Oracle Utilities
Cloud Service Foundation

Administrative User Guide

Release 20C

F35477-01

December 2020

ORACLE

Oracle Utilities Cloud Service Foundation Administrative User Guide

Release 20C

F35477-01

December 2020

Contents

Chapter 1

Oracle Utilities Cloud Service Foundation

This section documents the administrative functions for the Oracle Utilities Cloud Service Foundation.

Cloud Service Foundation

This chapter provides information on how to administer features associated with Oracle Utilities Cloud Service Foundation functionality.

> **NOTE:** This chapter does not apply to all implementations. Contact your system administrator to confirm if your implementation includes Oracle Utilities Cloud Service Foundation.

Process Automation Tool

The Process Automation Tool allows customers and implementers to orchestrate and automate a set of infrastructure related multi-step processes. The tool provides the ability to monitor the progress of such process and each of the steps. The tool also supports user actions at the process and step level.

The process automation tool is made out of the following system entities:

- Infrastructure Process - a Business Object of the Service Task MO.
- Infrastructure Process Step - a Business Object of the Service Task MO.
- Infrastructure Process Type - a Business Object of the Service Task Type MO.
- Infrastructure Process Step Type - a Business Object of the Service Task Type MO.

An Infrastructure Process references a type that includes the steps that are a part of that process. The steps are executed one by one according to their sequence as long as the previous step ended successfully. When a step in a process fails, the process will stop and wait for a user to take further actions.

An Infrastructure Process can span up to two product environments. This means that it can invoke and monitor steps in two systems. This, for example, allows it to orchestrate a configuration migration (using the Configuration Migration Assistant, described in your cloud service's *Administrative User Guide*) from one product environment to another. Such a migration

can include steps to create the export data on the **source** environment, move the export file to the **target** environment and execute all the import tasks on the target environment, all as a single Infrastructure Process. In this case the Infrastructure Process will be created on the **target** environment and will pull the configuration from the **source** environment.

NOTE:
This is done in order to ensure that only application users with sufficient security privileges in any system environment can create Infrastructure Processes that can update that environment.

The rule is that when an Infrastructure Process is created to run processes on two systems, the process will always be created on the **environment with the higher security requirements**. For example, if you need to run processes on the Development and Test environments, the infrastructure process will be created and executed from the Test environment (assuming that the Test environment has higher security requirements than the Development environment).

The steps that make an Infrastructure Process reference a step type that governs their behavior. A comprehensive set of process and step types is provided with the product. This set can be extended by implementers as needed.

Creating an Infrastructure Process

To create a new process, navigate using **Admin > Implementation Tools > Infrastructure Process**. Use the Infrastructure Process Query to search for an existing process. One an Infrastructure Process is selected, you are brought to the maintenance portal to view and maintain the selected record.

The following points provide information about creating a new Infrastructure Process.

Infrastructure Process references an **Infrastructure Process Type**. When creating a new process you need to select the type that the process will be based on.

The **Name** of the process is used for tracking purposes. The name doesn’t have to be unique in the system.

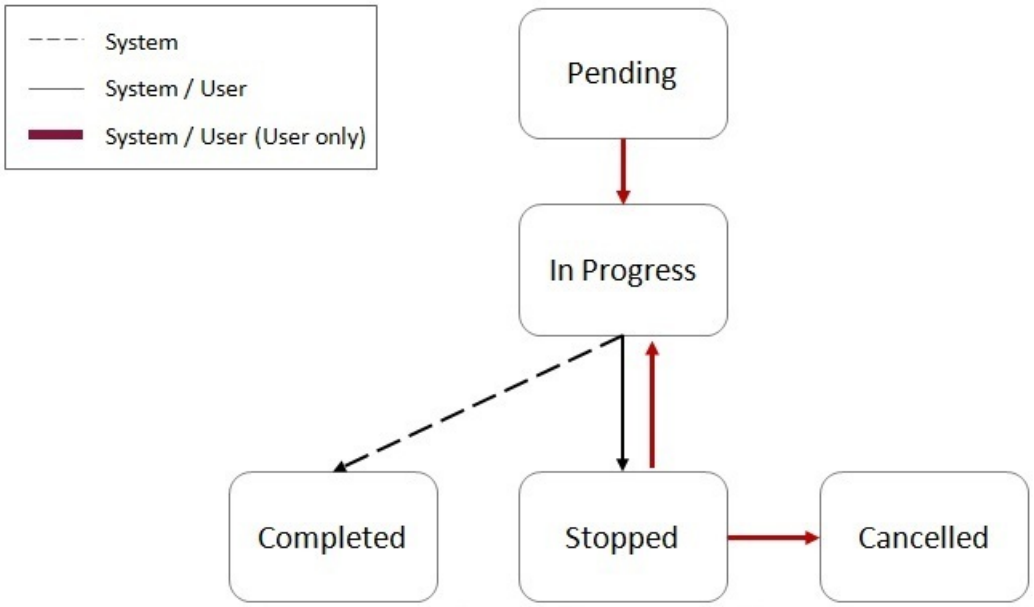
The **Process Steps** list is read-only and represents the steps that were configured on the Infrastructure Process Type.

NOTE:
Changing information on the Infrastructure Process Type, including the steps, will not affect existing Infrastructure Process records.

The **Process Details** section represents additional information that is required according to the corresponding process type definition.

Executing an Infrastructure Process

Infrastructure Process Lifecycle



New processes are created in a Pending status. While in this status the Infrastructure Process record can be modified, deleted or duplicated.

> **NOTE:** Once an Infrastructure Process record is created, its type cannot be changed. To change the type, you can delete the process that was created with the wrong type (if it is still in Pending status) and create a new process with the correct type.

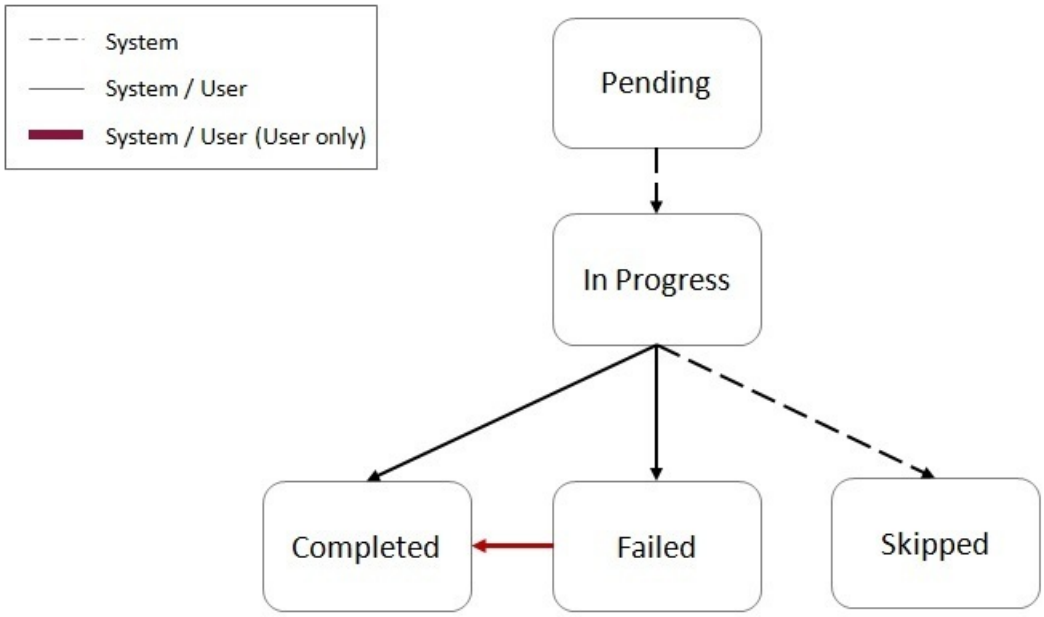
Use the **Start** action, on the maintenance portal to begin the process execution. The process status will change to **In Progress**.

> **NOTE:** While the Infrastructure Process is in **Pending** status, the steps will not have a status or any available actions. When the process transitions to **In Progress** status the process step records are created. From that point on they have a status and actions according to the step that and status.

While the process is **In Progress** the steps of the process will execute one by one according to their sequence. The process can be stopped using the **Stop** action. In this case the current step that is in progress will continue but the next step will not be executed. If any of the steps fail, the process will also transition into **Stopped** status and will not execute the next step.

When the process is in **Stopped** status it can be resumed by selecting the **Resume** action, or it can be cancelled by selecting the **Cancel** action.

Infrastructure Process Step Lifecycle



A process steps is created in **Pending** status and is transitioned to **In Progress** by the process logic when its turn to execute arrives.

When a step is **In Progress**, the following transitions are possible:

- Transition the step into Completed status upon successful completion of the step's work or by selecting the **Complete** action.
- Transition the step into **Failed** status when an error was detected during the step's work or by selecting the **Fail** action.
- Steps may include logic that will transition the step into **Skipped** status if that logic determines that the step should not be executed.

When the step is In Progress, selecting the **Job Status** action will display information about the job running status (if it is a batch job type of step).

> **NOTE:** Manually changing the step status to **Completed** or **Failed** while the step is In Progress should be carefully considered. This should only be done when step stays **In Progress** status for a long time and upon investigation it is clear that some manual intervention is required.

When a step is in **Failed** status it is still possible to manually transition the step into Completed status.

> **NOTE:** Transitioning the step from **Failed** to **Completed** status should only be done if it is clear that the step was able to successfully accomplish its task. This will likely be established after some investigation.

Monitoring Infrastructure Process Progress

Infrastructure Process Control

The Infrastructure Process Control portal provides an overview of Infrastructure Processes that are active (i.e. in **Pending** or **In Progress** status) or processes that did not complete (i.e. in **Stopped** status) within the last X days.

Navigate to Infrastructure Process Control portal, using **Admin > Implementation Tools > Infrastructure Process Control**.

Available actions on this portal are:

- **New Process** - creates a new Infrastructure Process
- **Environment Setup** - directs the user to the master configuration related to the process automation tool

Select any of the processes from the portal lists for more information.

Monitoring a Running Infrastructure Process

When a process is running (in **In Progress** status), you can monitor its progress by refreshing the page. To configure automatic page refresh, refer to the User Portal Preferences for this portal.

The Infrastructure Process maintenance portal provides information about the status of the overall process as well as each of the steps.

You can also view the details of each individual step including their log for more information or to investigate exceptions.

Dealing with Exceptions

Infrastructure Process steps can be divided into two main categories:

- Steps that start a batch process (job)
- Steps that are not batch process related

In addition, steps can be local or remote. Local steps execute on the same product environment that the process is executing while remote steps are executed on a different product environment.

Batch Job Step Exceptions

Batch job steps start a batch job either locally or on a different product environment.

Batch job step in Failed status as a result of a batch job error

When a batch process (local or remote) ends with an error, the process step is automatically transitioned to **Failed** status.

In this case, you should:

1. Ascertain the cause of the job failure (in the local or remote environment).
2. If the issue is temporary or can be fixed, fix it and then rerun the batch job manually using the Batch Job Submission page (in the local or remote environment). If the resubmitted job ended successfully, you can transition the process step to the **Completed** status and use the **Resume** action at the process level to continue the process execution.

If the issue cannot be resolved, further investigation is required. In this case the process should remain in **Stopped** status.

Batch job step in In Progress status while batch job ended with an error

When an Infrastructure Process Step that is associated with a batch job stays in **In Progress** status while the batch job ended with an error, further investigation is required. One possible cause for that situation could be an error that causes the job to fail and at the same time also prevents the job from communicating its status back to the originating process step.

In this case you should manually transition the step into **Failed** status and continue your investigation.

NOTE: If you decide to stop the Infrastructure Process altogether (as a result of the exception), it is a good idea to mark the batch job that failed as “Do Not Attempt Restart” in the Batch Run Control for that batch run. This will ensure that

the next time the Infrastructure Processes runs, when it gets to that batch process step it will not attempt to resume the execution that ended with an exception, possibly causing a repeat of the exception.

Batch job step in In Progress status while batch job ended successfully

When an Infrastructure Process Step that is associated with a batch job stays in **In Progress** status while the batch job ended successfully, further investigation is required. In this case, once you ascertained that the job did actually end successfully, you can manually transition the process step into **Completed** status and use the **Resume** action at the process level to continue its execution.

Non-Batch Step Exceptions

Infrastructure Process Steps that are not associated with batch jobs can experience errors as well and therefore can fail. As a reminder, the logic for these steps can experience local or remote errors. When such a step fails, information about the failure can be found in the process step log.

Process Automation Tool Setup

The process automation tool requires the setup of the following components:

- Master Configuration
- External Systems
- Message Senders

In addition, a security setup is required.

The setup itself is done automatically as part of the environment provisioning process. The information in this section is provided for reference and in case changes in existing setup are required.

Security Setup

Before the process automation tool can be used, the security credentials of the Message Senders, used by the process automation tool, must be updated.

In order to execute the security setup, navigate using **Admin > Implementation Tools > Process Automation Security Setup**.

The setup script will ask you to provide a User ID, a Password and to indicate whether or not to override the existing security setup.

Please contact your system or security administrator for more information about what User ID should be used for this setup.

You should select to override existing security setup when you need to update the User ID or the Password on the existing Messages Senders, used by the process automation tool. A common use case will be mandatory periodic password rotation for the user account that was selected for the setup.

NOTE: At the end of the process, you will be asked to flush all the system cache. This is required in order for the information update to take effect.

Master Configuration Setup

The process automation tool relies on master configuration settings in order to identify all the product environments that are available to it. The Master Configuration is automatically setup as part of the system provisioning process.

This section describes the setup options in case manual changes are needed.

To access the configuration for the process automation tool, navigate using **Admin** > **General** > **Master Configuration** and look for Process Automation Configuration.

The Main section includes the definition of the **Current Environment** code and the **default file extension** for CMA export files.

The Current Environment code is a **unique** code that is given to each product environment.

The Environment List section describes all the existing product environments that the process automation tool need to interact with, for example, in order to run a process that moves integration configuration from one environment to another.

The environment in the list includes the following details:

- Environment: the environment code.
- **Product**: defined in the Process Automation Product extendable lookup.
- **Domain**: defined in the Process Automation Domain extendable lookup.
- **Migration Source**: select “Yes” if you can migrate configuration data from this environment.

  NOTE: You should not allow automated migration of configuration from your development environment directly to your production environment but you would allow migrating from the development to test environments.

- **External System**: reference to the External System definition that will support communication with the row’s environment.

External Systems and Message Senders

The process automation tool communicates with other product environments through Outbound Messages.

The automated setup for the process automation tool creates a set of Message Senders and External Systems to support communication with other product environments.

Each product environment which is defined on the Process Automation Configuration has a corresponding unique External System. Each External System contains the details of all the message types that it supports, the message sender for each message type and additional data if required.

Configuring New Infrastructure Processes

The process automation tool configuration is essentially a collection of process types and step types that support the automation of various multi-step processes. A set of Infrastructure Process Types and Infrastructure Process Step Types is provided with the base product. Customers and implementers can create new process and step types.

This section describes the actions required to configure new Infrastructure Process Type and Infrastructure Process Step Type.

Configuring a New Infrastructure Process Step Type

In order to configure a new step type, navigate using **Admin > Implementation Tools > Infrastructure Process Step Type**. Use the Infrastructure Process Step Type Query to view existing step types.

CAUTION: Base product step type names start with “K1” and should not be change. Changing these step types can break the process and step types that are provided with the product.

The following points provide information about creating a new Infrastructure Process Step Type.

When creating a new step type, a business object has to be selected first. The Infrastructure Process Step Type BO contains the information necessary for the Infrastructure Process Step that references that step type. For example, a batch process step type BO contains information related to the batch code that should be submitted and the parameters that will be passed to it. You can use existing Infrastructure Process Step Type BOs or create new BOs according to your business requirements.

The **Related Transaction BO** references the Infrastructure Process Step BO. The step BO contains the business logic associated with the step. For example, a batch process step BO is responsible to submit a batch job with to the information from the step type. You can use existing Infrastructure Process Step BOs or create new BOs according to your business requirements.

The **Process Step Type Details** section represents additional information that is required for this specific step type BO.

Configuring a New Infrastructure Process Type

In order to configure a new process type, navigate using **Admin > Implementation Tools > Infrastructure Process Type**. Use the Infrastructure Process Step all-in-one portal to view existing process types.

> **CAUTION:** Base product process type names start with “K1” and should not be change. Changing these process types can break the base product process types that are provided with the product.

The following points provide information about creating a new Infrastructure Process Type.

When creating a new process type, a business object has to be selected first. The Infrastructure Process Type BO contains information necessary for the Infrastructure Process that references that process type. You can use existing Infrastructure Process Type BOs or create new BOs according to your business requirements.

The **Related Transaction BO** references the Infrastructure Process BO. The process BO contains the business logic associated with the process. This includes the orchestration of the process steps. The process BO also contains information that is shared by step business logic during the process. For example, a process BO that handles migration of data between product environments can have a buffer that will hold the information pulled from the source environment using one step so that another step can use it to push that to the target environment. You can use existing Infrastructure Process BOs or create new BOs according to your business requirements.

The **Process Type Steps** section includes a collection of step types. The sequence of steps has to be unique and this will be sequence of execution.

> **CAUTION:** When selecting step types for a process type you need to consider the fact that not all step types can be used on all process types. For example, some steps need to get or save information in the process itself so it can be used by other steps of that process. These types of steps can only work with a certain type of process. Please refer to the detailed description in each of the base product provided Infrastructure Process Step Types to see what process types they can be referenced on.

The **Infrastructure Process Type Details** section represents additional information that is required for the specific process type BO.

CMA Migration Infrastructure Process Types

The following Infrastructure Process Types are provided to support the following configuration migration use cases:

- CMA Accelerator Load - import an exported file created by the Configuration Migration Assistant export process.
- CMA Migration - a wholesale or piecemeal configuration migration (using the Configuration Migration Assistant), exporting configuration from a source environment and importing it to a target environment.

CMA Accelerator Load (K1-CMA-LOAD)

The K1-CMA-LOAD Infrastructure Process Type supports importing of an exported CMA file.

K1-CMA-LOAD is based on K1-CMAMigrationProcessType Infrastructure Process Type BO and uses the K1-CMAMigration as the Infrastructure Process BO.

The CMA Accelerator Load process type includes load an exported CMA and to run all the batch processes that are a part of the CMA import.

NOTE:
This process type does not include a step for a manual user review. It is assumed that when loading an accelerator, there shouldn't be anything to review and the new configuration content is expected to be new for the most part or OK to override existing configuration.

Using the CMA Accelerator Load Infrastructure Process Type

In order to import a CMA export file (e.g. an accelerator), you need to:

- Upload the file to the system designated storage location for CMA uploads. The location is configured on the CMA Master Configuration entry and is automatically defaulted to an Oracle Cloud Object Storage location during the system installation process.
- Create a new infrastructure process with the K1-CMA-LOAD Infrastructure Process Type and provide the following additional details:
  - **Name**.
  - The **Process Steps** list is for reference only.
  - **Export filename**: specify the CMA export file name that was uploaded to the system. You should specify the full name including the file extension name.
  - **User Review Needed**: select "Yes" only if you are expecting the CMA file to include data that already exists in the system and need to be reviewed before changes are applied.
  - **Default Status For Add**, **Default Status For Update** are relevant in case you selected to have a user review. In this case define the default status of migration data items when imported into the target environment.
- Once the process is saved, you can start it by selecting the Start action.

CMA Migration (K1-CMA-MIGRATE)

The K1-CMA-MIGRATE Infrastructure Process Type supports a full or partial migration of configuration data, from a remote environment (the source) to the environment that the process is running on (the target).

NOTE: The configuration migration process is done as a PULL (running on the target environment and pulling data from the source environment) to make sure that users without security privileges to make changes in a product environment "A" won't be able to run a process from a remote environment "B" that will force changes into "A".

K1-CMA-MIGRATE is based on K1-CMAMigrationProcessType Infrastructure Process Type BO and uses the K1-CMAMigration as the Infrastructure Process BO.

This configuration migration process type includes all the necessary steps to create an export CMA file from a migration request on the source environment, store the file in a shared file location (accessible by the source and target environment processes) and to run all the import batch processes on the target environment to process the file. This process type also includes a step for a manual user review of the imported data if needed.

NOTE: The shared file location is configured on the Migration Assistant Configuration master configuration entity and is defaulted to an Oracle Cloud Object Storage location during the system installation process.

Using the CMA Migration Infrastructure Process Type

In order to migrate configuration data from a remote environment (the source environment) to the current environment (the target environment) you need to:

- Create a new infrastructure process with the K1-CMA-MIGRATE Infrastructure Process Type and provide the following additional details:
  - **Name**.
  - **From Environment**: select the source environment from which the data will be migrated to the target environment (the current environment). The list of valid environments to choose from is taken from the Process Automation Master Configuration. The target environment is always the current environment - the one that this process is created on.
  - The **Process Steps** list is for reference only.
  - **Migration Request**: specify the CMA Migration Request ID that will be used for the migration process. The migration request should exists on the source environment and will be validated against that environment.
  - **Export filename**: specify the CMA export file name that will be used for the export and import process. You should specify the full name including the file extension name. If you leave this field empty, the system will default the file name according to the migration request ID and the default file extension name.
  - **User Review Needed**: select “Yes” only if you are expecting the CMA file to include data that already exists in the system and need to be reviewed before changes are applied.
  - **Default Status For Add**, **Default Status For Update** are relevant in case you selected to have a user review. In this case define the default status of migration data items when imported into the target environment.
- Once the process is saved, you can start it by selecting the Start action.

NOTE: Before using this Infrastructure Process Type, make sure that the import and export directories for the Migration Assistant Configuration master configuration entry are pointing to the SAME shared location. This is required for a successful completion of this process type.

Data Conversion Support for Cloud Implementations

This chapter provides information and requirements on data conversion support for Oracle Utilities Cloud implementations.

Data Conversion Overview

The ability to convert customer data from legacy systems to OUAF based products is critical for every implementation. The following chapter describes the overall process, tools and considerations that are specific to OUAF based product implementations on Oracle Cloud.

Conversion Steps

In general data conversion effort in the cloud is divided into the following steps:

- Prepare for conversion
- Create conversion data files
- Upload conversion data files to the cloud

- Process conversion data files
- Process converted data in cloud application
- Approve converted data for production mode

NOTE:
It is important to note that the conversion process is a highly iterative process. All the steps above are likely to be done (completely or partially) multiple times during the conversion process.

For example, customer typically start with a small set of data to convert, they fix issues discovered with that data and then re-create new data files and proceed to convert them. This can happen many times, each time with either a different data sample or bigger and bigger data conversion sets until all the required data has gone through the conversion process.

Prepare for Conversion

Preparing for conversion will typically include the following actions:

- Identify the OUAF product entities (Maintenance Objects) for which data will be converted. Refer the specific OUAF product documentation for information about converted entities for that product. Typically, each of the product converted entities will have a set of conversion staging tables that will receive the data from your legacy system before they get processed into the system.

  NOTE: Conversion staging schema design may vary between products, please refer to specific product's documentation for more details.

- Evaluate your project's requirements for
  the Information Lifecycle Management (ILM) and Data Archiving. If your implementation uses ILM, the ILM Date must be provided as part of the legacy data extract for each eligible table.

  NOTE: ILM support may vary between products, please refer to specific product's documentation for more details.

- Identify tables and maintenance objects whose data requires special treatment. That may include high volume tables, extremely large CLOB data, tables with multiple CLOB fields or any other special considerations related to the legacy data being converted.

  Address the special circumstances by customizing the default configurations:

  - Enable conversion for tables that are not marked for conversion in the metadata use the Master Configuration
  - If your table(s) contain multiple CLOBs, but you don't plan to supply the legacy data for all of them, create Conversion Task Type for this table(s) and populate **Exclude CLOB Field** list and then add a corresponding entry to Master Configuration's Special Instructions
  - If you dealing with high volumes of data, you may want to adjust the load parameters and options for the SQL Loader. In this case, create a **custom Control File Header** (Managed Content), configure a dedicated Conversion Task Type referencing this custom Control File Header and then add a corresponding entry to Master Configuration's Special Instructions. Note that creating the custom Control File Header will result in a control file composed from the custom fragment and the generated fields list.
  - If the generated field definitions list in the control file doesn't satisfy your requirements, you may completely override the control file. In this case, create a **custom Control File** (Managed Content, same as above), configure a dedicated Conversion Task Type referencing this custom Override Control File and then add a corresponding entry to Master Configuration's Special Instructions. Make sure your custom Control File contains all the run-time substitution variables that appear in the base Control File Header.

  See Conversion Task Types and Conversion Master Configuration chapters for more details.

- Generate the conversion artifacts for the product entities you plan to convert data for. This will include, for example, a set of data specification files that describe the format of data required for each file.
- Explore conversion tasks generated for tables and/or maintenance objects, examine the input data specifications and file lists and determine the expected data formats and input file names.
- Identify your various sources of data and extraction methods.

Create Conversion Data Files

In this step you create the set of data conversion files that were identified for the conversion process. You can choose to generate data files for each staging table or create a smaller set for each converted Maintenance Object.

The data conversion files typically do not include all the data to begin with but start small and grow with every iteration of the conversion process until they reach their maximum size at the end.

NOTE: When performing data conversion in the cloud, customers don't have access to the database staging tables. Data scrubbing and transformation activities has to take place before the data files are generated (in each conversion iteration) so the result of these activities is reflected on the data files that are moved to the cloud to be loaded into the staging tables.

Upload Data Conversion Files to the Cloud

Since data conversion files are created locally they have to be uploaded to the cloud environment in order to be processed and loaded into the conversion staging tables.

In this step you should setup a dedicated input location on the object storage and upload the input data files there.

For more details refer to File Locations .

Process Conversion Data Files

After uploading the files to the cloud, you should run a set of batch processes that will process the data files using Oracle SQL Loader and insert the data to the product conversion staging tables.

The staging table load process also produces output files that provide feedback on the load process.

You can download these output files for review.

NOTE: The process that imports the data in the files to the staging tables, using Oracle SQL Loader, is using the artifacts that were generated in the "Prepare for Conversion" step.

For more details refer to Understanding Load Data Process .

Process Converted Data in Cloud Application

When the data intended for conversion (for a conversion iteration) is loaded into the appropriate conversion staging tables, the OUAF application in the cloud can process that data according to the product's data model, technical and business rules.

Refer to the OUAF product specific documentation for further guidelines and information on the specific processes and guidelines related to data conversion.

NOTE:
This is one step of the overall iterative process but it is also includes many internal steps, for example:

1. Validation of the data in the conversion staging tables.
2. Generation of the system assigned keys for the converted entities.

3. Migration of the converted data from the staging tables into the actual system tables.

When the processing of the converted data in this iteration is completed, information from the system is used to identity changes necessary to source data, the extraction process or other activities involved in producing the data to be converted.

The process can then start over by loading a new set of data into data conversion files, load them to the staging tables in the cloud and process them once more to get more feedback.

Approve Converted Data for Production Mode

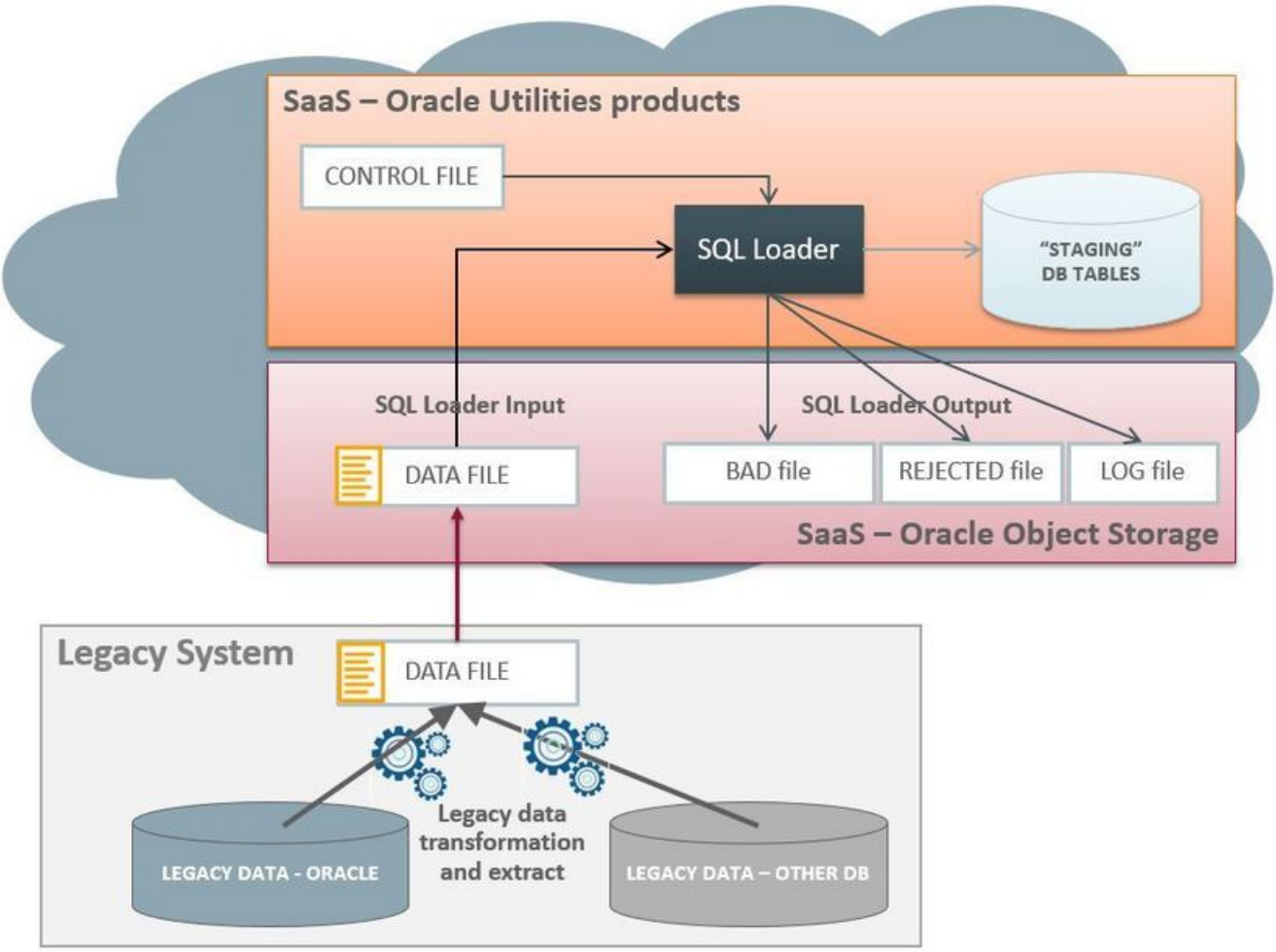
When the quality and scope of data converted has reached its target, the process can stop. Once the overall data has been approved for production mode the conversion process comes to an end. The process of approving the data includes many activities, for example:

- Running various reports to compare converted data with legacy data to verify the quantity and quality of the data.
- Running reports and tests in the new application to verify the data and the system's behavior using that data.
- Manual review of random and key data in the system.

SQL Loader

Loading the data in the data conversion files into the product conversion stating tables is done using Oracle SQL Loader.

Oracle SQL Loader is an Oracle Database Tool that enables mass upload of data from various sources into database tables in a fast and safe way.



SQL Loader processing is determined by Control Files. The control files for the data conversion process are provided for each of the product conversion staging tables as part of the artifact generation which is a part of the preparations activities before starting the conversion process.

The Control Files typically define:

- The location of input files to load.
- The character set of the input data.
- The target table(s) for the load and how to insert the data (e.g. append to or override).
- How to map the data in the input file to tables and fields.
- The location of the output files (e.g. log, bad or rejected records).
- Options for parallel processing and load balancing.

The control files that are generated for the product include all the parameters necessary to load the product conversion staging tables. There are some customization options for these files that are discussed later.

After loading the control files, SQL Loader reads the records in the input data file(s), converts the input into database tables and columns that are inserted into the database.

SQL Loader has 2 modes of operation:

- **Conventional Path Load** – data is inserted to the database by using database SQL INSERT statements.

- **Direct Path Load** – data is inserted into the database using data blocks that are written directly to the database. This method is much faster and therefore is the method selected for the data conversion process.

SQL Loader has the ability to deal with multiple data files mapped to multiple tables as well as loading data into CLOB fields.

Using SQL Loader When Processing Conversion Data Files

Processing of uploaded conversion data files is done via a Batch Process (for each staging table or MO) that does the following:

- Prepares parameters for the SQL Loader process and retrieves Control File for an input table or MO
- Invokes SQL Loader and processes the legacy data extract files that were uploaded to the dedicated File location on cloud
- Archive the input file(s) and upload the SQL Loader output files to the File location so they can be subsequently downloaded by the customer for review.

Conversion Artifacts

The base product includes a set of artifacts that support the conversion process. These artifacts are provided on demand (generated as a result of a user request in the system).

The available artifacts are:

- Input Data Specifications
- SQL Loader Control Files
- Files List

Conversion Input Data Specifications

The system will generate a text file with data specification instructions for each of the conversion tables that are supported by the base product.

The specification files are for reference ONLY and are not used by the system at run time. These files describe the data format for each table so that the data specified can be loaded correctly with the Control File that is also generated by the system.

For more details refer to Conversion Artifact Generator .

SQL Loader Control Files

The control files are for internal use and are generated for each of the conversion tables that are supported by the product. These files are used by SQL Loader to interpret and load the data that is provided in the conversion data files.

The control file contains:

- General load parameters and options
- Data field specifications and table-specific load instructions

For more details refer to Conversion Artifact Generator .

Configuring Conversion

The following sections describe the settings and configurations used by various conversion processes.

Conversion Master Configuration

The **Conversion Data Upload Master Configuration** controls generic aspects of legacy data load process.

Main section captures the definitions globally applicable for all converted objects (tables and/or maintenance objects):

- **Default Conversion Instructions for Artifact Generator** reference conversion task types that define load parameters for individual tables or maintenance objects.
  The Master Configuration captures conversion task types with default configurations for:
  - Typical Table
  - Typical maintenance Object
  - Key Table.

Special Instructions section captures the definitions that support various idiosyncratic data upload scenarios

- **Override Conversion Eligibility — Table** list allows you to specify tables that are not marked for conversion in the metadata, but your implementation wishes to include in the conversion scope.

  NOTE: If the table you wish to convert has a related key table, this key table has to be explicitly added to this list.

- **Override Instructions** for **Tables** and **Maintenance Objects** define Conversion Instructions (conversion task types) for objects that require special processing or data formatting. For example, the default conversion instruction defines that CLOB data is supplied as a secondary file. The special instruction for a specific table may define that CLOB data is supplied in the main data file.

Conversion Task Types

To configure a new conversion task type, navigate using Admin –> Conversion Support –> Conversion Task Type.

Conversion task type controls the data load parameters for the target table or maintenance object. Conversion task type also defines the transactional business object for Conversion Task. The algorithms linked to the transactional business object are responsible for the generation of the Conversion Artifacts: control files, input data specifications and file lists.

Conversion Task Types defines:

General Input Data Instructions (all Conversion Task Types Business Objects)

- **Control File Header** - a fragment of the control file that contains general load parameters. This fragment is included in the generated control file. This is a Foreign Key reference to the **Managed Content** entry.
- **Override Control File** - a full text of the control file that contains both load parameters and field definitions. The entire text is copied into generated control file ‘as is’. This is a Foreign Key reference to the **Managed Content** entry.
- **Date and DateTime fields’ formats**. These attributes are referencing the Extendable Lookup values for *K1–ConvLoadDateFormatLookup* and *K1–ConvLoadDateTimeFmtLookup business objects*
- Strings to use as **data delimiters** and **data enclosing characters**. These attributes are referencing the Extendable Lookup values for *K1–ConvFileDelimiterLookup* and *K1–ConvFileEnclosingCharLookup* business objects
- **CLOB data extract rule** indicates whether the CLOB data is supplied as part of the input data file or as a separate, secondary file
- **CLOB data line separator** defines how the records are separates in the secondary (CLOB) data file. This value should be set according to the operating system where the legacy data extract was created. This attribute is referencing the Extendable Lookup values for *K1-ConvCLOBLineSeparatorLookup* business object.

  For Linux CLOB Line Separator set the value to ***LF***, for Windows, set the value for ***CR+LF***

CLOB Field Instructions for Tables/MO-s that contain multiple CLOB fields (Business Objects *K1-ConvArtMultClobMOTaskType* and *K1-ConvArtMultClobTblTaskType*)

- Target Object — T**able** or **Maintenance Object**
- **Exclude CLOB Fields** list — the list of CLOB fields that are not expected to be provided with the legacy data extract and therefore should be excluded from the generated control file.

The following Conversion Task Types are pre-configured in the product:

- **K1-CNV-TABLE** - meant to be used for a single table load. It is referencing a Transactional Conversion Task Business Object whose algorithms generate Table Conversion Artifacts
- **K1-CNV-MO** - meant to be used for a single maintenance object load. It is referencing a Transactional Conversion Task Business Object whose algorithms generate Maintenance Object Conversion Artifacts
- **K1-CNV-KEY-TABLE** – meant to be used for the Key Tables. Key Tables are index-organized tables (IOT) and as such cannot be loaded in multiple concurrent sessions of SQL loader. This Task Type is referencing a Control File Header (Managed Content) where ***Parallel Load*** parameter is set to ***false***

Special configurations for tables that include more than one CLOB field

- If no dedicated Conversion Task Type is configured and defined on the Master Configuration as Override Instruction for the Table or MO, the system uses the default Conversion Instruction defined on the Master Configuration. In this case, the generated Control File and Input Data File Specifications will reference a **single CLOB field** among Table's multiple CLOB fields (the one with the lowest table field sequence).
- If a dedicated Conversion Task Type is configured for the Table or Maintenance Object, using business objects *K1-ConvArtMultClobMOTaskType* or *K1-ConvArtMultClobTblTaskType*), but no fields were added to **Exclude CLOB Fields** list, the generated Control File and Input Data File Specifications will reference **all CLOB fields**.
- If a dedicated Conversion Task Type is configured for the Table or Maintenance Object, using business objects Business Objects *K1-ConvArtMultClobMOTaskType* or *K1-ConvArtMultClobTblTaskType*), and some fields were added to **Exclude CLOB Fields** list, the generated Control File and Input Data File Specifications will **omit excluded CLOB fields.**

Conversion Artifact Generator

Conversion Artifacts is a set of supporting files used for data upload with SQL Loader. The artifacts are generated based on the table and maintenance object metadata and the definitions from Conversion Master Configuration and Configuration Task Types. Conversion artifacts include SQL Loader control files, input data file specifications and other supplemental files.

- **Input data file specifications** provide detailed field-by-field description of the expected data type and format, field order and other instructions. Your implementation may use these specifications to create the legacy data extract.
- **Control file** defines the rules and options of the data upload
- **File List** enumerates input data files expected to be provided for upload

The generator reads Conversion Master Configuration in order to determine the conversion task type associated with the table or maintenance object.

The generator creates **Conversion Tasks**, one instance per table and/or per maintenance object, creates attachments that store the artifacts and links the attachments to the Conversion Task as Related Objects.

NOTE: IMPORTANT: The regeneration is necessary each time the metadata or conversion configurations are amended.

Conversion Tasks query is provided with the base product.

You can choose to generate artifacts for table-level and/or maintenance object-level load, or for the entire system or for specific Table or Maintenance Object:

Generate Artifacts in bulk — for all Tables and/or Maintenance Objects

- Submit new batch job for Batch Control K1–CNVAG. This process performs a complete refresh of the conversion artifacts. It deletes existing and creates new Conversion Tasks for every Table and/or Maintenance Object that includes tables marked for conversion.

Generate Artifacts for an individual Table or Maintenance Object

- Navigate to **Admin —> Conversion Support —> Generate Conversion Artifacts**.
- Select one of the options:
  - **Table search** allows you to select specific target table
  - **Maintenance Object search** allows you to select specific target Maintenance Object.

NOTE: The process overrides all previously generated artifacts and replaces existing files.

Accessing the Artifacts

Navigate to Conversion Task query and search for the task that is corresponding to a table or a maintenance object. or use other available search criteria. Explore the Attachments collection to view the artifacts.

Conversion Tasks

The following sections provide information about Conversion Tasks and generated conversion artifacts.

Conversion Task Query

Conversion Task query allows to search by variety of criteria:

- Creating User, Conversion Task Type and Status
- Linked Object – Table or Maintenance Object
- Exact Conversion Task ID

Conversion Task

The purpose of the **Conversion Task** is to maintain the conversion artifacts for Table or Maintenance Object. Please consider the following with respect to conversion tasks.

- The business object lifecycle supports multiple attempts to re-generate the artifacts.
- The re-generation could be triggered either manually or by the Conversion Artifact Generator. Only one set of the artifacts is relevant at any point of time, hence only one Conversion Task at the time can exists for a given Table or Maintenance Object.
- Typically, Conversion Tasks are created by Conversion Artifact Generator. However, it can also be created manually by navigating to **Admin—> Conversion Support —> Conversion Task**.
- Use the Conversion Task Query to search for an existing entry. Select the entry from the search results to navigate to the maintenance portal to view and maintain the selected record.
- Conversion Task references a Conversion Task Type and contains standard business object record information, including the creation date/time, status update date/time and creating user.
- The task references the object – **Table or Maintenance Object** – for which the artifacts are generated.
- The collection of task's **Related Objects** store references to the Attachments that store a snapshot of the generated files. A new attachment is created after each generation run and linked to the Conversion Task for audit purposes.

Conversion Batch Controls

The following batch controls are delivered with base product:

Batch Control	Description
K1-CNVLD	Conversion - Load Data using SQL Loader. This process facilitates the legacy data extract upload using SQL Loader. The input data files are expected to be uploaded to a file storage location. See Input and Output File Location for more information about file storage locations. One run uploads the data into a single target Table or Maintenance Object. The process is multi-threaded. This process is discussed in more details in the Data Load Batch chapter
K1-CNVDS	Conversion - Disable Conversion This process resets the environment indicators and prevents further conversion activities. Once conversion is disabled, the following actions are blocked: • Switching between Conversion and Production • Table truncation • Disable Indexes and Rebuilding the Statistics • Data insertion into Production • Data upload into staging **NOTE:** The process should be executed only once and only after all of the conversion activities are completed and the application is ready for the real production use.
K1-CNVEN	Conversion - Enable Conversion This process resets the environment indicators and enables it for conversion activities. It should be executed at the start of the Conversion project.
K1-DRPIN	Conversion - Disable Indexes on Production Tables This process can be used in two modes: • Populate Table or Maintenance Object batch parameter to process specific object • If no specific target is populated, the process disables indexes on all converted tables in Production schema
K1-SDSIN	Conversion - Disable Indexes on Staging Tables This process can be used in two modes: • Populate Table or Maintenance Object batch parameter to process specific object • If no specific target is populated, the process disables indexes on all converted tables in Staging schema
K1-RIUSP	Conversion - Rebuild Indexes and Update Statistics in Production This process can be used in two modes: • Populate Table or Maintenance Object batch parameter to process specific object • If no specific target is populated, the process rebuilds indexes and updates statistics on all converted tables in Production schema • Specify the sampling size (percentage) for statistics gathering. If not specified, the system will use default dbms_stats.auto_sample_size. • Specify whether you wish to rebuild indexes, to gather table statistics or both. It is recommended to rebuild the indexes and update statistics right after the data upload.
K1-RIUSS	Conversion - Rebuild Indexes and Update Statistics in Staging This process can be used in two modes: • Populate Table or Maintenance Object batch parameter to process specific object • If no specific target is populated, the process rebuilds indexes and updates statistics on all converted tables in Staging schema

	• Specify the sampling size (percentage) for statistics gathering. If not specified, the system will use default dbms_stats.auto_sample_size • Specify whether you wish to rebuild indexes, to gather table statistics or both It is recommended to rebuild the indexes and update statistics right after the data upload.
K1-CLNTB	Conversion - Truncate Tables in Production This process can be used in two modes: • Populate Table or Maintenance Object batch parameter to process specific object • If no specific target is populated, the truncates all converted tables in Production schema
K1-SCLTB	Conversion - Truncate Tables in Staging This process can be used in two modes: • Populate Table or Maintenance Object batch parameter to process specific object • If no specific target is populated, the truncates all converted tables in Staging schema
K1-CLNRT	Conversion — Truncate XML Resolution Tables This process is allowed to run only in the Staging schema and may be used in two modes: • Populate Table or Maintenance Object batch parameter to process specific object • If no specific target is populated, the truncates all XML Resolution tables in Staging schema
K1-CLNKY	Conversion - Truncate Key Reference Tables This process is allowed to run only in the Staging schema and may be used in two modes: • Populate Table or Maintenance Object batch parameter to process specific object • If no specific target is populated, the truncates all Key Reference tables in Staging schema

Configuring Data Load Batch Controls for Conversion Project

The generic batch process K1-CNVLD performs the load of the input data file(s) into a target table or maintenance object, specified as one of the batch parameters.

Consider creating individual batch controls per object (table or maintenance object).

It allows leverage the parallel processing capacity of the system; the data load processes for all objects can be submitted near-simultaneously, thus minimizing the time needed for the legacy data upload, which is only an initial step in the overall data conversion process.

Use batch control K1-CNVLD for rehearsal load batch run for various tables and/or maintenance objects and determine the optimal load strategy:

- Whether the data should be extracted and loaded as single Tables and/or Maintenance Objects
- Whether and how to split the data files and the number of threads for the batch
- Potential log file size and the feasible logging level

Configure data load Batch Controls for each individual Table or Maintenance Object. Pattern the configuration after K1-CNVLD and set the default parameter values for:

- **Input, Output and Archived File Storage Locations**. The location (compartments and buckets) has to be defined in advance in the cloud object storage; batch parameters are referencing Extendable Lookup ***F1–FileStorage*** value(s) corresponding to these locations. The parameter value should be composed as **file-storage://**<Extendable Lookup Value>/<object storage bucket name>, for example: *file-storage://OS-SHARED/CONV-Input*
- Target **Table** or **Maintenance Object** Either one of these two parameters must be populated.
- **File Extension**. When specifying the **file extension**, include all the extensions available. For example. if the file name is ***XXX.csv.gz,*** enter parameter value ***csv.gz***
- **Log Level.** Specify the value *LOG* to force SQL Loader to produce the most detailed data upload process log.

- **Retain Input Option**. This parameter controls whether the original input data files should be left ‘as is’ at the input data location or purged from the object storage or moved to the archived data location. When the input data is archived the timestamp is appended to the original file name.

  NOTE: Note: if the batch process was not able to upload the input data file, this file remains at the input data location.

- **Max Number of Upload Errors**.

  The value of this parameter pertains to the desired errors threshold for the SQL Loader. The load process halts after the specified number of records in error is reached. The SQL Loader log contains a message SKIPPED=NNN where NNN marks the position of the last record in error in the input data file. The process may then be resumed from this position if the batch is submitted with the ***skip*** parameter populated with the value of SKIPPED from the log.

Default **Thread Count** depends on the type of the uploaded table. Regular tables can be uploaded in a single or multiple threads. Key tables must be uploaded in a single thread

Input Data Files

This section provides information on inputting data files for data conversion.

Overview

This section provides information on the requirements and details for file upload, loading table, maintenance object, CLOB, layout, data formatting, and so on related to inputting data files for data conversion.

File Creation and Upload

The extract data may be uploaded in both compressed and decompressed form.

Supported compressed formats include zip and gzip. The files with extension **.gz** and **.zip** will be automatically decompressed by the system.

NOTE: For the further details on supported compressed formats please refer to the Oracle Utilities Application Framework documentation.

File Creation

Use Conversion Task query to retrieve conversion task(s) for table(s) or maintenance object(s) that you are planning to convert. Review input data specifications. Create the data extract text file, using field delimiters, enclosing characters and data formatting according to the specifications.

If the files were created under Windows operating system, it is recommended to remove the carriage return character prior to compression (not applicable for secondary files with CLOB data).

File Compression with gzip

Use gzip utility to compress the file.

- For Windows operating system, download and install gzip utility for Windows. Run the following:

```
# Navigate to the gzip folder
cd C:\gzip-1.3.12-1-bin\bin
# Run gzip.
# New file will be created in the same directory as the original the input file
```

```
gzip -k <full path to the input data file>
# Sample command for the file TABLE_NAME.txt located under
# C/myDataDirectory/:
gzip –k C/myDataDirectory/TABLE_NAME.txt
```

- For Unix/Linux operating systems, run the following command:

```
gzip < <inputFile name> > <outputFile name with .gz suffix>
# Sample command for the file TABLE_NAME.txt
gzip < TABLE_NAME.txt > TABLE_NAME.txt.gz
```

Loading Table or Maintenance Object

The conversion data upload supports two types of the extract:

- Data extract that contains the data for a single target staging table.

  In this case, the upload process inserts records into a single staging table. The input data file consists of similar records, each represents a row in the target table.

- Data extract that contains the data for the group of target staging tables that represent a Maintenance Object.

  In this case, the upload process populates all the tables that belong to a maintenance object. The input data file contains multiple types of records that are inserted into various tables.

  NOTE: If the data is uploaded for a Maintenance Object, each record has to be prefixed with the table name that is used as record type indicator. This requirement is also mentioned in the input file specification

- The application stores the large volumes of data in partitioned tables. If the Maintenance Object contains partitioned tables, the data should be uploaded in a certain order:
  - During the data upload, the primary table in the Maintenance Object should be loaded first, then the indexes should be rebuilt, followed by the data upload into the child tables

Multiple Data Files

The large volume of the converted data may require the extract to be split into multiple parts.

In this case the multi-part extract for a target staging Table or Maintenance Object is loaded with a multi-threaded batch run.

Sample scenario:

10M of the Person records are being converted. The extract file is split into 10 files, PERSON1, PERSON2 PERSON3 etc., each file contains 1M records. All 10 files are uploaded to the input file location.

Batch process ‘Load Person data’ is submitted with 5 threads; each thread picks up 2 files and uploads the data into staging tables.

Loading CLOB Data

The CLOB data can be loaded in two ways:

- Extracted as part of the main data file. This method is suitable for the relatively small CLOB data size. Refer to Limitations for more details.

- Extracted into a separate file. This method is suitable for larger CLOB data sizes and also for large volume tables or maintenance objects. In this scenario, the secondary file has to be uploaded at the same time as the primary data file.

> **NOTE:** When loading CLOB data as a secondary file, a separate file is created for each CLOB field in the. It means that if the table 'XXX' contains CLOB fields 'clob1' and 'clob2' , the system would expect to process files XXX, XXX_clob1 and XXX_clob2. See also Notes on Secondary Data Files

Notes on Secondary Data Files (CLOB Data)

Secondary data file is loaded simultaneously with the primary data file. The CLOB data extract should contain exactly the same number of records as a primary file.

> **NOTE:**
> When splitting the extract into multiple data files, make sure that the numbers of extracted records in the primary and secondary files are matching.
>
> It is applicable for both table-level and maintenance-object level load.

Sample Scenario:

Table CI_PER includes CLOB column C1_PER_DATA_AREA.The CLOB field is used to store customer's resume.

The legacy table contain 10K records and the CLOB data is available for some of the records

The primary data file, CI_PREM.csv contains 10K lines, one line per record

The secondary data file, CI_PREM_CLOB, contains 10K records too. The records with the empty CLOB are represented by CLOB delimiter.

Primary File – Regular Data Fields	*Secondary File - CLOB*
12345678,John,Doe,.....	*John's resume <endclob>*
45165609,Smith,Mary,.....	*Mary Smith resume<endclob>*
33336733,Brand,Brent,.....	*<endclob>*
06737482,Sunshine,Julie,.....	*Julie Sunshine resume<endclob>*

Splitting single table data extract with CLOB

The following example illustrates how to split the single table large volume extract correctly if the CLOB data is supplied in the secondary file:

Assuming the table has 10K records, split into two equal parts. The following files should be created:

- Primary File 1 with 5K records
- Secondary File 1 with 5K CLOB entries
- Primary File 2 with 5K records
- Secondary File 2 with 5K CLOB entries

Splitting maintenance object data extract with CLOB

The following example illustrates how to split the maintenance object large volume extract correctly if the CLOB data is supplied in the secondary file:

Assuming the maintenance object has two tables, one of the tables includes CLOB column and the primary table has 10K records. The data supposed to be split into two equal parts. The following files should be created:

- Primary File 1 with 5K records for the primary table and corresponding records for the child table
- Secondary File 1 with CLOB entries corresponding to the table in the Primary File 1
- Primary File 2 with 5K records for the primary table and corresponding records for the child table
- Secondary File 2 with CLOB entries corresponding to the table in the Primary File 2

NOTE:
For Maintenance Object-level load consider supplying CLOB data in the primary file, to avoid complicated logic of data splitting.

Refer to CLOB Data - XML for limitations concerning XML CLOB data

File Layout and Data Formatting

The input data specifications file for each Table or Maintenance Object that are subject to conversion are stored as an attachment linked to the corresponding conversion tasks. The Conversion Artifact Generator creates conversion task for each table and/or maintenance object.

Specifications are generated based on the metadata and the conversion configurations defined on Conversion Task Type and Conversion Master Configuration. Configurations control delimiters and enclosing characters and define CLOB data processing rules.

The data extract file is expected to contain one record per each target table row. The data field order in the record has to be exactly the same as it is listed in the specification file.

The specifications contain:

- Data delimiters and expected secondary data file names
- Field-by-field data formatting rules for

  data types and size, Date, DateTime and Time formatting
- Special notes on CLOB fields

The fields in the record are expected to be separated by a delimiter character (or string).

For the not-nullable columns with no data, a single space enclosed in the enclosing character (or string) has to be supplied, for example “ “

The CLOB data in the main record has to be enclosed in the CLOB Delimiter string, for example *<clobend>some clob data<clobend>*

The CLOB records in the secondary data file has to be separated by CLOB Delimiter string followed by the new line sign, defined as the CLOB Line Separator. IMPORTANT! the last entry in the CLOB data file has to end with the new line, too

- If the data extract is created on Linux, use Line Feed to separate CLOB records and make sure that there is no space after CLOB Delimiter string
- If the data extract is created on Windows, use Line Feed followed by Carriage Return.

Enclosing character (or string) can also be used to encapsulate data that includes a field delimiter, for example *“Hello, World”*

Defining Custom Data Delimiters and Enclosing Characters

The default configurations are delivered with the base product.

Your entire legacy data or a specific table, or the data extract mechanism may require use of a different data delimiter and/or enclosing characters. To enable alternative delimiters, perform the following steps:

- Define new data delimiter string, enclosing string, CLOB delimiter string and/or CLOB Line Separator using the corresponding Extendable Lookups
- Configure Conversion Task Type using newly created delimiters
- Amend Conversion Master Configuration, setting default (for all tables or maintenance objects) or override (for specific table or maintenance Object) instructions.
- Regenerate Conversion Artifacts

Now you can create data extract according to the newly generated specifications.

Input Data for a Staging Table

The extract for a single table contains delimited data records. Depending on the conversion configuration, the CLOB data should be supplied either inline or in a separate, secondary file.

Input Data File Names

Input data file names has to follow the rules described in this chapter. The names are case-sensitive.

File Type	File Name
Single data file per table, no CLOB	File name equal target table name, i.e. CI_PREM
Single data file per table, **single** CLOB column in the table, CLOB data in secondary file	Primary file name equal target table name, i.e. CI_PREM Secondary file name is composed as target table name concatenated with '_CLOB', i.e. CI_PREM_CLOB
Single data file per table, **multiple** CLOB columns in the table, CLOB data in secondary file	Primary file name equal target table name, i.e. CI_PREM **For each CLOB column** being uploaded, the secondary file name is composed as target table name concatenated with CLOB column name concatenated with'_CLOB', i.e. CI_PREM_COL1_CLOB, CI_PREM_COL2_CLOB
Multiple data files per table, no CLOB	File name is composed as target table name concatenated with file number, i.e. CI_PREM1, CI_PREM2 etc
Multiple data files per table, CLOB data in secondary file	Primary file name is composed as target table name concatenated with numeric file number, i.e. CI_PREM1, CI_PREM2 etc Secondary file name is composed as target table name concatenated with '_CLOB' and file number, i.e. CI_PREM_CLOB1, CI_PREM_CLOB2 etc
Multiple data files per table, **multiple** CLOB columns in the table, CLOB data in secondary file	Primary file name is composed as target table name concatenated with numeric file number, i.e. CI_PREM1, CI_PREM2 etc **For each CLOB column** being uploaded, the secondary file name is composed as target table name concatenated with CLOB column name concatenated with'_CLOB' and file number,, i.e. CI_PREM_COL1_CLOB1, CI_PREM_COL1_CLOB2, CI_PREM_COL2_CLOB1, CI_PREM_COL2_CLOB2 etc

Input Data for Maintenance Object

The extract for a maintenance object contains delimited data records for all tables included in the maintenance object. Depending on the conversion configuration, the CLOB data should be supplied either inline or in a separate, secondary files, one file per table containing CLOB. Refer to Notes on Secondary Data Files (CLOB Data) for limitations concerning XML CLOB data and more details.

Input Data File Names

Input data file names has to follow the rules described in this chapter. The names are case-sensitive.

File names for Maintenance Object contain Maintenance Object Code. If the code includes spaces or special characters, it should be replaced with '_', for example:

For actual MO Code: SP/MTR HIST, in the file name, use SP_MTR_HST

File Type	File Name
Single data file per Maintenance Object, no CLOB	File name equal target maintenance object name, i.e. PREMISE
Single data file per Maintenance Object, **single** CLOB column in the table**,** CLOB data in secondary file	Primary file name equal maintenance object name, i.e. PREMISE Secondary files name composed as maintenance object name concatenated with table name concatenated with '_CLOB'. For example, if PREMISE's CI_PREM table contains CLOB, the secondary file name is: PREMISE_CI_PREM_CLOB
Single data file per Maintenance Object, **multiple** CLOB columns in the table, CLOB data in secondary file	Primary file name equal maintenance object name, i.e. PREMISE **For each CLOB column** being uploaded, the secondary files name composed as maintenance object name concatenated with table name concatenated with CLOB column name concatenated with '_CLOB'. For example, if PREMISE's CI_PREM table contains two CLOB columns COL1 and COL2 , the secondary file names are: PREMISE_CI_PREM_COL1_CLOB, PREMISE_CI_PREM_COL2_CLOB
Multiple data file per Maintenance Object, no CLOB	File name is composed as target maintenance object name concatenated with file number, i.e. PREMISE1, PREMISE 2 etc
Multiple data file per Maintenance Object, single CLOB column in the table, CLOB data in secondary file	Primary file name is composed as target maintenance object name concatenated with file number, i.e. PREMISE1, PREMISE 2 etc Secondary files name composed as maintenance object name concatenated with table name concatenated with '_CLOB' concatenated with file number. For example, if PREMISE's CI_PREM table contains CLOB, the secondary file names are: PREMISE_CI_PREM_CLOB1, PREMISE_CI_PREM_CLOB2, etc
Multiple data file per Maintenance Object, **multiple** CLOB columns in the table, CLOB data in secondary file	Primary file name is composed as target maintenance object name concatenated with file number, i.e. PREMISE1, PREMISE 2 etc **For each CLOB column** being uploaded, the secondary files name composed as maintenance object name concatenated with table name concatenated with column name concatenated with '_CLOB' concatenated with file number. For example, if PREMISE's CI_PREM table contains two CLOB columns COL1 and COL2 , the secondary file names are: PREMISE_CI_PREM_COL1_CLOB1, PREMISE_CI_PREM_COL1_CLOB2, PREMISE_CI_PREM_COL2_CLOB1, PREMISE_CI_PREM_COL2_CLOB2 etc

Limitations for Input Data

Key Tables

Due to technical limitations, Key tables cannot be loaded in multi-threaded mode. If you decide to create input data files using the Maintenance Object method, consider the following:

- Create data extract(s) for a Maintenance Object data, but exclude the Key table(s) and run data upload in multi-threaded batches
- Create separate extract for the Key table(s) only and upload them in single threaded batches.

Duplicate Records

Duplicate records are not rejected during the upload. This may result in the index corruption and other issues in the target table(a).

The default SQL Loader data upload method configured in the system is Direct Path. Among the advantages of a Direct Path method are higher performance and more efficient resources utilization. Constraints are disabled during Direct Path load and as a result the duplicate PK records are not rejected.

There are two approaches to address the issue

- **Clean up the duplicate records from the legacy data extract BEFORE the upload**
- Customise the default configuration and switch to the Conventional Path load method.
  - **Create custom Control File Header**: Duplicate Managed Content K1-SQLLoaderControlFileHeader .and modify to use Conventional Path Load (Consult SQL Loader documentation for more detailed information).
  - **Modify Conversion Instructions** (Conversion Task Type): specify SQL Loader Control File Header created above
  - **Regenerate Conversion Artifacts**

CLOB Data Size

The maximum size of the CLOB data included in the main data file is 10K. If converted CLOB data size is expected to be larger than 10K, create secondary data files.

CLOB Data - XML

The XML data formatted using new line characters cannot be uploaded as part of the primary data file. If your CLOB data includes formatted XML, you should create secondary data file(s) for CLOB columns.

Conversion Orchestration

This section provides information on orchestrating a data conversion.

Input and Output File Location

The following locations has to be defined in order to support legacy data upload:

- File location for input data files upload
- File location for SQL Loader output files and also for archived input data files

See Object Storage chapter for more details on how to define the file location.

Understanding Load Data Process

The setup of the batch controls that is necessary to support data conversion and legacy data extract upload is described in Data Load Batch Controls .

The following sections explain the process of legacy data extract upload with SQL loader.

Load Data Batch Run Parameters

The data upload batch is processing single target object – Table or Maintenance Object. The data extract may be supplied in one or multiple files, created and named according to the Input File instructions for a Staging Table and Maintenance Object .

The process is multi-threaded and optimized for the parallel load.

Using the batch parameters you can control:

- Level of logging
- Data extract retention
- Resume the interrupted load
- Maximum number of load errors

Multi-Threading

The process is threading by dividing the multiple data files between the threads. Each thread in turn, is processing its files one by one.

The exception is Key Tables. They are index-organized tables and have to be loaded in the single thread, due to the SQL Loader restrictions.

Load Outcome – Log, Bad and Rejected Files

SQL Loader output files are available for download after each run . Refer to Input and Output File Location for details. To ensure file name uniqueness, the output file names are concatenated with the current timestamp.

- **Log file** contains the information about the run, including number of record processed, errors encountered and other useful information. The level of logging is controlled by the batch parameter ***log*** .
- **Bad** and **Rejected** files contain the records that were rejected by either the SQL Loader or the database. If the errors are not extensive and could be fixed manually, these files can be edited, renamed and used as an input again.

Interrupted Load

The data load process can be re-started. Once SQL Loader reached the maximum number of errors allowed for the run, it records the position of the last record in error in the log. The restarted load begins pulling the data from the file after skipping certain number of records.

In order to continue processing interrupted load, perform the following steps:

- Retrieve the log file. Find statement '*Specify SKIP=**n** when continuing the load.*' in the log file.
- Re-upload the input data file to the server
- Submit the batch once again, this time specifying the ***n*** as a ***skip*** batch parameter

Tables Cleanup and Index Maintenance

The SQL Loader is running in parallel mode in order to maximize the performance. Therefore the target tables are not emptied prior to the data insertion and it has to be done implicitly, by running corresponding batch jobs.

Indexes are disabled during the data upload. The product provides batch processes that support index disabling/re-building and statistics update.

- User batch control ***K1-SCLTB*** to truncate tables in **Staging** area and batch control ***K1-CLNTB*** to truncate tables in **Production**
- Use batch control ***K1-SDSIN*** to disable indexes in **Staging** area and batch control ***K1-DRPIN*** to disable indexes in **Production**
- Use batch control ***K1–RIUSS*** to rebuild indexes and/or update table statictics in Staging and ***K1–RIUSP*** to perform the same in **Production**
- Use batch control ***K1-CLNKY*** to truncate **Key Reference** tables in **Staging** area
- Use batch control ***K1-CLNRT*** to truncate **XML Resolution** tables in Staging Area
- Use batch process ***K1–SDSTG*** to disable Triggers in **Staging** and batch control ***K1–PDSTG*** to disable triggers in **Production**
- Use batch process ***K1–SENTG*** to enable Triggers in **Staging** and batch control ***K1–PENTG*** to enable triggers in **Production**

The supported scenarios are:

- **Truncate Specific Table or Maintenance Object**: In order to truncate the specific table, submit the corresponding batch job, specifying ***table*** parameter. In order to truncate the data that stored in a specific Maintenance Object, submit the corresponding batch job, specifying ***maintenanceObject*** parameter. The process will also disable indexes in the truncated tables.

  NOTE: If the truncated table's metadata defines a Key table, the Key table gets truncated too.

- **Truncate all Converted Tables**: In order to truncate all converted tables run the corresponding batch job without specifying ***table*** or ***maintenanceObject*** parameters. The process will disable indexes and truncate all tables marked for conversion in the metadata.
- **Truncate Key Reference Tables or XML Resolution Tables for Specific Table or Maintenance Object**: In order to truncate Key Reference table linked to a staging table, submit the corresponding batch job, specifying ***table*** parameter. In order to truncate Key Reference tables for a specific Maintenance Object, submit the corresponding batch job, specifying ***maintenanceObject*** parameter.
- **Truncate all Key Reference Tables or XML Resolution Tables**: Submit the corresponding batch job without specifying ***table*** or ***maintenanceObject*** parameters to truncate all Key Reference tables.
- **Disable Indexes for Specific Table or Maintenance Object**: In order to disable indexes for the specific table, submit the corresponding batch job, specifying ***table*** parameter. In order to disable indexes for the specific Maintenance Object's tables, submit the corresponding batch job, specifying ***maintenanceObject*** parameter.
- **Disable Indexes for all converted Tables**: In order to disable indexes on all converted tables run the corresponding batch job without specifying ***table*** or ***maintenanceObject*** parameters. The process will disable indexes and truncate all tables marked for conversion in the metadata.

Load Performance Tuning

The data specifics – volumes, structure and other aspects – may justify customizing the default configurations for all or some tables.

The SQL Loader is processing data according to the Control File. The fragment that contains general load parameters is stored as Managed Content that is referenced on the Conversion Task Type.

You can create new Managed Content, configure the Conversion Task Type and specify the new settings on Conversion Master Configuration, for all or specific Tables or Maintenance Objects.

Understanding the Conversion Orchestration Process

The following sections provide information on general aspects of conversion orchestration.

Enabling Conversion

Conversion activities are possible as long as conversion is enabled in the environment. The authorized Conversion Administrator may run the corresponding batch job to enable conversion.

Switching between Staging and Production Schema

In the environment enabled for conversion, the online application is running against the database schema that contains synonyms pointing to either Staging or Production tables.

The synonyms could be switched by running the corresponding batch job.

The switching is possible only while Conversion is Enabled

Navigate to **Admin –> Conversion Support –> Switch Schema** to perform the action.

Preparing for Conversion

The preparation steps include evaluating initial conversion configurations provided by the product, customizing the configurations if needed, and finally, generating conversion artifacts.

The preparation should be done when the environment is running against Production schema

Conversion Development

The creation of the legacy data extract and rehearsal of the data load and subsequent conversion data processing is done while the environment is running against Staging schema.

Consider creating and scheduling the sequence of the batch processes for both data load and following validation and insertion.

Uploading Data

SQL Loader implicitly disables the indexes in the non-partitioned tables before the upload. In the partitioned tables the indexes needs to be explicitly disabled by the batch process prior to the upload. Once the data upload is completed the indexes needs to be rebuilt.

You may combine the table truncation, data upload and the index maintenance and statistic updates into a batch chain and create a dedicated batch job chain for each converted Table or Maintenance Object.

Conversion Run

For the mock-up, or dress rehearsal or the actual go-live conversion run you may choose to amend data load batch controls for tables/maintenance objects to improve performance:

- Reduce the logging level – set ***log*** batch parameter to NOLOG
- Do not retain input data – set ***retaininput*** batch parameter to PURGE

Disabling Conversion

After conversion is completed, the various activities such as table truncation and index maintenance should not be allowed.

Upon conversion project completion, switch the schema to Production and request that the authorized Conversion Administrator run the corresponding batch job to disable Conversion in the environment.

Enabling Incremental Conversion

The legacy data conversion may be done in stages. After the first chunk is converted and the system begins running in production, you may need to convert the next portion(s) of the data. This orchestration macro-scenario is called *incremental conversion*; it imposes certain limits and restrictions on the conversion activities and it has to be approved by the management.

If the incremental conversion has been approved , request that the authorized Conversion Administrator run the corresponding batch job to enable the incremental conversion in the environment.

Security Setup for Conversion

The services associated with data conversion are combined into the following user groups:

- **Conversion Administrators** includes the services linked to the most sensitive database operations: enable/disable conversion and switching database synonyms between production and staging schema
- **Conversion Development** includes the services associated with conversion-related configurations
- **Conversion Operations** provides access to services associated with data load batch processes. This group could be extended to include services associated with specific table(s) or maintenance object data loads

These user groups are included in the base product. Your implementation may chose to use it as is or create a different setup.

Conversion Reconciliation Reports

Your cloud service subscription includes online database querying capabilities via BI Publisher and SQL Developer Web. It can be used to create and run data reports against both production and staging schema tables. Contact your Security Administrator for the access information.

Batch Operations Support for Cloud Implementations

This chapter provides information and requirements on batch operations in support of Oracle Utilities Cloud implementations.

Batch Operations Overview

Batch Operations provides a user interface to support Oracle Utilities Application Framework (OUAF) integration with Oracle Scheduler. It facilitates customers and implementers to define, manage and schedule stream of batch jobs to run at periodic intervals using a user friendly interface. In addition, users can monitor the progress of the streams, each of the steps and also perform required exception handling.

Note. Batch Operations user interface can only be used if Oracle Scheduler is used for scheduling and managing batches. Use of third party solutions or schedulers is not supported in current release.

Batch Operations is supported by the following system entities:

- Batch Job Stream Definition – a Business Object based on the Batch Job Stream Definition maintenance object
- Scheduler Program – a Business Object based on the Scheduler Program maintenance object

Batch Operations - Oracle Scheduler Integration

The Oracle Database includes an enterprise wide scheduler that can be used to simplify the scheduling of background/batch jobs. The scheduler is implemented by the DBMS_SCHEDULER package. Oracle Scheduler is a sophisticated feature that provides functionality ranging from various scheduling methods of background jobs to job prioritizations to supporting clustering environments.

Refer to **Batch Scheduler Integration for Oracle Utilities Application Framework** (Doc ID 2196486.1) on My Oracle Support for more information about integration with Oracle Scheduler.

Batch Job Stream Definitions

This section provides information on Batch Job Stream Definitions.

Defining a Batch Job Stream Definition

In order to define a Batch Job Stream Definition, from the **Admin** menu, select **Batch Operations**, select **Batch Job Stream Definition**, and select **Add**.

A Batch Job Stream Definition includes specifying the batch jobs and the order in which they should be run as “Stream Steps”, the user to use to run these batch jobs and a “Schedule” to state when and how often the batch job stream should run. Batch jobs can be setup to run in sequence or in parallel fashion or a combination of both.

The batch job to run for each stream step can be either specified directly or via a “Scheduler Program”. A scheduler program is an OUAF counterpart of DBMS program in Oracle Scheduler. Refer to Configuring a New Scheduler Program for more details.

A Stream step includes specifying the following details

- Step Name: the name of the step
- Batch Control: the batch control to run for that step.
- Thread Count: the number of threads to use to run this step’s batch control
- Scheduler Program: the scheduler program to use for that step. Either a batch control or scheduler program can be specified but not both
- Step Conditions: specifies when the current step should run. It includes the following details
  - Step Name: the step that should have run prior to current step
  - Step Status: the status the prior step should be in when its finished
  - Step Condition: indicates how (And/Or) prior step should combine with next subsequent step conditions. For example, if step C should execute after step B and A have succeeded, then step C’s has two step conditions 1)A Succeeded And 2)B Succeeded.

A Batch Job Stream Definition’s schedule can be specified using various frequencies.

- Daily, By Minutes: run every ‘x’ minutes every day except on specified excluded days
- Daily, By Time: run at certain times every day except on specified excluded days
- Monthly, By Month Day Number: run on a specific day of the month, at specified time
- Monthly, By Week Days: run on days of a specific week in a month, at specified time
- Weekly: run every ‘x’ weeks on specific days, at specified time
- Yearly, By Month Day Number: run on a day of the month, at given time in specified months

- Yearly, By Week Days: run on days of a specific week, at given time in specified months
- Custom: run on a specific date at specified time

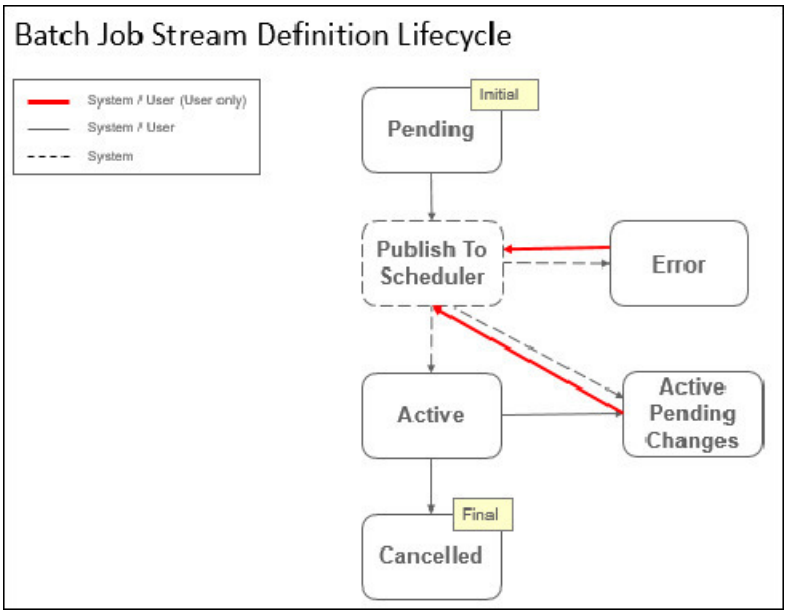
A Batch Job Stream Definition can also be created without defining a schedule. In this case, it will execute immediately when it is sent to Oracle Scheduler.

There are custom ways in which a batch job stream definition can be setup. This involves creating a Scheduler Program. Refer to Tips For Setting Up Batch Job Stream Definitions and Scheduler Programs for more information.

Activating a Batch Job Stream Definition

Once the Batch Job Stream Definition is ready, it can be activated in Oracle Scheduler to run at periodic intervals by transitioning the Batch Job Stream Definition to Active state. An activated Batch Job Stream Definition is called a Batch Job Stream. A Batch Job Stream runs based on the schedule defined on it.

NOTE: A Batch Job Stream with no schedule runs immediately and only once when activated.



By default, every batch job in a Batch Job Stream Definition is internally setup to run in such a way that if that batch job fails the Batch Job Stream run stops execution. User then has an option to fix the issue and re-start the stream.

Updating an Active Batch Job Stream Definition

A Batch Job Stream Definition can be changed in two ways while Active and in Active Pending Changes states. While active, any changes made to the Batch Job Stream Definition are immediately published to the Oracle Scheduler to reflect on the next Batch Job Stream run. It is recommended that any simple changes to Batch Job Stream Definition be done in Active mode.

Changes that are complex or require thorough review can be made by transitioning the Batch Job Stream Definition to Active Pending Changes state. In this state, changes are saved in OUAF and not immediately published to Oracle Scheduler. Once changes are made and reviewed, the changes can be published to Oracle Scheduler by transitioning to Active state.

Changes to a Batch Job Stream Definition can be made by editing it or using the edit options in the Batch Job Stream Definition steps tab.

Inactivating a Batch Job Stream Definition

A Batch Job Stream can be inactivated from running further by cancelling it. Cancelling a Batch Job Stream Definition removes its information from the Oracle Scheduler.

Migrating Batch Job Stream Definitions

Batch Job Stream Definitions can be migrated to other environments using the CMA (Configuration Migration Assistant) process or the Process Automation Tool. A Criteria Based Migration Request needs to be created as a pre-requisite before migrating the Batch Job Stream Definitions.

Once the Batch Job Stream Definitions are migrated to other environments, any Scheduler Programs that are referenced by Batch Job Stream Definitions should be activated in the target environment for a corresponding program to be created in the Oracle Scheduler. If the scheduler program already exists in target environment, then the scheduler program should be re-activated so latest changes can be sent to Oracle Scheduler.

Once all referenced scheduler programs are activated, the respective Batch Job Stream Definitions should be activated so the stream definition can be created and activated in Oracle Scheduler to run periodically.

If a Batch Job Stream Definition already exists in the target environment and is in Active state, the status is set to Active Pending Changes when the Batch Job Stream Definition is migrated. The Batch Job Stream Definition should be manually re-published/activated so the stream updates can be sent to Oracle Scheduler.

Tips For Setting Up Batch Job Stream Definitions

Configuring job streams to avoid open-ended stream issue

The job streams should always be configured such that the stream “ends” only once all the steps in the stream are completed either by configuring it on the “end of stream” step condition or by defining the inter-dependencies between the steps. This ensures the stream run execution does not stop in the middle of the stream.

Jobs that run past midnight must have same business date as other jobs in a stream

In situations where a stream runs too long and has some jobs run past midnight, all jobs in the stream can be set to run with same business date. Create a scheduler program to set the business date program option value to ‘SYSDATE’ and job stream scope set to the stream name. Use this scheduler program as the initial step on the stream before other steps.

How to split a big stream into multiple smaller stream

If the Batch Job Stream Definition is too long with many jobs to run, it can be split into multiple smaller streams such that one stream runs after the other. This can be done using a new batch control K1-EXDJS (Execute DBMS Job Stream). Create a Scheduler Program for every sub stream, specify the batch control as K1-EXDJS, batch parameter name as the sub stream name. Use this scheduler program as a last step on the previous stream before it ends so that as soon as the previous stream’s steps are completed, the last step will initiate the new sub stream. For example, if you have a stream with jobs A,B,C,B,E,F,G,H. Split the stream into stream 1 with A,B,C,D and stream 2 with E,F,G,H. Create a Scheduler Program with batch control K1-EXDJS, batch parameter name as stream 2. On stream 1 after step D, define a new step that invokes the created Scheduler Program. Note that there are no dependencies between sub-streams. That is each sub stream would be kicked off by the new batch control to run on it’s own and the stream that called the sub stream does not wait for it to finish. So such configuration is to be used with only those steps that run in parallel.

How to configure streams to run only on OUAF working calendar dates

To setup Batch Job Stream Definition to run only on working days, use the batch control K1-CWCEL. Create a Scheduler Program with this batch control, batch parameter Work Calendar set to required work calendar, Verify that **Run On Business Day** is set to true. Use this Scheduler Program as the first step to run on the stream. Define the stream to end if

this step fails. Define the stream's subsequent step to execute if this step succeeds. When the stream runs on a scheduled day, the first step checks if the day falls on a working day as per the work calendar. If the day is not a working day then the stream ends execution ensuring stream runs only on desired dates.

Job streams that check existence of a file

In situations where a job stream needs to check if a file has been uploaded/created by another process before it can continue with rest of it's processing, a special batch program "K1-OSCFE" is available. A step can be introduced referring to this new batch program with parameter specifying which object storage bucket to check the file in and also status to return if the file does not exist. This step can be added right before the step that has dependency on checking the existence of the file and continue processing or not.

Handling job streams inter-dependencies

In situations where a job stream should not run when another stream is running, the inter-dependencies between such job streams can be configured by using a special batch program "K1-CKCCR". Following steps are needed to setup such configuration.

- The new batch program accepts two parameters script name and script data. Write a CM script that checks whether a dependent job stream is running. If it's running, then the script should terminate with an error.
- On the batch job stream that needs to be paused/hold if another batch job stream is running, add a new step as the first step of the job stream. Use the new batch program on the first step(by setting the CM script as the input parameter) or create a scheduler program if the batch control is to be used across many job streams with different CM service scripts as input parameters. Set the dependent job stream name as the script data parameter.
- Update the "End Of Stream" step condition such that if first step has failed, end the stream. This ensures the batch job stream does not run if the other dependent job stream is already running.
- Update the step(s) intended to execute after the first step, the step condition that if first step succeeded only then these steps should continue.

Note. The special batch program is not restricted to only these situations. It can be used in many custom scenarios by writing the corresponding CM script that terminates if a desired situation is met.

Using FW DBMS Scheduler REST APIs

OUAF provides various REST services to directly interact with the DBMS scheduler. While batch operations uses the underlying services of these APIs to interact with the DBMS scheduler, there are certain restrictions in calling the REST services from outside the system or via a third party integration.

A copy of the batch job stream definitions created in batch operations is kept in OUAF along with publishing them to DBMS scheduler. Any changes to such batch job stream definitions such as step changes, schedule changes etc. must be done through the batch operations UI itself. The OUAF REST services should not be used directly for changing the definitions as this can cause the OUAF job stream definitions go out of sync with the definitions inside the DBMS scheduler. This can result in a risk of losing historical data tracking, logging, risk of losing definitions and re-work etc. It is advised that only those OUAF DBMS Scheduler REST services that pertain to handling job stream runs, not definitions be used from outside of the system or via third party integrations. Such services involve querying on the job stream run status, details, make adhoc submissions of the job stream run, cancelling a job stream run and getting past job stream runs.

Scheduler Programs

This section provides information on using scheduler programs.

Configuring a New Scheduler Program

In order to configure a new Scheduler Program, from the **Admin** menu, select **Batch Operations**, and select **Scheduler Program**. Use the **Scheduler Program** portal to search for and view existing scheduler programs.

A Scheduler Program is an entity that holds details of a batch control. A Scheduler Program is a counterpart of DBMS program in Oracle Scheduler. Every batch control's batch job can be run by Oracle scheduler only if it has a corresponding DBMS program defined. When a batch control is specified on a Batch Job Stream Definition a corresponding DBMS program for it is created in Oracle Scheduler. It is created in such a way that if the batch job run fails the step fails and the batch job stream stops processing. This is done by defaulting the scheduler exit status to 30.

A special Scheduler Program can also be created to set a handful of options on the Oracle Scheduler and change their default behavior. These options can be set at a GLOBAL level to affect all programs in Oracle Scheduler or they can be confined to specific batch control or batch job stream definition.

When confined to a specific batch control, the options are applicable to all scheduler programs created for that batch control across all batch job stream definitions. When applied to a specific batch job stream definition, the options are applicable to all scheduler programs part of that batch job stream definition. In addition, the options can be applied to a specific batch control on a desired batch job stream definition.

Some of these options are also available on a Scheduler Program specific to a batch control and can be overridden on it. For example, if a batch control has a parameter "thread pool" then the thread pool can be overridden on its Scheduler Program.

Scheduler Program Options

The following options are available to set in the Oracle Scheduler.

- **Business Date**: The default value is the database's current system date, typically useful in cases where jobs run past midnight so same date can be maintained for all job runs. The format is 'YYYY-MM-DD'. A business date can also be set as SYSDATE or its variance for example SYSDATE-1 or SYSDATE-2.
- **User ID**: The user ID to use when a batch job is submitted.
- **Poll Seconds**: The frequency at which batch tables are checked for any updates. Default value is 1 second, but if this adds too much overhead then it can be changed to anywhere between 1 to 60 seconds.
- **Notify Job Name**: DBMS job OUAF_NOTIFY is the default thread notification job. That is defined out of the box to send emails for every failed thread. If that behavior is undesired, a custom thread notification DBMS job can be created and it defined as the default handler with this option. The new notify DBMS job must exist.
- **Thread Notifications**: Default value is true and sends thread notifications for failed threads. Can be set to false to suppress the notifications.
- **Thread Pool**: Default value is DEFAULT, states the thread pool to use for all batch jobs.
- **Discard Queue**: Set to true to delete all online job queue entries (CI_BATCH_JOB table rows) created by Oracle Scheduler Stored Procedure OUAF_BATCH. The default is false. OUAF_BATCH uses individual CI_BATCH_JOB rows for the threads so it may become quite voluminous, and the job queue entries are typically not useful once a thread has started executing as the batch run tree (BRT) entries are used to track the job. This option can be used to clean up the unused entries.
- **Debug**: Default value is false. Can be set to true to see internally generated DBMS_OUTPUT debug messages.

Batch Job Stream Operations

The **Batch Job Stream Operations** portal provides an overview of batch job streams that are currently running or completed (i.e. in **Failed** or **Succeeded** or **Stopped** status) within the last week.

To navigate to Batch Job Streams Operations portal, from the **Admin** menu, select **Batch Operations**, and select **Batch Job Stream Operations**.

By default the batch job streams ran and completed in the last one week are displayed. Broadcast any of the stream run for more information on status of each of its steps.

Monitoring a Running Batch Job Stream

You can monitor the progress of a running Batch Job Stream by broadcasting it from the running query option search results. The broadcasted zone shows progress of the stream such as what steps have already completed and their status, what is the currently running step(s) along with steps that have not yet started. You can monitor the progress of the stream by refreshing the page or by configuring the page to refresh automatically. To configure automatic page refresh, refer to the **User Portal Preferences** for this portal.

Handling Exceptions

Batch Job Stream stuck in running status

A Batch Job Stream can be stuck in running status as one of its steps is in running status for too long. For example the corresponding batch job submission is stuck in pending status as the thread pool name set on the GLOBAL program options via a previous stream step is invalid. In such case, the running Batch Job Stream can be cancelled via the multi-select **Cancel** option. The thread pool name can then be set to a valid value and the Batch Job Stream can be submitted manually via the **Run Manually** option on the Batch Job Stream Definition or wait until it's scheduled to run next.

Errors in Batch Job Stream

A batch job submitted as part of a Batch Job Stream can run into error with one of its threads being in error. It could be because of business scenario or environment issue etc. In such a case, the corresponding Batch Job Stream stops execution and its step ends in failed state. The cause for the failure is noted on the Additional Details column on the stream step. You can fix the issue and restart the stream via the multi-select **Restart Stream** option on the batch job stream operations completed query option. Restarting a stream will result in the Batch Job Stream continuing execution from the step that failed and finish through to end.

NOTE:
Once a Batch Job Stream fails, its future scheduled runs no longer execute until the failed run is addressed.

Batch Queues Overview

Batch Queues is a new portal where users can view the status and progress of various batch jobs in the system. The portal provides visibility for users into what batch jobs are queued or running on existing thread pools DEFAULT and NOCACHE.

Users can view each of the batch job's details and broadcast them to view list of threads, their status and records details.

The list of batch jobs can be filtered by status, thread pool and batch control to view desired batch jobs.

Batch Alerts

This chapter provides information and requirements on batch alerts in support of Oracle Utilities Cloud implementations.

Batch Alerts Overview

Batch Alerts provide alerting and monitoring mechanisms for batch and batch job streams. Alerts are raised when desired batch jobs or batch job streams do not perform as expected.

Currently there are few level of service algorithms on batch controls to indicate how the corresponding batch jobs are doing.

New level of service algorithms are now introduced on the batch controls.

In addition, level of service algorithms are introduced on batch job streams to determine overall health of batch job streams.

New installation options health check algorithms are introduced for health component batch and batch job stream. Each health component's health check algorithm will invoke their level of service algorithms and return the overall results

Overall results from health check algorithms are displayed on the health check portal for monitoring purposes to view overall health of the system.

To receive alerts on overall health of the system, use the alert probe — bash script external utility that probes the system and sends email notifications to configurable set of email addresses.

The alert probe can be run on windows or linux machines. Alerts can be customized for errors and warnings. A paging alert is sent in case of errors. A non-paging alert is sent in case of warnings.

The frequency at which alerts are raised is configurable.

NOTE: The alert probe is provided on an use-as-is basis, no support is provided for it.

Alerting On A Batch Job Stream

To receive alerts on an active batch job stream, edit the batch job stream and add the desired level of service algorithm on it.

NOTE: Only algorithm types are delivered for Batch job stream level of service algorithms. A corresponding algorithm must be created before it can be added onto the batch job stream

Following level of services are currently supported on batch job streams

- Batch Job Stream has not started within 'X' minutes
- Batch Job Stream has failed
- Batch Job Stream ran too long (relative run)

Alerting On A Batch Job

Additional level of services are included on batch job that will be part of alerting mechanism

- Batch Job has processed low number of records
- Batch Job throughput is too low
- Batch Job ran too long (relative run)
- Batch Job has not started within 'X' minutes

Identity Cloud Service Integration

This chapter provides information about integration with Oracle Identity Cloud Service for Oracle Utilities cloud implementations.

NOTE: This section does not apply to all implementations. Contact your system administrator to confirm if your implementation includes Oracle Utilities Cloud Service Foundation and Identity Cloud Service integration.

Identity Cloud Service Integration Overview

Oracle Identity Cloud Service (IDCS) provides identity and access management functionality for the Oracle Utilities cloud services and supports single sign on (SSO) and identity federation capabilities.

IDCS instance is provided with each service subscription. This instance is also sometimes called IDCS tenancy or IDCS stripe. Once provisioned, it is administered exclusively by the client.

Security administrator uses IDCS to manage application users, who are given access to a one or multiple application environment(s). The security administrator may also assign another user to an administrative role in IDCS and delegate user management privileges.

Newly created users receive an account activation email from IDCS, and must reset their password.

The integration with IDCS supports Just-In-Time User Provisioning.

When a user attempts to access the utility application URL, the user is redirected to IDCS for authentication. Once the user is successfully authenticated by IDCS, they are redirected to the application and the authentication token is evaluated and validated. The system then checks if the user already exists in the application, and if not, it triggers the Just-In-Time User Provisioning logic that creates and activates the new user.

The **F1-IDMUser** business object is used for creation of new user records sourced from the external identity management integrations.

For the integration with IDCS, the algorithm is defined on the **F1-IDMUser** business object to determine the Template User whose information should be copied to the newly created user.

Just In Time User Provisioning

Just-In-Time User Provisioning is a process of creating a new user record upon first successful login. The implementation may follow either one of the main approaches:

- **Just-In-Time with Coarse-Grained User Authorization** allows security administrator to immediately begin working on the authorization and access setup, while other (“business”) users might wait until the proper access and authorization is configured for them in the application.
- **Just-In-Time with Fine-Grained User Authorization** provides both security administrator and the business user with instant access to the appropriate application functionality.

Coarse-Grained User Authorization (Default)

With course-grained user authorization, Just-In-Time Provisioning creates users with two authorization levels:

- **Security and Access Administrator.**
  - **Users with an administrative role in IDCS** are provisioned using a**Security Administrator Template User**. The user gains the access to all transactions and services related to application security and user authorization setup

- **Authenticated User**
  - **Users with no administrative role(s) in IDCS** are provisioned using a **Minimum Access Template User**. The user gains access to *My Preferences* page only. The security and access administrator completes the authorization and access setup manually.

No additional configuration effort is needed if your implementation chose to follow this approach. The scope of the actual security access for both Security Administrator and Authenticated User is customizable. Refer to Override Default Access for more information.

Fine-Grained User Authorization

With fine-grained user authorization,Just-In-Time Provisioning creates users with multiple authorization levels, according to IDCS Group membership and based on utility application configuration:

- **Security and Access Administrator**
  - **Users with an administrative role in IDCS** are provisioned using a **Security Administrator Template User**. The user gains the access to all transactions and services related to application security and user authorization setup
- **Business/Administrative Users** with access to a specific business and/or administrative functions in the application
  - **Users that are members of a Group in IDCS** are provisioned using a **Template User mapped to the IDCS Group**. The user gains access tothe application according to the Template User's setup.
- **Authenticated User**
  - **Users with no administrative role(s) in IDCS** and **no membership in IDCS Group** are provisioned using the **Minimum Access Template User**. The user gains access to *My Preferences* page only. The security and access administrator completes the authorization and access setup manually.

The mapping between IDCS Groups and Template Users is stored in theIDCS Integration Configuration.

Configuring Identity Cloud Service Integration

The integration configuration in the utility application has to be done in conjunction with the corresponding setup in IDCS.

Configuring Just In Time User Provisioning

In order to configure user authorization for Just in Time Provisioning, perform the following steps:

- Create User Groups that represent most typical business and/or administrative roles in the application
- Create Template Users that represent typical access level for major business and/or administrative functions
- Request your customer's IDCS Security Administrator to create Groups in IDCS that will correspond to the Template Users create above
- Setup the mapping between IDCS Groups and Template Users using IDCS Integration Configuration

Template Users and User Groups

- Define User Group(s) that represent the major business and/or administrative responsibilities in the application
- Define Template User(s) that represent the typical level of access to the application functionality

Identity Cloud Service Integration Configuration

The IDCS Integration Configuration master configuration stores the definitions related to the integration with IDCS.

For Just-In-Time User Provisioning, configure the **User Provision Rules**:

- Map the Groups defined in IDCS to the Template Users defined in the application.

Customizing the Integration

The product is delivered with out-of-the-box support for Just in Time User Provisioning with Coarse-Grained Authorization. Possible customization scopes are:

- Customize the default access level for Coarse-Grained Authorization. See Override Default Access Level for details.
- Configure the Just in Time User Provisioning with Fine-Grained Authorization.

Override Default Access Level

Both Security Administrator and Minimum Access Template Users are specified as parameters for the K1-IDCSUSRTM algorithm .

To override the default access level:

- Create new Template Users and setup user groups for default administrative access
- Create new Template Users and setup user groups for default minimum access

Create a new version of the K1-IDCSUSRTM algorithm and specify the Template Users created above as the algorithm's parameters

Override Logic to Determine Template User

The algorithm that determines the Template User is implemented as a pre-processing algorithm that determines the Template User on the **F1-IDMUser.** business object.

To override the logic used to determine the template user:

- Create a custom algorithm type and algorithm and use custom logic to determine the Template User for provisioning.
- Deactivate the existing product algorithm and specify the custom algorithm instead. Note that the algorithm that determines the Template User should be plugged before the algorithm that populates the data from the Template User.

Oracle Cloud Object Storage

Oracle Cloud Object Storage (Object Storage) is a part of Oracle Cloud Infrastructure Storage Services.

Oracle Cloud Object Storage can be accessed and managed using Oracle Cloud Infrastructure Console or programmatically using Command Line Interface (CLI), REST APIs or Java. The Object Storage that is used by the system is owned and managed by customers.

Oracle Utilities Cloud Services use Object Storage as the file storage location for all file related integrations. Batch processes that need to read or write files as input will access Object Storage.

The system requires some configuration in order to allow it to connect to the Object Storage and have access to the files that need to be processed by it. The following chapter describes the required configuration in order to connect to Object Storage as well as provides information on how to reference files in Object Storage in batch processes.

Please refer to Oracle Cloud Infrastructure documentation for more information on Object Storage.

File Storage Configuration for Object Storage

In order for the system to connect to Object Storage a File Storage Extendable Lookup has to be created with the following information:

- The Oracle Cloud Infrastructure tenancy details – which can be obtained from the Oracle Cloud Infrastructure Console(available to customers who purchased Object Storage).
- The API Key that is used by the system to connect to that Object Storage.

Refer to product documentation on “File Storage” for more details.

NOTE:

- Each File Storage Location configuration refers to a combination of Object Storage Tenancy, Compartment and User ID.Since customers can choose to create many Compartments in their Object Storage, access to each such Compartment will require a separate File Storage configuration.
- It is also recommended that each File Location configuration will reference a **different** Cloud Infrastructure User ID.This can limit, for example, the system’s access to production files from a non-production environment.
- API Keys are managed using Key Rings. Key Rings can be shared between File Storage Extendable Lookups but the Keys in these Key Rings cannot be shared across system environments. For example, keys generated for the Development environment **cannot** be used for the Testing or Production environments.Refer to the following topics in the Framework Administrative User Guide:

  External File Storage, **Key Rings**.

Referencing Files on Object Storage

Access to files is typically required in Batch processes. Some batch processes include direct reference to the files they need to process (for input or output) while some processes rely on additional configuration (for example, the Configuration Migration Assistant (CMA) Import and Export processes that rely on the CMA Master Configuration Settings).

Direct Reference

Reference to Object Storage can be used anywhere that a file location is allowed. The format is:

file-storage://<File Location>/<Bucket>/<Filename.ext>

- <File-Location>: The File Storage Extendable Lookup value that is defined for that file.It should include all the details the system needs to access the Object Storage location for that file.
- <Bucket>: The equivalent of a Folder in the Oracle Cloud Object Storage.Since the folder structure in Object Storage is flat, Buckets cannot include other Buckets and therefore only one Bucket can be noted here.

CMA Master Configuration

CMA Master Configuration includes the location of files for export and import processing:

- Import Folder: should reference a file location for an Object Storage Bucket(i.e. just the file location without the a file name, for example: “file-storage://CMALocation/CMA-Files”), “CMALocation” is the File Location Extendable Lookup name.
- Export Folder: should reference a file location for an Object Storage Bucket (e.g “file-storage://CMALocation/CMA-Files”).

NOTE: In order to take advantage of the automation of migration of configuration data from one system environment to another via the Process Automation Tool, the Import and Export directories have to point to the same location!

Conversion Data Upload Processes

The batch process that loads customer data files into the cloud (K1-CNVLD) using SQL Loader has two parameters that describes where to get the input data files and where to store the output files:

- inputFileStorage: should reference a file location for an Object Storage Bucket (e.g. “file-storage://CONVLocation/Input”),“CONVLocation” is the File Location Extendable Lookup name.
- outputFileStorage: should reference a file location for an Object Storage Bucket (e.g. “file-storage://CONVLocation/Output”).