	Number of changes expected in one of the polled tables to end the command. The default value is 1. Note that -UNIT_ROWCOUNT is not taken into account with -CDC_SET_NAME.

Examples

Wait for the CUSTOMERS table in the SALES_APPLICATION schema to have 200 row modifications recorded for the SALES_SYNC subscriber.

```
OdiWaitForLogData -LSCHEMA=SALES_APPLICATION
-TABLE_NAME=CUSTOMERS -GLOBAL_ROWCOUNT=200
-SUBSCRIBER_NAME=SALES_SYNC
```

OdiWaitForTable

Use this command to wait for a table to be created and populated with a predefined number of rows.

The OdiWaitForTable command regularly tests whether the specified table has been created and has been populated with a number of records. The test is repeated every -POLLINT milliseconds until the table exists and contains the desired number of rows (-GLOBAL_ROWCOUNT), or until a timeout (-TIMEOUT) is reached.

Usage

```
OdiWaitForTable -CONTEXT=<context> -LSCHEMA=<logical_schema>
-TABLE_NAME=<table_name> [-TIMEOUT=<timeout>] [-POLLINT=<pollInt>]
[-GLOBAL_ROWCOUNT=<globalRowCount>] [-TIMEOUT_WITH_ROWS_OK=<yes|no>]
```

Parameters

Parameters	Mandatory	Description
-CONTEXT=<context>	No	Context in which the logical schema will be resolved. If no context is specified, the execution context is used.
-GLOBAL_ROWCOUNT=<globalRowCount>	No	Total number of rows expected in the table to terminate the command. The default value is 1. If not specified, the command finishes when a new row is inserted into the table.
-LSCHEMA=<logical_schema>	Yes	Logical schema in which the table is searched for.
-POLLINT=<pollInt>	No	Period of time in milliseconds to wait between each test. The default value is 1000.
-TABLE_NAME=<table_name>	Yes	Name of table to search for.

Parameters	Mandatory	Description
-TIMEOUT=<timeout>	No	Maximum time in milliseconds the table is searched for. If this value is equal to 0, the timeout is infinite. The default value is 0.
-TIMEOUT_WITH_ROWS_OK=<yes|no>	No	If this parameter is set to Yes, at least one row is detected, and the timeout occurs before the expected number of records is detected, the API exits with a return code of 0. Otherwise, it signals an error. The default value is Yes.

Examples

Wait for the DE1P1 table in the ORA_WAITFORDATA schema to exist, and to contain at least 1 record.

```
OdiWaitForTable -LSCHEMA=ORA_WAITFORDATA -TABLE_NAME=DE1P1
-GLOBAL_ROWCOUNT=1
```

OdiXMLConcat

Use this command to concatenate elements from multiple XML files into a single file.

This tool extracts all instances of a given element from a set of source XML files and concatenates them into one target XML file. The tool parses and generates well formed XML. It does not modify or generate a DTD for the generated files. A reference to an existing DTD can be specified in the -HEADER parameter or preserved from the original files using -KEEP_XML_PROLOGUE.

> **Note:**
>
> XML namespaces are not supported by this tool. Provide the local part of the element name (without the namespace or prefix value) in the -ELEMENT_NAME parameter.

Usage

```
OdiXMLConcat -FILE=<file_filter> -TOFILE=<target_file>
-XML_ELEMENT=<element_name> [-CHARSET_ENCODING=<encoding>]
[-IF_FILE_EXISTS=<overwrite|skip|error>]
[-KEEP_XML_PROLOGUE=<all|xml|doctype|none>] [-HEADER=<header>]
[-FOOTER=<footer>]
```

Parameters

Parameters	Mandatory	Description
-FILE=<file_filter>	Yes	Filter for the source XML files. This filter uses standard file wildcards (?,*). It includes both file names and directory names. Source files can be taken from the same folder or from different folders. The following file filters are valid: • /tmp/files_*/customer.xml • /tmp/files_*/*.* • /tmp/files_??/customer.xml • /tmp/files/customer_*.xml • /tmp/files/customer_??.xml
-TOFILE=<target_file>	Yes	Target file into which the elements are concatenated.
-XML_ELEMENT=<element_name>	Yes	Local name of the XML element (without enclosing <> characters, prefix, or namespace information) to be extracted with its content and child elements from the source files. Note that this element detection is not recursive. If a given instance of <element_name> contains other instances of <element_name>, only the element of higher level is taken into account and child elements are only extracted as a part of the top element's content.
-CHARSET_ENCODING=<encoding>	No	Target files encoding. The default value is ISO-8859-1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html
-IF_FILE_EXISTS=<overwrite|skip|error>	No	Define behavior when the target file exists. • overwrite: Overwrite the target file if it exists. • skip: Do nothing for this file. • error: Raise an error.

Parameters	Mandatory	Description
-KEEP_XML_PROLOGUE=<all|xml|doctype|none>	No	Copies the source file XML prologue in the target file. Depending on this parameter's value, the following parts of the XML prologue are preserved: • all: Copies all of the prologue (XML and document type declaration). • xml: Copies only the XML declaration <?xml...?> and not the document type declaration. • doctype: Copies only the document type declaration and not the XML declaration. • none: Does not copy the prologue from the source file. **Note**: If all or part of the prologue is not preserved, it should be specified in the -HEADER parameter.
-HEADER=<header>	No	String that is appended after the prologue (if any) in each target file. You can use this parameter to create a customized XML prologue or root element.
-FOOTER=<footer>	No	String that is appended at the end of each target file. You can use this parameter to close a root element added in the header.

Examples

Concatenate the content of the IDOC elements in the files ord1.xml, ord2.xml, and so on in the ord_i subfolder into the file MDSLS.TXT.XML, with the root element <WMMBID02> added to the target.

```
OdiXMLConcat "-FILE=./ord_i/ord*.xml" "-TOFILE=./MDSLS.TXT.XML" -XML_ELEMENT=IDOC
"-CHARSET_ENCODING=UTF-8" -IF_FILE_EXISTS=overwrite -KEEP_XML_PROLOGUE=xml
"-HEADER=<WMMBID02>" "-FOOTER=</WMMBID02>"
OdiXMLConcat "-FILE=./o?d_*/ord*.xml" "-TOFILE=./MDSLS.TXT.XML" -XML_ELEMENT=IDOC
"-CHARSET_ENCODING=UTF-8" -IF_FILE_EXISTS=overwrite -KEEP_XML_PROLOGUE=none
"-HEADER=<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n<WMMBID02>"
"-FOOTER=</WMMBID02>"
```

Concatenate the EDI elements of the files ord1.xml, ord2.xml, and so on in the ord_i subfolder into the file MDSLS2.XML. This file will have the new root element EDI_BATCH above all <EDI> elements.

```
OdiXMLConcat "-FILE=./o?d_?/ord*.xml" "-TOFILE=./MDSLS2.XML" -XML_ELEMENT=EDI "-
CHARSET_ENCODING=UTF-8" -IF_FILE_EXISTS=overwrite -KEEP_XML_PROLOGUE=xml "-
HEADER= <EDI_BATCH>" "-FOOTER=</EDI_BATCH>"
```

OdiXMLSplit

This tool extracts all instances of a given element stored in a source XML file and splits it over several target XML files. This tool parses and generates well formed XML.

It does not modify or generate a DTD for the generated files. A reference to an existing DTD can be specified in the `-HEADER` parameter or preserved from the original files using `-KEEP_XML_PROLOGUE`.

> **Note:**
>
> XML namespaces are not supported by this tool. Provide the local part of the element name (without the namespace or prefix value) in the `-ELEMENT_NAME` parameter.

Usage

```
OdiXMLSplit -FILE=<file> -TOFILE=<file_pattern> -XML_ELEMENT=<element_name>
[-CHARSET_ENCODING=<encoding>] [-IF_FILE_EXISTS=<overwrite|skip|error>]
[-KEEP_XML_PROLOGUE=<all|xml|doctype|none>] [-HEADER=<header>]
[-FOOTER=<footer>]
```

Parameters

Parameters	Mandatory	Description
`-FILE=<file>`	Yes	Source XML file to split.

Parameters	Mandatory	Description
-TOFILE=<file_pattern>	Yes	File pattern for the target files. Each file is named after a pattern containing a mask representing a generated number sequence or the value of an attribute of the XML element used to perform the split: • **Number Sequence Mask**: Use the * (star) value to indicate the place of the file number value. For example, if the <file_pattern> is equal to target_*.xml, the files created are named target_1.xml, target_2.xml, and so on. • **Attribute Value Mask**: Between square brackets, specify the name of the attribute of <element_name> whose value should be pushed to create the file name. For example, customer_[CUSTID].xml creates files named customer_041.xml, customer_123.xml, and so on, depending on the value of the attribute CUSTID of the element used to split. Note that if a value repeats over several successive elements, target files may be overwritten according to the value of the -OVERWRITE parameter. Note that the pattern can be used for creating different files within a directory or files in different directories. The following patterns are valid: • /tmp/files_*/customer.xml • /tmp/files_[CUSTID]/customer.xml • /tmp/files/customer_*.xml • /tmp/files/customer_[CUSTID].xml

Parameters	Mandatory	Description
-XML_ELEMENT=<element_name>	Yes	Local name of the XML element (without enclosing <> characters, prefix, or namespace information) to be extracted with its content and child elements from the source files. Note that this element detection is not recursive. If a given instance of <element_name> contains other instances of <element_name>, only the element of higher level is taken into account and child elements are only extracted as a part of the top element's content.
-CHARSET_ENCODING=<encoding>	No	Target files encoding. The default value is ISO-8859-1. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html
-IF_FILE_EXISTS=<overwrite|skip|error>	No	Define behavior when the target file exists. • overwrite: Overwrite the target file if it exists. • skip: Do nothing for this file. • error: Raise an error.
-KEEP_XML_PROLOGUE=<all|xml|doctype|none>	No	Copies the source file XML prologue in the target file. Depending on this parameter's value, the following parts of the XML prologue are preserved: • all: Copies all of the prologue (XML and document type declaration). • xml: Copies only the XML declaration <?xml...?> and not the document type declaration. • doctype: Copies only the document type declaration and not the XML declaration. • none: Does not copy the prologue from the source file. **Note**: If all or part of the prologue is not preserved, it should be specified in the -HEADER parameter.
-HEADER=<header>	No	String that is appended after the prologue (if any) in each target file. You can use this parameter to create a customized XML prologue or root element.
-FOOTER=<footer>	No	String that is appended at the end of each target file. You can use this parameter to close a root element added in the header.

Examples

Split the file `MDSLS.TXT.XML` into several files. The files `ord1.xml`, `ord2.xml`, and so on are created and contain each instance of the IDOC element contained in the source file.

```
OdiXMLSplit "-FILE=./MDSLS.TXT.XML" "-TOFILE=./ord_i/ord*.xml" -XML_ELEMENT=IDOC
"-CHARSET_ENCODING=UTF-8" -IF_FILE_EXISTS=overwrite -KEEP_XML_PROLOGUE=xml
"-HEADER= <WMMBID02>" "-FOOTER= </WMMBID02>"
```

Split the file `MDSLS.TXT.XML` the same way as in the previous example except name the files using the value of the `BEGIN` attribute of the IDOC element that is being split. The XML prologue is not preserved in this example but entirely generated in the header.

```
OdiXMLSplit "-FILE= ./MDSLS.TXT.XML" "-TOFILE=./ord_i/ord[BEGIN].xml"
-XML_ELEMENT=IDOC "-CHARSET_ENCODING=UTF-8" -IF_FILE_EXISTS=overwrite -KEEP_XML
PROLOGUE=none "-HEADER= <?xml version=\"1.0\" encoding=\"UTF-8\"?>\n<WMMBID02>"
"-FOOTER=</WMMBID02>"
```

OdiZip

Use this command to create a ZIP file from a directory or several files.

Usage

```
OdiZip -DIR=<directory> -FILE=<file> -TOFILE=<target_file> [-OVERWRITE=<yes|no>]
[-RECURSE=<yes|no>] [-CASESENS=<yes|no>]
[-ENCODING=<file_name_encoding>]
```

Parameters

Parameters	Mandatory	Description
`-DIR=<directory>`	Yes if `-FILE` is omitted	Base directory (or folder) that will be the future root in the ZIP file to generate. If only `-DIR` and not `-FILE` is specified, all files under this directory are archived.
`-FILE=<file>`	Yes if `-DIR` is omitted	Path from the base directory of the file(s) to archive. If only `-FILE` and not `-DIR` is specified, the default directory is the current work directory if the `-FILE` path is relative. Use `*` to specify the generic characters. Examples: `/var/tmp/*.log` (all files with the `log` extension of the directory `/var/tmp`) `arch_*.lst` (all files starting with `arch_` and with the extension `lst`)
`-TOFILE=<target_file>`	Yes	Target ZIP file.

Parameters	Mandatory	Description
-OVERWRITE=<yes|no>	No	Indicates whether the target ZIP file must be overwritten (Yes) or simply updated if it already exists (No). By default, the ZIP file is updated if it already exists.
-RECURSE=<yes|no>	No	Indicates if the archiving is recursive in the case of a directory that contains other directories. The value No indicates that only the files contained in the directory to copy (without the subfolders) are archived.
-CASESENS=<yes|no>	No	Indicates if file search is case-sensitive. By default, Oracle Data Integrator searches files in uppercase (set to No).
-ENCODING=<file_name_encoding>	No	Character encoding to use for file names inside the archive file. For the list of supported encodings, see: https://docs.oracle.com/javase/8/docs/technotes/guides/intl/encoding.doc.html This defaults to the platform's default character encoding.

Examples

Create an archive of the directory `C:\Program files\odi`.

```
OdiZip "-DIR=C:\Program Files\odi" -FILE=*.* -TOFILE=C:\TEMP\odi_archive.zip
```

Create an archive of the directory `C:\Program files\odi` while preserving the `odi` directory in the archive.

```
OdiZip "-DIR=C:\Program Files" -FILE=odi\*.* -TOFILE=C:\TEMP\odi_archive.zip
```