OFS Compliance Studio

Administration and Configuration Guide

Release 8.1.2.0.0

January 2023

F48792-01

ORACLE

Financial Services

OFS Compliance Studio Administration and Configuration Guide

Copyright © 2023 Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are “commercial computer software” pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

For information on third party licenses, click here.

Document Control

Table 1 lists the document control of this guide:

Table 1: Document Control

Version Number	Revision Date	Change Log
12	January 2023	Added FCC_M_ER_PROCESSING_COLUMNS and FCC_DS_REF_COLUMN_MAPPING tables details in the Default Data in the tables section. Added a new Disable the User in Compliance Studio after SSO Login section. Added description for column details of the FCC_ER_MAPPING in the Output Tables section.
11	November 2022	• Updated a note related to clean-up action in the Clean-up Steps When the Create Index and Load Data Job Terminated Manually section. • Added a note related to unique constraint error in the STG_CUSTOMER_IDENTIFCTN_DOC table in the Data Survival section.
10	October 2022	Added new steps 1 and 6 in the ER Schema Changes section.
09	September 2022	Added a note in the Steps (third step of data survival).
08	August 2022	• Added additional steps in the Clean-up Steps When the Bulk Similarity Job Terminated Manually and Clean-up Steps When the Data Survival Job Terminated Manually sections. • Removed the Configure Quantifind section. • Updated steps 2 and 3 in the Elastic Search Changes section. • Updated a note related to STG_PARTY_ADDRESS_PRE in the Insert Data into Pre-Staging Tables section. • Updated script download path in the Data Slicing Utility Script section.
07	July 2022	• Updated minor changes in the Output Tables section.

Table 1: Document Control

Version Number	Revision Date	Change Log
06	June 2022	- Added new **Cleanup Steps for Job Termination** sections in all ER jobs. - Add new Clean-up Steps When the Create Index and Load Data Job Terminated Manually section. - Add new Clean-up Steps When the Bulk Similarity Job Terminated Manually section. - Add new Clean-up Steps When the Data Survival Job Terminated Manually section. - Add new Clean-up Steps When the Load Data in FCC_ER_OUTPUT Job Terminated Manually section. - Updated the script in ER Schema Changes section. - Removed the NOTE related to IDNTYADMN in Groups in Identity Management section.
05	May 2022	As part of the v8.1.2.0.1 version, the following sections are added/updated: - New job is added in the Status Codes section. - New table is added in the Output Tables section. - Added Additional Configurations subsection in Create Index and Load the Data section - Added validation step in Data Survival section - Added new Job Load Data in FCC_ER_OUTPUT Table - Added Initial Run for High Volume Data section. - Added Clean-up Steps When the Create Index and Load Data Job Terminated Manually subsection in **Appendix**. - Added Utility Scripts section in **Appendix.** - Renamed the subsections and section, Resetting Entity Resolution Back to Day 0 in **Appendix**.
04	April 2022	Updated the following sections: - Removed Kubernetes-related information from Add or Modify Python Packages to the fcc-python Interpreter section. - Added a Note in Configure the SSH Connection section. - Removed the “Modify the Python Docker Images for the Python Interpreter” section. Updated the content for deprecated ore Interpreter section.

Table 1: Document Control

Version Number	Revision Date	Change Log
03	March 2022	Updated the following sections: • User Access and Permissioning Management • PGX Interpreter • Entity Resolution • ML Name and Address Incremental Training API • Data Memory Limit Added the following sections: • Changing Default Features and Custom Model Training • PGX Permissions • Roles, Functions and Permissions • Clean-up Steps When the Bulk Similarity Job Terminated Manually
02	November 2021	Updated the document for the v8.1.1.1.0 release.
01	October 2021	This is the first version created for the v8.1.1.0.0 release.

Table of Contents

1 Preface

This guide provides information related to the Oracle Financial Services (OFS) Compliance Studio application administrator.

1.1 Audience

This guide is intended for Administrators, and the basic knowledge of the following is recommended:

- UNIX commands
- Database concepts
- Big Data
- Python
- Scala
- Spark
- Oracle R
- SQL
- PGX
- PGQL
- Markdown

1.2 Related Documents

We strive to keep this and all other related documents updated regularly; visit the OHC Documentation Library to download the latest version available there. The list of related documents is provided here.

- Oracle Financial Services Compliance Studio Installation Guide
- Oracle Financial Services Compliance Studio User Guide
- Oracle Financial Services Compliance Studio Matching Guide
- Oracle Financial Services Compliance Studio Data Model Guide
- Oracle Financial Services Compliance Studio Release Notes and Readme

1.3 Conventions

Table 2 explains the text conventions used in this guide.

Table 2: Convention

Convention	Description
Italics	• Names of books, chapters, and sections as references • Emphasis
Bold	• The object of an action (menu names, field names, options, button names) in step-by-step procedures • Commands typed at a prompt • User input

Table 2: Convention

`Monospace`	● Directories and subdirectories ● File names and extensions ● Process names ● Code sample, including keywords and variables within a text and as separate paragraphs, and user-defined program elements within a text
Hyperlink	Hyperlink type indicates the links to external websites and internal document links to sections.
<Variable>	Substitute input value

1.4 Abbreviations

Table 3 lists the abbreviations used in this document.

Table 3: Abbreviations

Abbreviation	Meaning
OFS	Oracle Financial Services
OFSAA	Oracle Financial Services Analytical Application
BD	Behavior Detection
FCDM	Financial Crime Data Model
ICIJ	International Consortium of Investigative Journalists
MMG	Model Management and Governance
SSO	Single Sign-On
SSH	Secure Shell

2 About Compliance Studio Administration

OFS Compliance Studio is an advanced analytics application that supercharges anti-financial crime programs for better customer due diligence, transaction monitoring, and investigations by leveraging the latest innovations in artificial intelligence, open-source technologies, and data management. It combines Oracle's Parallel Graph Analytics (PGX), Machine Learning for AML, Entity Resolution, and notebook-based code development and enables Contextual Investigations in one platform with complete and robust model management and governance functionality.

Topics:

- Capabilities offered by Compliance Studio
- Configurable Features
- Administration Overview

2.1 Capabilities offered by Compliance Studio

- Purpose Built for Fighting Crime
  - Fully defined and sourced Financial Crime Graph Model supporting detection and investigation
  - Provided Accelerators for finding the needles in the haystack.
  - What if Analysis for existing Scenarios
  - Integration with ECM and Investigation Hub to provide meaningful guidance to investigators for rules-based and ML-generated alerts
  - Enterprise-ready and compatible with the underlying OFSAA framework
  - Works with earlier 8.0.x releases of Oracle Financial Crime and Compliance Management Anti Money Laundering (AML), Enterprise Case Management, and Fraud applications.
- Entity Resolution for AML
  - Entity Resolution to enhance monitoring effectiveness and provide a single customer view
  - Linking and Resolution across internal & external data to improve single entity detection
  - Allows for Scenario/Model detection across internal data
  - Multi-attribute enabled with ML boosts for Name/Address models
  - Prebuilt Integrations and easily configurable for Data Sources like ICIJ, Safari, etc.
- Analytics of Choice
  - Choose from our proprietary models or bring your own
  - Fully embedded Graph Analytics Engine and Financial Crime Model
  - Supports non-ML
  - Embedded with a highly scalable in-memory Graph Analytics Engine (PGX)
  - Industry's most intuitive Graph Query Language to gain rapid insights
- Model Management & Governance
  - End-to-end management from model creation to model deployment.
    - — Data Ingestion (Oracle DB, Graph, Hive)

        - Model Development
        - Supports virtually all open source packages, interpreters, etc.
        - Supports non-ML
        - Process in Database or Big Data
        - Model Training
        - Model Performance Evaluation
        - Model Explainability
        - Model Tracking and Audit
        - Approval Mechanisms
        - Model Deployment
        - Scheduling
        - Ongoing Monitoring
    - API Based on model inference and training services
    - Citizen Canvas
    - Text2Graph Models
    - Graph2Text
    - Auto-Tagging
- ML Foundation for Financial Crimes
    - Integrated with Oracle Financial Crime Application Data and readily usable across the enterprise financial crime data lake.
    - Pre-engineered features and transformations to address each use case
    - Simplified APIs for each stage of the modeling lifecycle
    - Leverage the power of Graph, Supervised ML, and Unsupervised ML to build typology detection models, detect anomalies, and risk score customers or events
    - Event Scoring for false positive prediction and disposition
    - Dashboards and KPIs to measure the impact of ML and generate insights for business users
    - Ongoing Monitoring of Model Performance and Concept Drift

2.2 Configurable Features

The following are the key configurable features in Compliance Studio:

- Create users and roles to access Compliance Studio to access through AAI/SSO
- Assign roles and groups with required permissions
- The ability to customize and create interpreter variants to provide or restrict access to users
- Modify ready-to-use Python packages and versions
- Customize rulesets to generate similarity edges and resolved entities
- Apply Graph Fine-Grained Access Control to redact the sensitive data in the Graphs

- Move source data to the PGX server using connector jobs to create graphs in FCDM and ICIJ workflows
- Monitor tasks that the logged-in users perform
- Offers ready-to-use extract, transform, load (ETL) operations for the creation of a global graph
- Entity resolution based on configurable rules.

2.3 Administration Overview

This section provides an overview of administration activities performed by an Administrator after installing the Compliance Studio application.

The following are the key configuration activities performed by an Administrator in Compliance Studio:

- Users*:* To access the application, users must be authenticated. In Compliance Studio, users and roles are authenticated based on Realms, such as FCCRealm, SAMLRealm, etc. These Realms use Identity Management systems to authenticate users. FCCRealm - uses Oracle Financial Services Analytical Applications Infrastructure (OFSAAI), and SAMLRealm uses an identity provider (IDP).
- Preconfigured Groups, Roles, and Actions: After authentication of users and roles, they must be authorized to use the application. The Compliance Studio offers a rich permission system, and users are mapped to the permissions to use the application.
- Configure Interpreters: Interpreters are used to execute code in different languages. Plug-ins enable users to use a specific language to process data on the selected execution platform. The Compliance Studio provides ready-to-use interpreters, such as jdbc-interpreter, python interpreter, etc. In Compliance Studio, you can either use a default interpreter variant or create a new variant for an interpreter to provide access to the database for different users. Interpreters are linked using credentials (a wallet and a password) to enable secure data access. Interpreters are configured based on usage.
- Entity Resolution: OFS Compliance Studio provides Entity Resolution (ER) capability. It allows firms to break through barriers in their data by gaining single views of their customers and their external entities and have the choice of monitoring them both under one consolidated Global Party.
- OFS Compliance Studio Entity Resolution is a configurable process that allows data to be matched and merged to create contextual links in the global graph or resolve relational party records to a global party record as part of ingestion. OFS Compliance Studio has pre-built configurations supporting matching (or linking) in the FCGM and resolving entities in CSA for data being loaded into Financial Services Data Foundation (FSDF).
- Configure a Data Source: The data source configuration allows you to view the newly added edges or nodes in the graph. Define the source of the data, specify the order in which the files must be read, etc.
- Configure Graph: The Compliance Studio provides an intuitive way for creating graphs used in notebooks, where you can load graphs from external sources or create custom graphs. Using PGX, you can load multiple graphs into a notebook and create PGQL queries against different graphs. The result obtained from running a paragraph in a notebook can be used as an input for other paragraphs in the notebook. The results of analytics algorithms are stored as transient properties of nodes and edges in the graph. Pattern matching can then be used against these properties.
- Apply Graph Fine-Grained Access Control: The Graph Fine-Grained Access Control and Redaction changes are applied to the Compliance Studio to redact the sensitive data in the

Graphs. With a role-based access control approach, you can restrict access at any level of granularity.

- Prepare the Batches: Batches are prepared to execute the ETL operations. Batches enable you to move data from Oracle Database or Big Data to Compliance Studio, load graphs, and run notebooks.
- Perform the Batches: These batches contain Sqoop Job, Connector Job, Graph Job, and Similarity Edge Generation Job in OFSAI. You can also execute ETL operations by running the scripts without configuring the batches.
- Verify Batch Execution: Verify the status of all tasks at the end of the batch execution. You can verify both the overall status of the Batch and individual task status.
- Schedule Scenario Notebook Execution: You can schedule a notebook execution using the scheduler.

> **NOTE** In the current release, Notebook execution using Batch is deprecated and will be removed in the future release. It is recommended to use the scheduler to execute the notebook in Batch.
>
> You can see the Example of Creating a batch in Scheduler for Notebook Execution.

- Monitoring Tasks on Notebook Server: Tasks are created when the end-user executes notebooks or paragraphs. It is important to know the execution status of whether the tasks are created, rejected, canceled, etc. The Tasks page allows you to view the status of the task and associated notebooks, paragraphs, interpreters, etc. By default, all the tasks are listed on the Task page. You can view the specific task using filters such as the task's status, date of creation, and notebook name.

2.3.1 Key Concepts

This section provides insight into the following key concepts:

- **Interpreter**: An interpreter is a program that directly executes instructions written in a programming or scripting language without requiring them previously to be compiled into a machine language program. They are plug-ins that enable users to use a specific language to process data in the backend. Examples of Interpreters are jdbc-interpreter, spark-interpreters, python-interpreters, etc. Interpreters allow you to define customized drivers, URLs, passwords, connections, SQL results to display, etc.
- **Zeppelin Interpreter**: A plug-in enables Zeppelin users to use a specific language or data-processing-backend. For example, to use the Scala code in Zeppelin, you need a %spark interpreter.
- **Zeppelin**: Interactive browser-based notebooks enable data engineers, data analysts, and data scientists to be more productive by developing, organizing, executing, and sharing data code and visualizing results without referring to the command line or requiring the cluster details. Notebooks allow these users not only allow to execute but to interactively work with long workflows.
- **Markdown (md)**: A plain text formatting syntax designed so that it can be converted to HTML. Use this section to configure the markdown parse type.
- **Parallel Graph Analytics (PGX)**: Graph analysis lets you reveal latent information that is not directly apparent from fields in your data but is encoded as direct and indirect relationships - metadata - between elements of your data. This connectivity-related information is not obvious

to the naked eye but can have tremendous value when uncovered. PGX is a toolkit for graph analysis, supporting both efficient graph algorithms and fast SQL-like graph pattern matching queries.

- **PySpark**: A Python API is written in Python to support Spark. Spark is a distributed framework that can handle Big Data analysis. Spark is a computational engine that works with huge sets of data by processing them in parallel and batch systems.
- **Spark**: A fast and general-purpose cluster computing system. It provides high-level APIs in Java, Scala, Python, and R. Spark is an optimized engine that supports general execution graphs.
- **PGQL**: A graph query language built on top of SQL, bringing graph pattern matching capabilities to existing SQL users and new users interested in graph technology but who do not have an SQL background.
- **Data discovery**, **exploration**, **reporting**, and **visualization** are key components of the data science workflow. Zeppelin provides a "Modern Data Science Studio" that supports ready-to-use Spark and Hive. Zeppelin supports multiple language backends, which has support for a growing ecosystem of data sources. Zeppelin's notebooks provide interactive snippet-at-time experience to data scientists. You can see a collection of Zeppelin notebooks in the Hortonworks Gallery.
- **Keytab File**: A Keytab is a file containing pairs of Kerberos principles and encrypted keys (which are derived from the Kerberos password). You can use a keytab file to authenticate to various remote systems using Kerberos without entering a password. However, when changing your Kerberos password, you must recreate all your keytabs files. They are commonly used to allow scripts to automatically authenticate using Kerberos, without requiring human interaction or access to the password stored in a plain-text file. The script can use the acquired credentials to access files stored on a remote system.
- **Oracle Wallet**: Oracle Wallet is a file that stores database authentication and signing credentials. It allows users to securely access databases without providing credentials to third-party software, and easily connect to Oracle products.
- **Sqoop Job**: Sqoop is a tool designed for efficiently transferring bulk data between Hadoop and structured data stores such as relational databases. Sqoop job creates and saves the import and export commands. It specifies parameters to identify and recall the saved job. This re-calling or re-executing is used in the incremental import, which can import the updated rows from the RDBMS table to Hadoop Distributed File System (HDFS).
- **Elasticsearch:** Elastic search is a distributed search and analytics engine for all data types, including textual, numerical, geospatial, structured, and unstructured.

3 User Access and Permissioning Management

Compliance Studio uses a realm based on unique authentication and authorization for its users. Realm is a security policy domain defined for the application server. It is used to authenticate and authorize users of Compliance Studio.

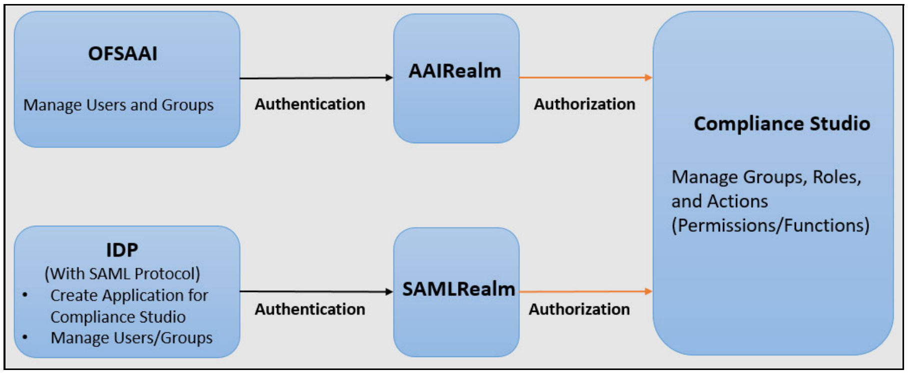
Realms (AAIRealm, SAMLRealm) are selected based on the Identity Provider (IDP) during the installation. For more information, see the OFS Compliance Studio Installation Guide.

The Compliance Studio application is accessed using the following realms that you have selected during the installation of the Compliance Studio application:

- **AAIRealm**: This uses Oracle Financial Services Analytical Applications Infrastructure (OFSAAI) Identity Management system for user authentication. The OFSAAI facilitates System Administrators to provide access, monitor, and administer users along with the Infrastructure metadata operations. For more information, see Access Compliance Studio Using AAI Realm section.
- **SAMLRealm**: The SAMLRealm uses an identity provider (IDP) Identity Management system to support the SAML2.0 protocol for user authentication. Security Assertion Markup Language (SAML) is an open standard that allows identity providers (IDP) to pass authorization credentials to service providers (SP). IDP acts as the Single Sign-On (SSO) service. Users and Groups are created in the IDP. For more information, see Access Compliance Studio Using SAML Realm section.

The following image illustrates the authentication and authorization process in Compliance Studio.

Figure 1: Compliance Studio - Authentication and Authorization process



Topics:

- Users
- Preconfigured Groups, Roles, and Actions
- Viewing Users and Viewing and Creating Groups, Roles, and Actions
- Access Compliance Studio Using AAI Realm
- Access Compliance Studio Using SAML Realm

3.1 Users

Authenticated and authorized users can access Compliance Studio. For example, Mark, a user, can access Compliance Studio after logging in where the user is authenticated and authorized.

3.1.1 System User - Group Mapping

Table 4 lists the System User - Group Mapping:

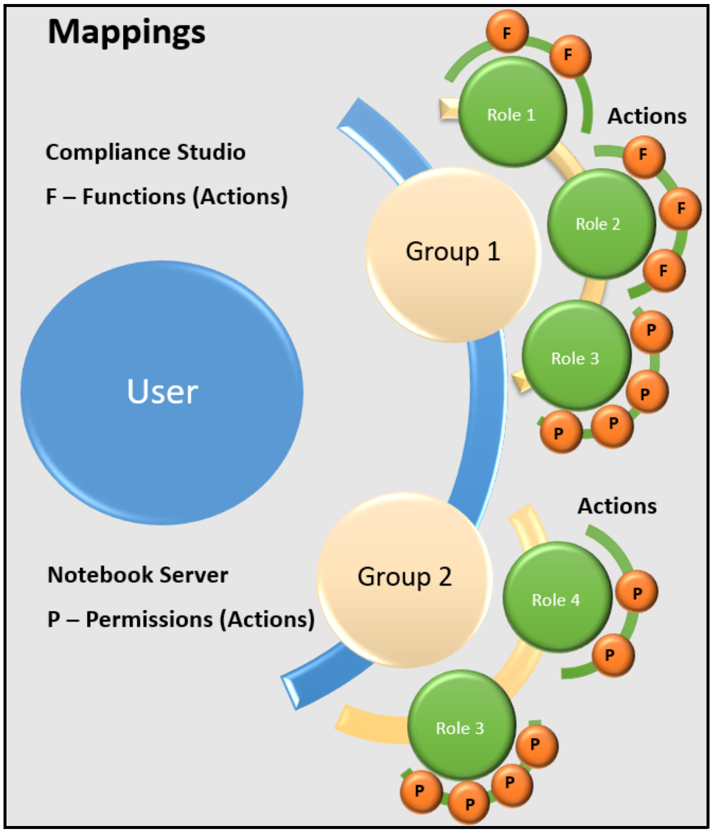
Table 4: System User - Group Mapping

User	Group Name
MMGUSER	• Workspace Administrator • Modeling User • Modeling Reviewer • Modeling Approver • Identity Administrator group • Identity Authorizer group • Datastudio User
SYSADMN	• Identity Administrator group • Datastudio User

3.2 Preconfigured Groups, Roles, and Actions

- User is mapped to single or multiple groups in AAI or a SAML-supported Identity Management System.
- A group of the same name will exist in Compliance Studio.
- In Compliance Studio, groups are mapped to single or multiple roles.
- Roles are mapped to single or multiple actions.

Figure 2: Mapping



> **NOTE** Compliance Studio has two User Interfaces. In Compliance Studio, actions are called Functions, and in Notebook Server, these are called Permissions. Both are the same.

3.2.1 Groups in Identity Provider and Compliance Studio

Users are mapped to groups in the Identity Provider, and Groups are mapped to Roles, Permissions, and Functions in Compliance Studio.

3.2.1.1 Groups in Identity Management

Groups need to be created in the Identity Provider, which matches the groups' names within Compliance Studio. Compliance Studio will not use any roles which are mapped in your Identity Provider but if you cannot create groups with roles, create a dummy role and map to the required groups if needed.

Users then need to be mapped to these groups.

> **NOTE** Group names might be similar to existing groups in AAI, but they are not the same. New groups need to be created in Identity Provider so the Users can be mapped to these groups.

Table 5 the Preconfigured Groups which exist in Compliance Studio and need to be created in the Identity Provider.

Table 5: Preconfigured Group

Group ID	Name	Description
IDNTYADMN	Identity Administrator group	Identity Administrator group
IDNTYAUTH	Identity Authorizer group	Identity Authorizer group
MDLUSR	Modeling User	Modeling User Group
MDLREV	Modeling Reviewer	Modeling Reviewer Group
MDLAPPR	Modeling Approver	Modeling Approver Group
SANDBOXADM	Sandbox Administrator	Sandbox Administrator Group
WKSPADMIN	Workspace Administrator	Workspace Administrator Group
MDLBATCHUSR	Modeling Batch User	The scheduler can use this Group for executing batches.
DSUSRGRP	Datastudio User	Datastudio User Group
DSREDACTGRP	DSREDACTGRP	This Group will be applicable to only those users for whom graph redaction is required.
DSADMIN	DSADMIN	-

3.2.2 Roles and Permissions

Groups in compliance Studio are mapped to roles, and roles map to permissions. For details on preconfigured roles and permissions, see the Role - Permission Mapping section.

3.3 Viewing Users and Viewing and Creating Groups, Roles, and Actions

3.3.1 Viewing Users in Compliance Studio

Viewing Users is a part of the identity manager UI. Note that Users cannot be created here; instead, the UI enables to view which users are currently active and logged in to the UI.

3.3.2 Viewing and Creating Groups

The Groups section allows users to view and manage all Groups and Permissions/Functions for it. Groups can be added, updated, and deleted. However, it is recommended to use Preconfigured Groups in Compliance Studio.

To add/update/delete the Groups, perform the following:

1. Login to the Compliance Studio application.

   In the Header, the following user-interface elements are displayed for OFS Compliance Studio:

   User Name: Displays the logged-in user name with the following options in the drop-down list:

   - Admin
   - Logout

2. Click **Admin**. The Identity Management window is displayed. For more information on adding, updating, and deleting Groups, see the OFS Admin Console User Guide.

> **NOTE** Groups are managed only through Compliance Studio.

3.3.3 Viewing and Creating Roles

The Roles section allows users to view and manage all Roles and Permissions/Functions for it. The Roles can be added, updated, and deleted. However, it is recommended to use Preconfigured Roles in the application.

> **NOTE** Compliance Studio has two User Interfaces, Compliance Studio and Notebook Server. Some roles are specific to Compliance Studio, and some are specific to Notebook Server.
>
> To access the Compliance Studio screens from the custom Role that is created in the Notebook Server. You must create a role in the Compliance Studio through Identity Management first and then create the same Role in the Notebook Server. See the following sections for more information:
>
> - Roles in Compliance Studio
> - Roles in Notebook Server

3.3.3.1 Roles in Compliance Studio

1. Log in to the Compliance Studio application.
2. Navigate to the Identity Management. For more information on adding, updating, and deleting Groups, see the OFS Admin Console User Guide.

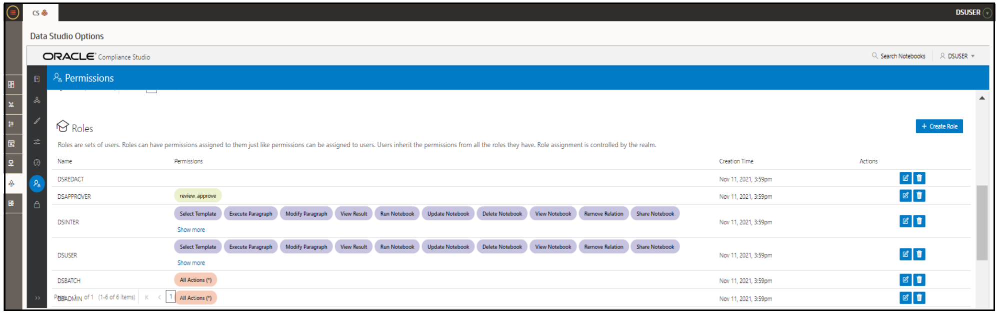
3.3.3.2 Roles in Notebook Server

> **NOTE** Ensure that you have already created a role in the Compliance Studio before creating the same Role in the Notebook Server.

To create a Role in Notebook Server, perform the following:

1. Login to the Compliance Studio application. The **Workspace Summary page** is displayed.
2. Launch CS Production Workspace using Launch Workspace. The CS Production window is displayed.
3. Navigate to **Data Studio Options**, and then click **Permissions**. The Notebook Server window is displayed.

Figure 3: Permissions



Roles

Roles are sets of users. Roles can have permissions assigned to them just like permissions can be assigned to users. Users inherit the permissions from all the roles they have. Role assignment is controlled by the realm.

Name	Permissions	Creation Time	Actions
DSREDACT		Nov 11, 2021, 3:59pm	
DSAPPROVER	review_approve	Nov 11, 2021, 3:59pm	
DSINTER	Select Template, Execute Paragraph, Modify Paragraph, View Result, Run Notebook, Update Notebook, Delete Notebook, View Notebook, Remove Relation, Share Notebook Show more	Nov 11, 2021, 3:59pm	
DSUSER	Select Template, Execute Paragraph, Modify Paragraph, View Result, Run Notebook, Update Notebook, Delete Notebook, View Notebook, Remove Relation, Share Notebook Show more	Nov 11, 2021, 3:59pm	
DSBATCH	All Actions (*)	Nov 11, 2021, 3:59pm	
DSADMIN	All Actions (*)	Nov 11, 2021, 3:59pm	

Page 1 of 1 (1-6 of 6 items)

4. In the Roles pane. Click **Create Role**. The Create window for Roles is displayed.

5. Enter the name of the Role.

> **NOTE** Enter up to 30 characters.

6. Select the required Permissions, Notebooks, Interpreter Variants, General actions, Interpreters, and Graphs. The selected permissions are underlined.

7. To select all the options, click **All** All Actions (*). All the options in the Create window are selected.

   To deselect all the options, click **All** All Actions (*) again.

8. Click **Save**. A confirmation message is displayed.

To modify Roles, click **Modify**. The Update Roles window is displayed. Go to required options, such as Permission, Notebook, Interpreter Variant, General, Interpreter, Graph, etc.

To delete the Roles, click **Delete** for the corresponding Role.

3.3.4 Mapping Functions/Permissions to Roles

Permissions are a set of actions. Permissions can be added, updated, and deleted. However, it is recommended to use Preconfigured Permissions in the application.

3.3.4.1 Role - Functions Mapping

1. Log in to the Compliance Studio application.

2. Navigate to the Identity Management. For more information on adding, updating, and deleting functions, see the OFS Admin Console User Guide.

3.3.4.2 Role - Permissions Mapping

The process is the same as Creating Roles. For more information on adding, updating, and deleting Permissions, see the Roles in Notebook Server section.

3.4 Access Compliance Studio Using AAI Realm

This section provides information on creating users who can access Compliance Studio using the AAIRealm method of authentication through Oracle Financial Services Analytical Applications Infrastructure (OFSAAI). The users with SYSADMN and SYSAUTH roles in OFSAAI can create and authorize users, respectively.

Identity Management in the OFSAAI facilitates System Administrators to provide access, monitor, and administer users along with the Infrastructure metadata operations. The Security Management System (SMS) component is incorporated with Password Encryption, Role and Data-Based Security, Access Control, and Audit Trail feature to provide a highly flexible security envelope. Administrators can create, map, and authorize users defining a security framework that can restrict access to the data and metadata in the warehouse, based on a fine-grained access control mechanism. These activities are done at the initial stage and then on a required basis.

To create and authorize the users in OFSAAI, perform the following steps:

1. Login to AAI using SYSADMIN.
2. Create the following Groups:
   - Create the new groups with the same name as the pre-configured groups. See the Preconfigured Groups, Roles, and Actions section for more information.
   - Create the new groups in AAI and Compliance Studio for custom groups. See Viewing Users and Viewing and Creating Groups, Roles, and Actions section.

NOTE	<ul><li>Roles assigned to a Group in AAI are not considered in Compliance Studio.</li><li>Compliance Studio manages Roles; Group-Role is mapped in the Compliance Studio. For more information, see the Group - Role Mapping section.</li><li>To create a valid group in AAI, you need to map it to Domain and map at least one role. Note that for Compliance Studio, neither domain nor roles mapped to it in AAI are relevant. So, you can assign a dummy domain and role for group creation in AAI.</li></ul>

3. Create a user and map the user groups to the respective user.

The default permissions mapped to these users and user groups are available in the Roles, Functions and Permissions section.

3.5 Access Compliance Studio Using SAML Realm

This section provides information on managing users who can access Compliance Studio with Identity Provider (IdP or IDP). The IdP acts as the Single Sign-On (SSO) service provider for implementations between Compliance Studio, Investigation Hub, and Enterprise Case Management. This configuration prevents separate login for each application.

An identity provider (IdP) is a service that stores and verifies user identity. IdPs work with single sign-on (SSO) providers to authenticate users. An identity provider (IdP or IDP) stores and manages users' digital identities. An IdP checks user identities via username-password combinations and other factors, or it may simply provide a list of user identities that another service provider (like an SSO) checks.

See the Preconfigured Groups, Roles, and Actions section for Preconfigured Groups to access Compliance Studio using SAMLRealm.

To integrate Compliance Studio with IDP as the SSO provider, follow these steps:

1. Create the following Groups in the IDP system. For more information on creating groups in IDP, see the OFS Admin Console User Guide.
   - Create the new groups with the same name as the pre-configured groups. See the Preconfigured Groups, Roles, and Actions section for more information.

- Create the new groups in IDP and Compliance Studio for custom groups. See Viewing Users and Viewing and Creating Groups, Roles, and Actions section.

> **NOTE** You can either use Groups from Preconfigured Groups or create custom Groups. For more information, see the Preconfigured Groups, Roles, and Actions or Viewing Users and Viewing and Creating Groups, Roles, and Actions section, respectively.

2. Create two SAML applications in IDP.
3. Configure the SAML applications.

   Key configurations in the SMAL applications are as follows:

   a. First Application:

   — Entity ID (SAML_ISSUER): `https://<FQDN of Compliance Studio Linux server>:7008`

   FQDN: Fully Qualified Host Name. Example: myserver.com

   — Assertion Consumer URL (SAML_ASSERTION): `https:// <FQDN of Compliance Studio Linux Server>:7008/saml/consume`

   Example: `https://<My Oracle Server>:7008/saml/consume`

   — Signed SSO: Response

> **NOTE** **Assertion** and **Response** in SAML response must be signed.

   — Include Signing Certificate in Signature: **Enabled**

   — Include Signing Certificate in Signature: **SHA-256**

   — Encrypt Assertion: **Disabled**

   — SAML Attribute Configuration:

   ```
   Name: <SAML_ROLE_ATTRIBUTE from config.sh>
   Format: Basic
   Value: <List of all groups>
   ```

   b. Second Application:

   — Entity ID: `https://<FQDN of Compliance studio Linux Server>:7001/cs`

   — Assertion Consumer URL: `http:// <FQDN of Compliance studio Linux Server>:7001/cs/home`

> **NOTE** **Response** in SAML response must be signed.

   — Include Signing Certificate in Signature: **Enabled**

   — Include Signing Certificate in Signature: **SHA-256**

   — Enable Single Logout: **Enabled**

   — Logout Binding: **POST**

- Single Logout URL (SAML_LOGOUT_URL): `http://<FQDN of compliance studio>:7001/cs/signoff`
- Logout Response URL: `http://<FQDN of compliance studio>:7001/cs/signoff`
- Encrypt Assertion: **Disabled**
- SAML Attribute Configuration

  ```
  Name: ofs_mapped_groups
  Format: Basic
  Value: <List of all groups>
  ```

4. Create a user and map the user groups to the respective user based on the user roles.

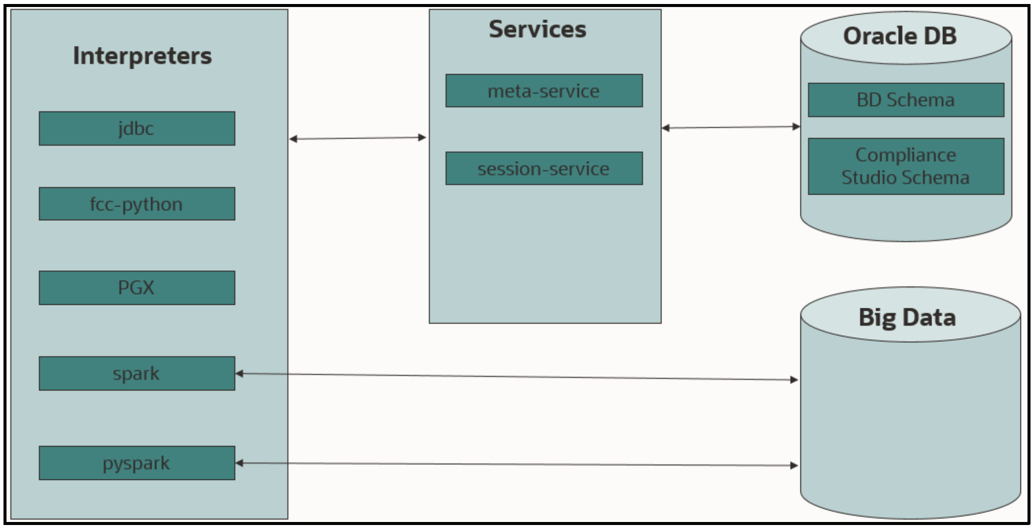
4 Interpreter Configuration and Connectivity

An interpreter is a program that directly executes instructions written in a programming or scripting language without requiring them previously to be compiled into a machine language program. Interpreters are plug-ins that enable users to use a specific language to process data in the backend. Examples of Interpreters are jdbc-interpreter, spark-interpreters, python-interpreters, etc. Interpreters allow you to define customized drivers, URLs, passwords, connections, SQL results to display, etc.

In Compliance Studio, Interpreters are used in Notebooks to execute code in different languages. Each Interpreter has a set of adjusted and applied properties across all notebooks. For example, using the python-interpreter makes it possible to change between versions, whereas the jdbc-interpreter offers to customize the URL, schema, or credentials. In Compliance Studio, you can either use a default interpreter variant or create a new variant for an interpreter. You can create more than one variant for an interpreter. The benefit of creating multiple variants for an Interpreter is to connect different versions of interpreters (Python version: 3, Python version: 2, etc.). This helps to connect a different set of users and database schema. For example, Compliance Studio schema, BD schema, etc. Compliance Studio provides secure and safe credential management such as Oracle Wallet (jdbc wallet), Password (jdbc password), or KeyStores to link to interpreter variants to access secured data.

The following image illustrates the examples of interpreters used in Compliance Studio and database connections.

Figure 4: Examples of Interpreters



Topics:

- Configure Interpreters
- Link Credentials
- Create a Credential
- Create an Interpreter Group
- Create an Interpreter Variant
- Enable Additional Spark or PySpark interpreter

4.1 Configure Interpreters

Compliance Studio has ready-to-use interpreters such as fcc-python, jdbc Interpreter, etc. You can configure them based on the use case. Additional variants of interpreters are created as multiple users might require different settings to access the database securely. The jdbc Interpreters use the credentials to enable secure data access.

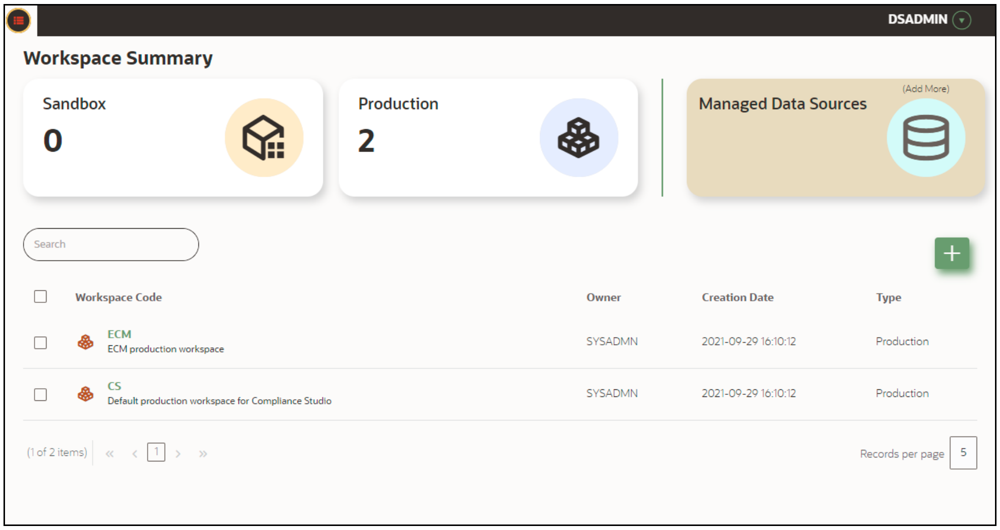
> **NOTE** fcc-python, pyspark, spark, and python are some of the other available interpreters.

Interpreters are configured when you want to modify URL, data location, drivers, enable or disable connections, etc.

To configure ready-to-use interpreters, follow these steps:

1. On the **Workspace Summary** page, select Launch workspace to display the **CS Production workspace** window**.**

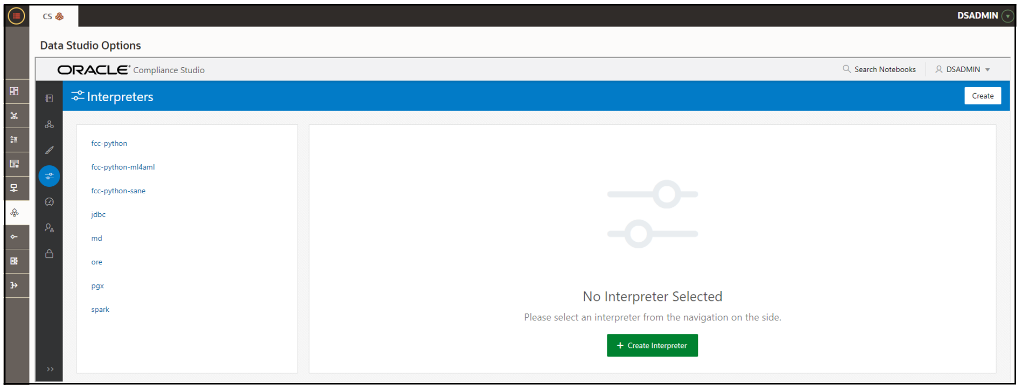
Figure 5: Workspace Summary screen



□	Workspace Code	Owner	Creation Date	Type
□	ECM ECM production workspace	SYSADMN	2021-09-29 16:10:12	Production
□	CS Default production workspace for Compliance Studio	SYSADMN	2021-09-29 16:10:12	Production

2. Hover the mouse over the Ruleset Details widget the following options are available:
   - Interpreters
   - Tasks
   - Permissions
   - Credentials
   - Templates
3. Click **Interpreter** that you want to view from the list displayed on the LHS. The default configured interpreter variant is displayed on the RHS.

Figure 6: Interpreters' screen



No Interpreter Selected

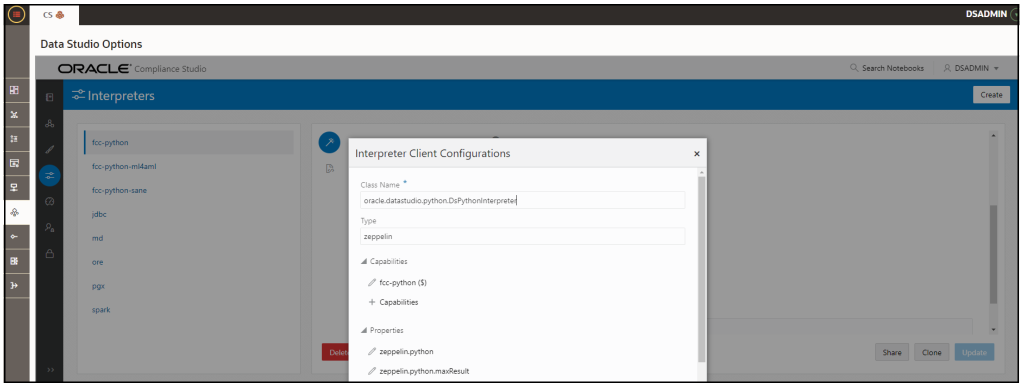
Please select an interpreter from the navigation on the side.

4. Modify the values in the fields as per requirement. For example, to modify a parameter's limit, connect to a different schema, PGX server, etc.

   You can modify the values in the following UI options:

   - Wizard

Figure 7: Wizard UI options



Interpreter Client Configurations

Class Name *: oracle.datastudio.python.DsPythonInterpreter

Type: zeppelin

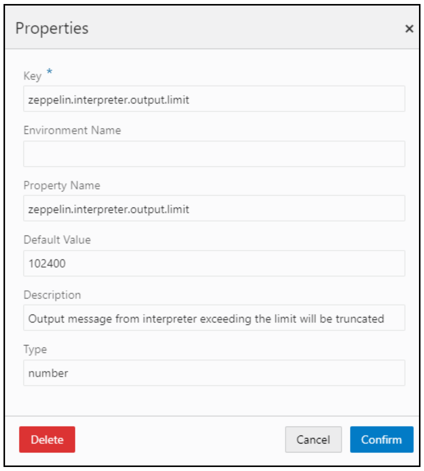
Capabilities
- fcc-python ($)
- + Capabilities

Properties
- zeppelin.python
- zeppelin.python.maxResult

An interpreter can group multiple interpreter clients that all run in one JVM process and can be stopped together.

For example, the spark interpreter group contains the spark and pyspark interpreter client.

Figure 8: Properties screen



Key *
zeppelin.interpreter.output.limit

Environment Name

Property Name
zeppelin.interpreter.output.limit

Default Value
102400

Description
Output message from interpreter exceeding the limit will be truncated

Type
number

Group Configuration

Initial Code

For example, when using a Spark interpreter group with spark and pyspark interpreter clients. If you define the initialization code for the spark interpreter group, the initialization code will run when the runtime environment is created, i.e., the first time a user runs a paragraph of either spark or pyspark in a notebook with Compliance Studio running in NOTEBOOK session mode.

Initial Code Capability

The initial code capability defines what interpreter client to use to run the group initial code. For example, in the spark interpreter group, we would select the spark capability as the initial code capability to create a spark context for the group JVM process.

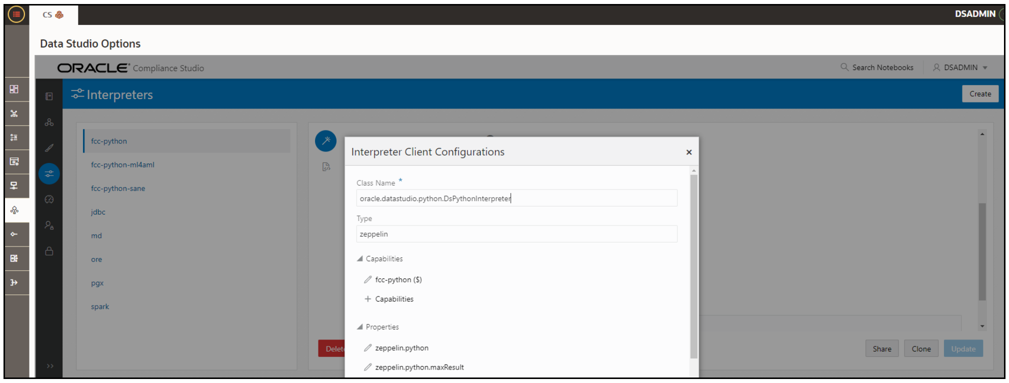
Credential Configurations

For linking any credentials to the interpreter, we have to define what credential types should be used and what credential mode to use. For example, the jdbc interpreter supports a credential type of type Password for the credential qualifier **jdbc_password** and a credential type of type Oracle Wallet for the credential qualifier **jdbc_wallet**. After defining the credential configuration, a new section for selecting the respective credential values will appear.

Interpreter Client Configuration

Interpreter properties can be configured for each interpreter client.

Figure 9: Interpreter Client Configuration



Interpreter Client Configurations

Class Name *
oracle.datastudio.python.DsPythonInterpreter

Type
zeppelin

Capabilities
- fcc-python ($)
- + Capabilities

Properties
- zeppelin.python
- zeppelin.python.maxResult

Lifecycle Configuration

Host Mode

In the Host lifecycle mode, the following properties can be configured:

- **Host**: The hostname on which the interpreter is listening. For example, localhost if the **interpreter** runs on the same machine as the server.
- Port: The port on which the interpreter is listening.

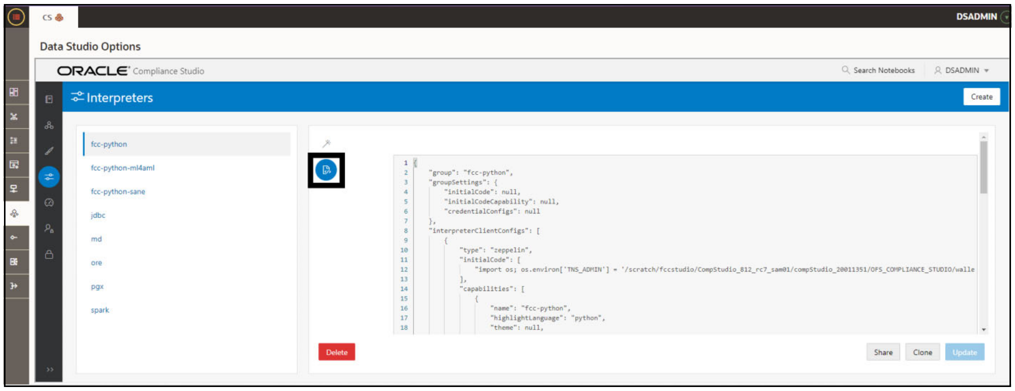
Credentials

A credential section appears if you have defined a credential configuration as part of the group settings. For each credential qualifier, an already defined credential can be selected. If the credential mode Per User is used, each individual user has to select their own credential.

- **JSON**:

You can modify the values in the properties of the interpreter in the JSON file, as shown in the following figure.

Figure 10: JSON file properties



```
{
    "group": "fcc-python",
    "groupSettings": {
        "initialCode": null,
        "initialCodeCapability": null,
        "credentialConfigs": null
    },
    "interpreterClientConfigs": [
        {
            "type": "zeppelin",
            "initialCode": [
                "import os; os.environ['TNS_ADMIN'] = '/scratch/fccstudio/CompStudio_812_rc7_sam01/compStudio_20011351/OFS_COMPLIANCE_STUDIO/walle
            ],
            "capabilities": [
                {
                    "name": "fcc-python",
                    "highlightLanguage": "python",
                    "theme": null,
```

5. Click **Update**. The modified values are updated in the Interpreter.
6. The user can also perform **Share**, **Clone**, and **Delete** operations on this screen.

Table 6 lists the Ready-to-use interpreter in Compliance Studio:

Table 6: Ready-to-use interpreter

Interpreters	Description
fcc-python Interpreter	The fcc-python interpreter is used to write Python code in a notebook to analyze data from different sources, machine learning, artificial intelligence, etc. In the fcc-python interpreter, you can configure the Python installed path, set the maximum number of results that must be displayed, change the Python version, add Python Packages, etc. The python interpreter uses a python virtual environment. Out-of-box Compliance Studio comes with three variants of python interpreters, fcc-python, fcc-python-ml4aml, and fcc-python-sane.
jdbc Interpreter	The jdbc interpreter is a ready-to-use interpreter used to connect to Studio schema. This Interpreter is used to connect and write SQL queries on any schema without any restriction. In the jdbc Interpreter, you can configure schema details, link Wallet Credentials to the jdbc Interpreter, etc.
md Interpreter	The md interpreter is used to configure the markdown parser type. This Interpreter displays text based on Markdown, which is a lightweight markup language. The connection does not apply to this Interpreter.
pgql Interpreter *(part of PGX interpreter)*	The pgql interpreter is a ready-to-use interpreter used to connect the configured PGX server. This Interpreter is used to perform queries on the graph in Compliance Studio. PGQL is a graph query language built on top of SQL, bringing graph pattern matching capabilities to existing SQL users and new users interested in graph technology but who do not have an SQL background.
pgx-python *(part of PGX interpreter)*	The pgx-python interpreter is a ready-to-use interpreter used to connect to the configured PGX server. It is a **python** based interpreter with a PGX python client embedded in it to query on graph present in the PGX server. By default, this Interpreter points to ml4aml Python Virtual environment.
pgx-algorithm Interpreter *(part of PGX interpreter)*	The pgx-algorithm interpreter is a ready-to-use interpreter that connects to the configured PGX server. This Interpreter is used to write an algorithm on the graph and is also used in the PGX interpreter.
pgx-java Interpreter *(part of PGX interpreter)*	The pgx-java interpreter is a ready-to-use interpreter that connects to the configured PGX server. It is **Java11** based interpreter with a PGX client embedded in it to query on graph present in the PGX server.

Table 6: Ready-to-use interpreter

pyspark Interpreter	The pyspark interpreter connects to the big data environment by default. Users must write code for connection either in the Initialization section or in the notebook's paragraph. This Interpreter is used to write the pyspark language to query and perform analytics on data present in big data. This requires additional configuration, which must be performed as a prerequisite or as post-installation with the manual change of interpreter settings. In the pyspark Interpreter, you can configure the Python binary executable to use for PySpark in both driver and workers, set true to use IPython, else set to false, etc.
spark Interpreter	The spark interpreter connects to the big data environment by default. Users must write for connection either in the Initialization section or in the notebook's paragraph. This Interpreter is used to perform analytics on data present in the big data clusters in the Scala language. This requires additional configuration, which must be performed as a prerequisite or as post-installation with the manual change of interpreter settings. In the spark interpreter, you can configure the cluster manager to connect, print the Read Eval Print Loop (REPL) output, the total number of cores to use, etc.
ore Interpreter	The ore Interpreter has been deprecated. We do not recommend using this interpreter since it will be removed in future versions of OFS Compliance Studio. We will be introducing “R” Interpreter instead of ore Interpreter.

4.1.1 fcc-python Interpreter

In Compliance Studio, the python interpreter uses a python virtual environment. Out-of-box Compliance Studio comes with three variants of python interpreters, fcc-python, fcc-python-ml4aml, and fcc-python-sane.

Each interpreter variant points to a different virtual environment.

Table 7 lists the fcc-python interpreter variants:

Table 7: fcc-python interpreter variants

Interpreter Variant	Virtual Environment Name	Description
fcc-python	defaultVirtualEnv	Default python interpreter.
fcc-python-ml4aml	ml4aml	Python interpreter for AIF and AMLES.
fcc-python-sane	saneVirtualEnv	Python interpreter for scoring Name and Address Matching.

We expect users to use the default python interpreter or new variants with their python version and python packages. For more info on creating a new python interpreter variant with the new virtual environment.

The libraries for the following Interpreter Variant:

- Default python interpreter has **Python 3.6.13** with the following libraries:

  ```
  Package               Version
  --------------------- ---------
  certifi               2020.6.20
  charset-normalizer    2.0.10
  conda-pack            0.6.0
  cx-Oracle             7.3.0
  cycler                0.10.0
  ds-interpreter-client 21.4.9
  idna                  3.3
  imbalanced-learn      0.6.2
  joblib                0.14.1
  kiwisolver            1.2.0
  matplotlib            3.3.3
  mmg                   8.1.1
  numpy                 1.19.2
  pandas                1.1.5
  Pillow                7.2.0
  pip                   21.3.1
  py4j                  0.10.7
  pyparsing             2.4.7
  python-dateutil       2.8.1
  pytz                  2020.1
  requests              2.27.1
  scikit-learn          0.23.2
  scipy                 1.5.2
  seaborn               0.9.1
  setuptools            58.0.4
  six                   1.15.0
  SQLAlchemy            1.3.11
  threadpoolctl         2.1.0
  urllib3               1.26.8
  wheel                 0.37.1
  ```

```
xgboost               1.0.1
```

- The python-ml4aml interpreter has **Python 3.7.7** with the following libraries:

```
Package               Version
--------------------- ---------
sqlalchemy            1.3.11
textblob              0.15.3
nltk                  3.4.5
xgboost               1.0.1
seaborn               0.11.0
scikit-learn          0.22.1
SHAP                  0.34.0
ELI5                  0.10.1
PDPbox                0.2.0
Pyspark               2.4.5
Bayesian Optimization 1.1.0
Flask Restful         0.3.8
Imbalanced learn      0.6.2
gunicorn              20.0.4
hyperopt              0.2.4
py4j                  0.10.9
scikit-optimize       0.7.4
statsmodels           0.11.1
pyod                  0.8.1
Cx_oracle             7.3.0
numpy                 1.19.2
scipy                 1.3.2
pandas                0.25.3
matplotlib            3.2.2
```

- The python-sane interpreter has **Python 3.6.13** with the following libraries:

```
Package               Version
--------------------- -------------
addressmatching       0.2.3
catboost              0.24.1
certifi               2021.5.30
charset-normalizer    2.0.12
conda-pack            0.6.0
```

```
cx-Oracle             7.3.0
cycler                0.10.0
deprecation           2.1.0
ds-interpreter-client 21.4.9
globalparty           8.1.2.0.0rc22
graphviz              0.14.1
greenlet              1.1.2
idna                  3.3
importlib-metadata    4.8.3
jaro-winkler          2.0.0
jellyfish             0.8.2
jep                   3.9.1
kiwisolver            1.2.0
matplotlib            3.3.2
mmg                   8.1.1
namematching          0.2.3
numpy                 1.19.2
packaging             20.4
pandas                1.1.5
Pillow                7.2.0
pip                   21.3.1
plotly                4.11.0
py4j                  0.10.7
pyparsing             2.4.7
python-dateutil       2.8.1
python-Levenshtein    0.12.0
pytz                  2020.1
pyxDamerauLevenshtein 1.6.1
requests              2.27.1
retrying              1.3.3
sane-common           0.2.3
scipy                 1.5.2
setuptools            58.0.4
six                   1.15.0
SQLAlchemy            1.4.31
textdistance          4.2.0
```

`typing_extensions`	`4.1.1`
`urllib3`	`1.26.8`
`wheel`	`0.37.1`
`zipp`	`3.6.0`

Topics:

- Configure an fcc-python Interpreter
- Change Python Version in the fcc-python Interpreter
- Add or Modify Python Packages to the fcc-python Interpreter

4.1.1.1 Configure an fcc-python Interpreter

To configure an fcc-python interpreter variant, follow these steps:

1. On the Interpreter page LHS menu, select fcc-python. The fcc-python interpreter pane is displayed.

2. On Interpreter Settings page, expand **Interpreter Client Configurations** and click Edit ✎ icon for **<Class Name> (zeppelin).** The Interpreter Client Configurations Window is displayed.

3. Enter the following information in the fcc-python interpreter variant pane as tabulated in the Table 7.

Table 8: fcc-python interpreter settings

Field	Description
zeppelin.python	Enter the Python installed path. The value points to the default Python version set for the Interpreter. **NOTE**: To use a different Python version, see Change Python Version in the fcc-python Interpreter section.
zeppelin.python.useIPython	Set to **True** to use IPython, else set to **False**.
zeppelin.python.maxResult	Enter the maximum number of results that must be displayed.
zeppelin.interpreter.output.limit	Output message from interpreter exceeding the limit will be truncated.

4.1.1.2 Change Python Version in the fcc-python Interpreter

In the fcc-python Interpreter, the Linux console uses the default python version in. `/user/fccstudio/python_user/bin/python` as value. If you want to modify the python version, either you can create an interpreter variant or modify the existing python version in the same interpreter variant.

> **NOTE** The **python2** is the default version used in the Linux console and is no longer supported. Hence, you can use any version of **python3** or any virtual environment with a specific python version or a specific version of python packages.

To use a different version of Python, follow these steps:

1. Navigate to the **fcc-python** Interpreter Settings page.

2. Expand **Interpreter Client Configurations** and click Edit ✎ icon for <Class Name> (zeppelin). The Interpreter Client Configurations Window is displayed.
3. Click `zeppelin.properties`. The Properties window is displayed.
4. Change the default Python version in the `Default Value` parameter to the new version. `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/python-packages/defaultVirtualEnv/bin/<Python Version>`.

   By default, it is python3.

   For example, `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/python-packages/defaultVirtualEnv/bin/python3`.
5. Create a new interpreter variant and configure the version in the `Default Value` parameter. For information on creating a new interpreter variant, see Create an Interpreter Variant section. For example, to use Python 3.6.13, create a new fcc-python interpreter variant and enter the value as python 3.6.13.

4.1.1.3 Add or Modify Python Packages to the fcc-python Interpreter

When a user wants to write something in Python, but the packages are not present. Use case: ML or AI code. By default, the Linux server has a limited number of packages present inside it.

To add desired Python packages to the fcc-python Interpreter, follow these steps:

- For Compliance Studio installed on-premise:

  To add or modify Python libraries to the fcc-python Interpreter, contact System Administrator to install the required additional Python libraries on the Processing Server (Studio Notebook Server). The newly added Python libraries must be accessible to the Linux user for Compliance Studio.

To add the python packages for python3, follow these steps:

1. Navigate to the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/python-packages/bin` directory.
2. Run the following command:

   ```
   python3 -m pip install <package name> --user
   ```

4.1.2 jdbc Interpreter

The jdbc Interpreter is a ready-to-use interpreter used to connect Studio schema without OFSAA. This Interpreter is used to connect and write SQL queries on any schema without any restriction. The jdbc interpreter has no security attributes. It can be used to access any schema. In the jdbc interpreter, you can configure schema details, link Wallet Credentials to the jdbc Interpreter, etc.

Topics:

- Configure a jdbc Interpreter Variant
- Link Wallet Credentials to jdbc Interpreter

4.1.2.1 Configure a jdbc Interpreter Variant

To configure a jdbc interpreter variant, follow these steps:

1. On the Interpreter page LHS menu, select jdbc. The jdbc interpreter pane is displayed.

2. On Interpreter Settings page, expand **Interpreter Client Configurations** and click Edit ✎ icon for **<Class Name> (zeppelin).** The Interpreter Client Configurations Window is displayed.
3. Enter the following information in the jdbc interpreter variant pane as tabulated in the Table 9.

Table 9: jdbc interpreter settings

Field	Description
common.max_count	Enter the maximum number of SQL results to display.
default.completer.schemaFilters	Enter comma-separated schema filters to get metadata for completions.
default.completer.ttlInSeconds	Enter the time to live SQL completer in seconds.
default.driver	Enter the default JDBC driver name.
default.password	Enter the default password. **NOTE:** This value can be null if you have entered the alias name in the `default.url` parameter for the jdbc interpreter.
default.precode	SQL, which executes while opening the connection.
default.statementPrecode	Runs before each run of the paragraph in the same connection.
default.splitQueries	Each query is executed apart and returns the result. Specify the presence of default split queries. Enter True to split or 'False' not to.
default.url	Enter the jdbc URL. NOTE: If you want to use the Oracle wallet credentials, you must enter the alias name in the following format: `jdbc:oracle:thin:@<alias_name>`
default.user	Enter the name of the default user.
ofsaa.metaservice.url	Enter the metaservice URL. For example, `http://<HOSTNAME>:7045/metaservice` Here, `<HOSTNAME>` refers to the server name or IP address where the compliance studio is installed.
ofsaa.sessionservice.url	Enter the session service URL. For example, `http://<HOSTNAME>:7047/sessionservice` Here, `<HOSTNAME>` refers to the server name or IP address where the compliance studio is installed.
pgx.baseUrl	Enter the PGX Base URL. This is the location where the data is pushed. For example, `http://<HOSTNAME>:7007`
zeppelin.jdbc.auth.type	Enter the default JDBC authentication type. The authentication methods supported are SIMPLE and KERBEROS

Table 9: jdbc interpreter settings

zeppelin.jdbc.concurrent.max_connection	Enter the number of maximum connections allowed.
zeppelin.jdbc.concurrent.use	Specify concurrent use of JDBC connections. Enter True to enable or 'False' to disable.
zeppelin.jdbc.interpolation	Enable ZeppelinContext variable interpolation into paragraph text.
zeppelin.jdbc.keytab.location	Enter the keytab file location.
zeppelin.jdbc.maxConnLifetime	Maximum of connection lifetime in milliseconds. A value of zero or less means the connection has an infinite lifetime.
zeppelin.jdbc.maxRows	The maximum number of rows fetched from the query.
zeppelin.interpreter.output.limit	Output message from interpreter exceeding the limit will be truncated.
zeppelin.jdbc.principal	Enter the principal name to load from the keytab file.

4.1.2.2 Link Wallet Credentials to jdbc Interpreter

Compliance Studio provides secure and safe credential management. Examples of credentials are passwords, Oracle Wallets, or KeyStores. Use this section to link credentials (a wallet and a password) to the jdbc interpreter variant to enable secure data access. This linking enables the jdbc interpreter to securely connect to the specified Oracle database. For more information on linking Wallet Credentials to jdbc Interpreter, see the Link Credentials section.

> **NOTE** The Credentials section is enabled if an interpreter variant can accept credentials.

You can also create new credentials and link to jdbc Interpreter. For more information, see Create a Credential section.

4.1.3 md Interpreter

This Interpreter displays text based on Markdown, which is a lightweight markup language. In the md interpreter, you can configure the markdown parser type. Markdown (md) is a plain text formatting syntax designed so that it can be converted to HTML. Use this section to configure the markdown parser type.

To configure the md interpreter variant, follow these steps:

1. On the md Interpreter page LHS menu, select md. The md interpreter pane is displayed.
2. On Interpreter Settings page, expand **Interpreter Client Configurations** and click Edit ✎ icon for **<Class Name> (zeppelin).** The Interpreter Client Configurations Window is displayed.
3. Enter the markdown parser type and click **Update**. To confirm the modified configuration.

4.1.4 PGX Interpreter

The PGX has the following interpreters:

- **pgql**: The pgql interpreter is a ready-to-use interpreter used to connect the configured PGX server. This Interpreter is used to perform queries on the graph in Compliance Studio. PGQL is a graph query language built on top of SQL, bringing graph pattern matching capabilities to existing SQL users and new users interested in graph technology but who do not have an SQL background.
- **pgx-algorithm:** The pgx-algorithm is a ready-to-use interpreter used to connect to the configured PGX server. This Interpreter is used to write an algorithm on the graph and is also used in the PGX interpreter.
- **pgx-java**: The pgx-java interpreter is a ready-to-use interpreter used to connect to the configured PGX server. It is **Java11** based interpreter with a PGX client embedded in it to query on graph present in the PGX server.
- **pgx-python**: The pgx-python interpreter is a ready-to-use interpreter used to connect to the configured PGX server. It is a **python** based interpreter with a PGX python client embedded in it to query on graph present in the PGX server. By default, this Interpreter points to ml4aml Python Virtual environment.

To configure the pgql interpreter variant, follow these steps:

1. On the Interpreter page LHS menu, select pgql. The pgql interpreter pane is displayed.
2. On Interpreter Settings page, expand **Interpreter Client Configurations** and click Edit ✎ icon for **<Class Name> (zeppelin).** The Interpreter Client Configurations Window is displayed.
3. Enter the following information in the pgql interpreter variant pane as tabulated in the Table 10.

Table 10: PGX interpreter

Field	Description
graphviz.formatter.class	Enter the class which implements the formatting of the visualization output. For example, `oracle.datastudio.graphviz.formatter.DataStudioFormatter`
graphviz.driver.class	Enter the class which implements the PGQL driver. For example: `oracle.pgx.graphviz.driver.PgxDriver`
base_url	Enter the base URL of the PGX. For example, `http://<HOSTNAME>:7007`
zeppelin.interpreter.output.limit	Enter the output message limit. Any message that exceeds the limit is truncated. For example, 102 or 400.
num_cached_resultsets	Maximum number of results sets kept open on the PGX server per interpreter session. Only checked when the interpreter is used, and therefore it should only be used with expiring interpreter sessions. For example: 5
resultset_expiration_time_secs	Number of seconds after which unused results sets are closed on the PGX server. Only checked when interpreter session is used and should only be used with expiring interpreter sessions. For example: 3600
zeppelin.python.useIPython	Set to 'True' to use IPython, else set to 'False'.

Table 10: PGX interpreter

zeppelin.python	Enter the Python installed path. The value points to the default Python version set for the Interpreter. **NOTE**: To use a different Python version, see Change Python Version in the fcc-python Interpreter section.

4.1.5 pyspark Interpreter

Users must write for connection either in the Initialization section or in the notebook's paragraph. This interpreter is used to write the pyspark language to query and perform analytics on data present in big data. This requires additional configuration, which must be performed as a prerequisite or as post-installation with the manual change of interpreter settings.

In the pyspark interpreter, you can configure the Python binary executable for PySpark in both driver and workers, set 'True' to use IPython, else set it to 'False'.

To configure the pyspark interpreter variant, follow these steps:

1. On the Interpreter page LHS menu, select pyspark. The pyspark interpreter pane is displayed.
2. On Interpreter Settings page, expand **Interpreter Client Configurations** and click Edit ✎ icon for **<Class Name> (zeppelin).** The Interpreter Client Configurations Window is displayed.
3. Enter the following information in the pyspark interpreter variant pane as tabulated in the Table 11.

Table 11: pyspark interpreter

Field	*Description*
zeppelin.pyspark.python	Enter the Python binary executable for PySpark in both drivers and workers. The default value is python. For example, `python`
zeppelin.pyspark.useIPython	Set to 'True' to use IPython, else set to 'False'.
zeppelin.interpreter.output.limit	Output message from interpreter exceeding the limit will be truncated

4.1.6 spark Interpreter

The spark Interpreter does not connect to any schema by default. Users must write for connection either in the Initialization section or in a notebook's paragraph. This interpreter performs analytics on data present in Big data clusters in the Scala language. This requires additional configuration, which must be performed as a pre-requisite or as post-installation with the manual change of interpreter settings.

In spark interpreter, you can configure the cluster manager to connect, print the Read–eval–print loop (REPL) output, the total number of cores to use, etc.

To configure the spark interpreter variant, follow these steps:

1. On the Interpreter page LHS menu, select spark. The spark interpreter pane is displayed.

2. On Interpreter Settings page, expand **Interpreter Client Configurations** and click Edit ✎ icon for **<Class Name> (zeppelin).** The Interpreter Client Configurations Window is displayed.

> **NOTE** The user must select the pyspark Class Name.
>
> For example, org.apache.zeppelin.spark.SparkInterpreter.

3. Enter the following information in the spark interpreter variant pane as tabulated in the Table 12.

Table 12: spark interpreter

Field	Description
pgx.baseUrl	Enter the PGX Base URL. This is the location where the data is pushed. For example, `http://<HOSTNAME>:7007`
spark.executor.memory	Enter the amount of memory to use for the executor process. Executor memory per worker instance. For example, 512m and 32g. In Spark, the executor-memory flag controls the executor heap size (similarly for YARN and Slurm). The default value is 512MB per executor. In addition, the driver-memory flag controls the amount of memory to allocate for a driver, which is 1GB by default and should be increased in case you call a collect or take(N) action on a large RDD inside your application.
spark.master	Enter the cluster manager to connect. For example, `local[*]`
spark.yarn.archive	Enter the archive containing the required. Spark jars for distribution to the YARN cache make Spark runtime jars accessible from the YARN side.
spark.app.name	Enter the name of the application. For example, `Zeppelin`
zeppelin.spark.ui.hidden	Set to True or False.
zeppelin.spark.maxResult	Enter the maximum number of results that must be fetched.
spark.pyspark.python	Enter the Python binary executable for PySpark in both driver and executors. For example, `python`
zeppelin.spark.enableSupportedVersionCheck	Set to 'True' or 'False'.
args	Enter the Spark command-line args.
zeppelin.spark.useNew	Set to 'True' to use the new version of the SparkInterpreter.
zeppelin.spark.useHiveContext	Set to 'True' to use HiveContext instead of SQLContext.
zeppelin.spark.uiWebUrl	Overrides Spark UI default URL. Value should be a full URL `(http://{hostName}/{uniquePath})`
zeppelin.spark.printREPLOutput	Enter to print the REPL output.

Table 12: spark interpreter

spark.cores.max	Enter the total number of cores to use. NOTE: Empty value uses all available cores.
spark.driver.bindAddress	Hostname or IP address where to bind listening sockets.
zeppelin.interpreter.output.limit	Output message from interpreter exceeding the limit will be truncated.

4.1.7 ore Interpreter

The ore Interpreter has been deprecated. We do not recommend using this interpreter since it will be removed in future versions of OFS Compliance Studio. We will be introducing “R” Interpreter instead of ore Interpreter.

4.2 Link Credentials

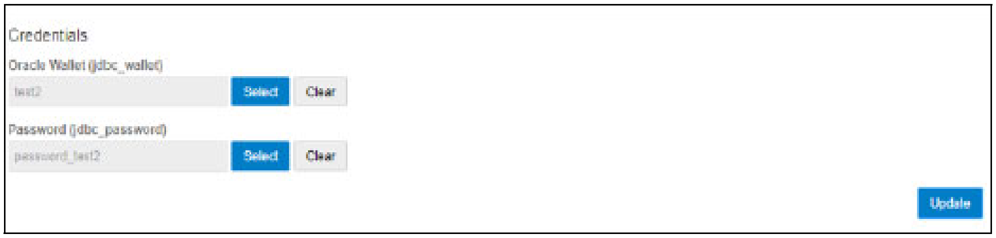
Compliance Studio provides secure and safe credential management. Examples for credentials are passwords, Oracle Wallets, or KeyStores. Use this section to link credentials (a wallet and a password) to the jdbc interpreter variant to enable secure data access. This linking enables the jdbc interpreter to securely connect to the specified Oracle Database. You can also create new credentials to connect to the new interpreter variants based on your requirement. For more information, see Create a Credential section.

NOTE You can link credentials only for jdbc interpreters. The Credential section is enabled if an Interpreter variant can accept credentials.

To link ready-to-use credentials to the required interpreters, follow these steps:

1. On the Interpreters page, select the required interpreters. For example, jdbc.
2. Go to the **Credentials** section.

Figure 11: Credentials screen

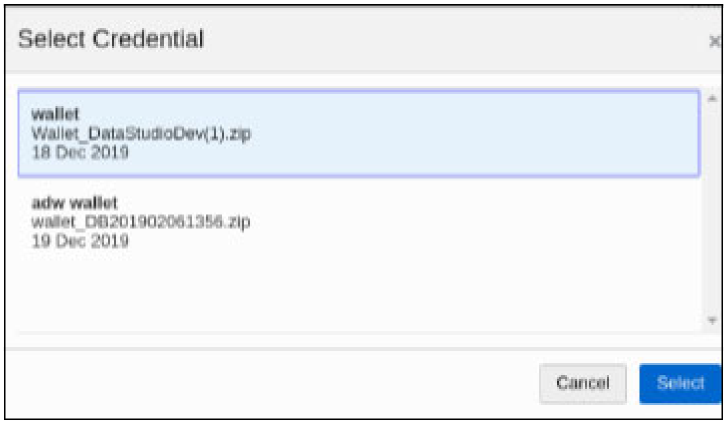


test2

password_test2

3. To select the Oracle Wallet (jdbc wallet) credential you want to link to the Interpreter variant, click Select. The Select Credential dialog is displayed.

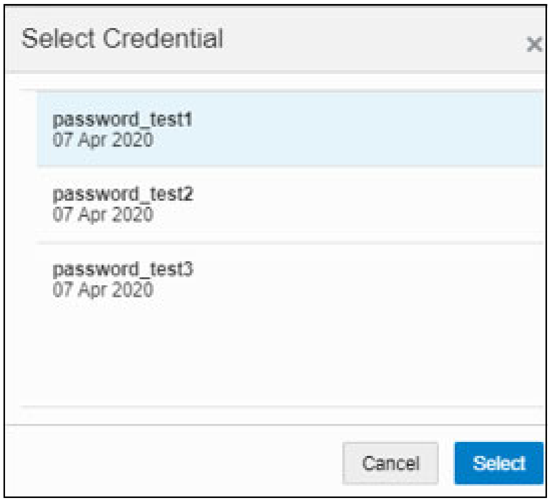
Figure 12: Select Credential's screen



4. Select the required Oracle Wallet (jdbc wallet).

5. To select the Password (jdbc password) that you want to link to the Interpreter variant, click Select. The Select Credential dialog is displayed.

Figure 13: Select Credential screen



6. Select the required Password (jdbc password). Click **Select**.

7. Click **Update** to save the changes. The required password and Oracle Wallet are linked to the jdbc Interpreter.

4.3 Create a Credential

New credentials are created when database details are changed or updated. For example, change in Transparent Network Substrate (TNS) due to hostname change or compulsory periodic update of schema passwords.

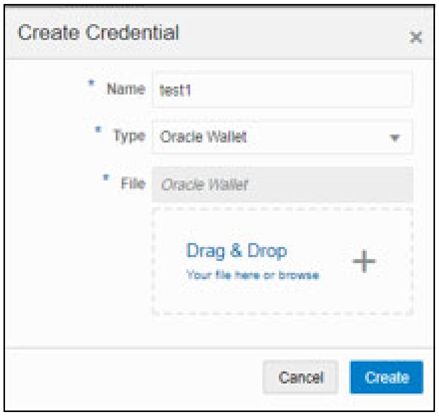
Oracle Wallet provides a simple and easy method to manage database credentials across multiple domains. It allows you to update database credentials by updating the Wallet instead of having to change individual data source definitions.

Use this section to add a new credential to the interpreters.

To create a credential, follow these steps:

1. On the Compliance Studio workspace LHS Menu, click Credentials. The Credentials page is displayed.
2. Click **Create**. The Create Credential dialog is displayed.

Figure 14: Create Credential screen



Create Credential

* Name: test1

* Type: Oracle Wallet

* File: Oracle Wallet

3. Enter the following information in the Create Credential dialog box as tabulated in the Table 13:

Table 13: Create Credential dialog box

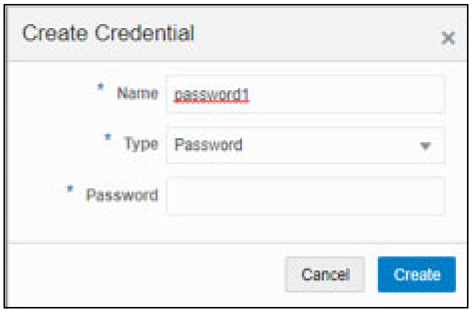
Field	Description
Name	Enter the name for the wallet credential.
Type	Select Oracle Wallet.
File	Upload the wallet zip file that includes the following files: ● `cwallet.sso` ● `ewallet.p12` ● `tnsnames.ora` **NOTE:** ● The wallet file must be in .zip format. ● The maximum file size allowed for the credential file is 128Kb.

4. Click **Create**. The wallet credential is created and displayed on the Credentials page.

To create a new password credential for the wallet, follow these steps:

1. Click **Create**. The Create Credential dialog is displayed.

Figure 15: Create Credential screen


Create Credential

* Name password1

* Type Password

* Password

2. Enter the following information in the Create Credential dialog as tabulated in the Table 14.

Table 14: Create Credential dialog

Field	Description
Name	Enter the name for the password credential.
Type	Select password type from the drop-down (wallet or keytab).
Password	Enter the wallet password for the password credential.

3. Click **Create**. The password is created for the wallet and displayed on the Credentials page.
4. Click the credential file name on the Credentials page to download the credential files.
5. To delete a required credential, click **Delete** 🗑. The credential is removed from the list.

4.4 Create an Interpreter Group

In Compliance Studio, you can either use a default interpreter group or create a new group for an interpreter. You can create more than one group for an interpreter. Multiple groups for an interpreter are created to connect different versions of interpreters (Python version: 3, Python version: 2) and connect a different set of users and database schema. For example, Compliance Studio schema, BD schema, etc.

To create a new interpreter group, follow these steps:

1. On the Interpreters page, click the required interpreters from the LHS list. For example, jdbc interpreter.
2. The default interpreter group is displayed on the RHS.
3. On the default interpreter, click **Clone** button to create a new group. The Create Interpreter Group dialog box is displayed.
4. Enter the Name for the new interpreter group. Click **Create**. A new group is created with a name, **<Interpreter Type>.<Group Name>.**
5. Provide the new schema details, such as the default.url, default.user, and default.password.

4.5 Create an Interpreter Variant

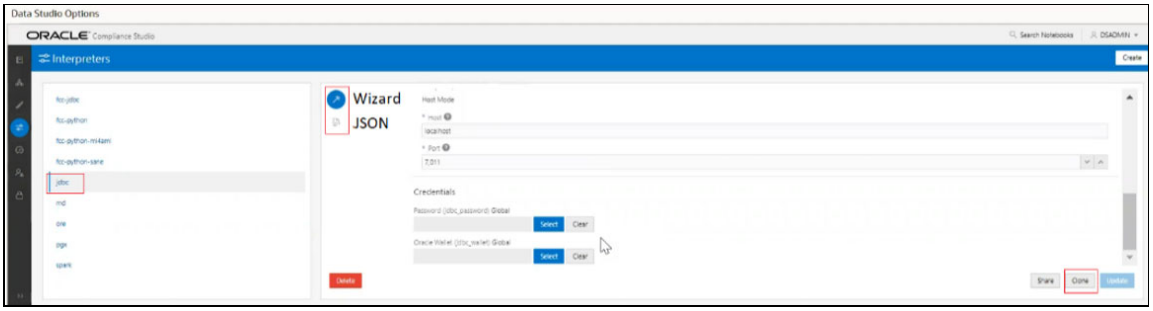
1. Log in to the Compliance Studio application.
2. Lunch the **CS Production** Workspace.

3. Hover the mouse over the Data Studio Options widget and Click **Interpreters**.

   By default, the Interpreters page lists all the available interpreters.

4. Click **jdbc** interpreter on the LHS. The default configured interpreter variant is displayed on the RHS:

   Figure 16: jdbc interpreter screens



* Host: localhost

* Port: 7,011

5. Click **Clone** on the RHS. The pop-up window displayed for the group name.

6. Enter the group name in the **Group Name** text box and click **Create**. The new group is created and displayed on LHS.

7. Click **<New group name>** on the LHS. The default configured interpreter variant is displayed on the RHS.

   You can modify the values in the interpreter properties in the JSON file or Wizard view.

4.6 Enable Additional Spark or PySpark interpreter

Interpreter variants do not apply to Spark or PySpark interpreters. Hence, you must enable an additional set of interpreters.

To enable an additional Spark or PySpark interpreter, see Enabling an Additional Spark or PySpark Interpreter chapter in the OFS Compliance Studio Installation Guide.

5 Schedule Scenario Notebook Execution

It is recommended to use the scheduler to execute the notebook in Batch.

Topics:

- *Prerequisites*
- Using Scheduler
- Using Shell Script

5.1 Prerequisites

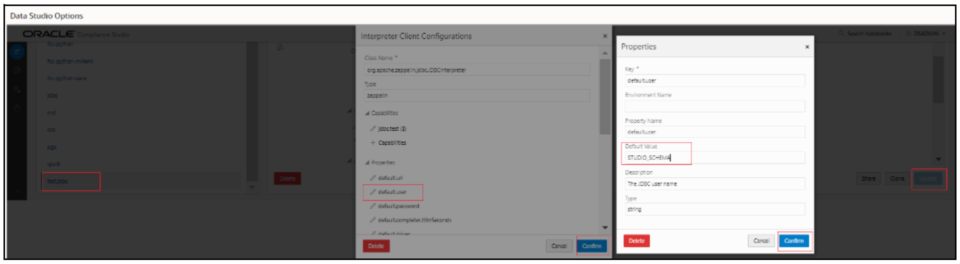
After installation, you need to create a new variant of the interpreter and change the schema from **STUDIO_SCHEMA** to **BD_SCHEMA** to execute Scenario notebooks.

To create a new variant and change the schema:

1. To create a new variant, see Create an Interpreter Variant section.
2. Click **<New group name>** on the LHS. The default configured interpreter variant is displayed on the RHS.

   You can modify the values in the interpreter properties in the JSON file or Wizard view.
3. On the Interpreter Settings page, expand **Interpreter Client Configurations** and click the Edit icon for **<Class Name> (zeppelin).** The Interpreter Client Configurations Window is displayed. Click **defaultuser** property. The property window is displayed.

Figure 17: Change the Schema value



Interpreter Client Configurations

Class Name *: org.apache.zeppelin.jdbc.JDBCInterpreter

Type: zeppelin

Capabilities: Jdbc.test ($)

Properties: default.url, default.user, default.password, default.completer.ttlInSeconds, default.driver

Delete Cancel Confirm

Properties

Key *: default.user

Environment Name:

Property Name: default.user

Default Value: STUDIO_SCHEMA

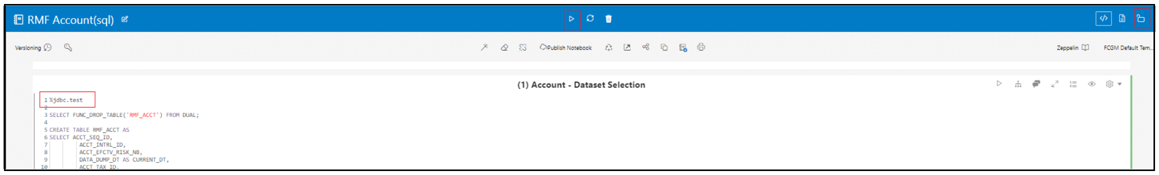
Description: The JDBC user name

Type: string

Delete Cancel Confirm

4. On the property window, change the value from STUDIO_SCHEMA to BD_SCHEMA in the **Default Value** text box. Click **Confirm**.
5. Click **Confirm** and click **Update**.
6. On RHS, click on JSON view and copy the interpreter's name that is required to update the interpreter name under each paragraph in the scenario notebook.
7. Navigate to the Compliance Studio server with the same URL by changing the port to 7008. `(http://hostname:7008 from http://hostname:7001/cs/home )`
8. Open the scenario notebook (**RMF Account(sql)**), unlock the notebook, and replace it with the new interpreter name in each paragraph.

Figure 18: Scenario notebook



```
%jdbc.test

SELECT FUNC_DROP_TABLE('RMF_ACCT') FROM DUAL;

CREATE TABLE RMF_ACCT AS
SELECT ACCT_SEQ_ID,
       ACCT_INTRL_ID,
       ACCT_EFCTV_RISK_NB,
       DATA_DUMP_DT AS CURRENT_DT,
       ACCT_TAX_ID,
```

9. Click Run Paragraph's execute ▷ icon to execute the notebook.

5.2 Using Scheduler

To schedule a notebook for execution using the scheduler, see the OFS Scheduler Service User Guide.

For more details, see Example of Creating a batch in Scheduler for Notebook Execution section.

5.3 Using Shell Script

NOTE This is deprecated in the current release and will be removed in the future release. It is recommended to use the scheduler to execute the notebook.

A notebook is a collection of documentation and snippets of executable code. The notebook allows large scripts to be broken into a modular collection of executable code with tailored results. Different languages, such as Scala, Python, and Oracle's own property graph query language (PGQL), can be combined into one notebook. Each notebook is mapped to the logged-in user.

When a notebook is published:

- The original notebook is cloned, and a published notebook is created.
- Any changes made to the original notebook will have no impact on the published notebook.
- Whenever the original notebook is re-published, a new version of the published notebook is created.
- The published notebook is in a read-only format.
- The published notebook can be run from the shell script.

The published notebook can be scheduled for execution with a set of threshold values required for generating alerts or trends.

NOTE

- Notebook Execution through shell script will be successful even if some paragraph fails in the notebook.
- Ensure there is no blank paragraph in the notebook before executing the notebook using shell script. If a blank paragraph exists in the notebook, the notebook will be executed continuously and not be completed.

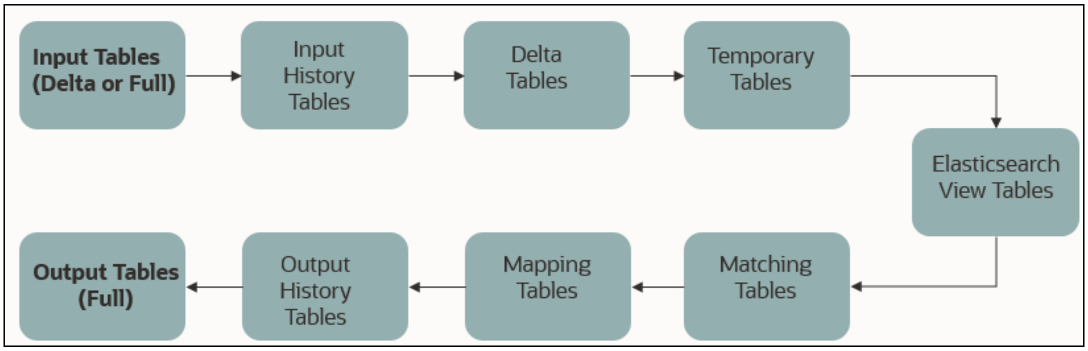
To execute the published scenario notebook with OFSAA, Create and Execute a Run Executable for Scenario Notebooks section.

6 Entity Resolution

OFS Compliance Studio provides Entity Resolution (ER) capability. It allows firms to break through barriers in their data by gaining single views of their customers and their external entities and have the choice of monitoring them both under one consolidated Global Party.

OFS Compliance Studio Entity Resolution is a configurable process that allows data to be matched and merged to create contextual links in the global graph or resolve relational party records to a global party record as part of ingestion. OFS Compliance Studio has pre-built configurations supporting matching (or linking) in the FCGM and resolving entities in CSA for data being loaded into Financial Services Data Foundation (FSDF).

Figure 19: Entity Resolution



- **Comparison for Delta Processing**

  The first time Entity Resolution runs, it operates on the full data set. This means the initial run will take longer than subsequent runs after the initial processing where deltas (changed records) are calculated (regardless of whether full or delta data is populated in the input tables) so that matching happens only on new and changed records for improved performance.

- **Candidate Selection**

  Elasticsearch is a distributed search and analytical engine for all structured and unstructured data types in OFS Compliance Studio.

- **Matching**

  Matching rules are used to compare entities to identify pairs that refer to the same entity. It creates a probable link between entities by comparing the attributes of the entities.

  For example, deduplicating customers, resolving derived entities, or linking customers or derived entities to external data such as Panama papers or sanctions lists with different rules and thresholds.

  For more information on scoring methods, see the OFS Compliance Studio Matching Guide.

  For more information on creating, see the **Creating a Ruleset** section in the OFS Compliance Studio User Guide.

- **Grouping**

  It is used to Group (entity Ids or Customer Ids) based on similarity links between entities using matching rules and applying the merge rules on similarities. Once it is grouped, the system assigns the global party id to each Group.

NOTE	Grouping is an automatic process. Grouping will be based on the match edges without any configuration.

- **Merge Rules**

  Merging rules are used to group multiple entities or customers into a single global party based on the merge ruleset.

  For more information on creating the Merging Rules, see the **Creating a Ruleset** section in the OFS Compliance Studio User Guide.

- **Persisting**

  Records identified for merging will be collapsed into a single global party record, and a mapping from this global party record to the original party records will be created. Ongoing changes to the original party records may impact the global parties. For more details, see Persisting the Data section.

- **Data Survival**

  When party records are identified for merging, a single output party record is created for the main or parent Dataset. Entity Resolution provides a mechanism to select the best data view from across the multiple-party records using attribute-by-attribute selection functions like Most Common or Maximum. It also provides a mechanism for transforming the child records stored in related tables, such as an address, email, or document ids.

  - **Merge and Split Global Parties**: Entity Resolution provides a mechanism to merge, split, create manually, and rearrange the entities for Global parties. Whenever there is a manual action (merge, split, create, rearrange) to the entities of a global party, the same data survival logic will be applied. See **Using the Merge and Split Global Parties** section in the OFS Compliance Studio User Guide on how to perform the actions.

  For more information on configuring the rules for attribute survival, see the Data Survival section.

NOTE	• When the records are not matched and not merged, they pass straight through and have a one-to-one mapping with the global party. • Where Entity has been resolved/unresolved, data origin is set to **EntRes** for all the records. • The Data Survival job cannot override the manual actions to a global party in batch mode.

Topics:

- Using Preconfigured Datasets and Rulesets
- Creating Pre-Staging Tables in FSDF
- Updating Data Tables for ER with FSDF
- Executing the ER Jobs
- Persisting the Data
- Entity Resolution Metadata

6.1 Using Preconfigured Datasets and Rulesets

6.1.1 Entity Resolution Types

The application is preconfigured to support the following Matching and Entity Resolution types:

- CSA_808
- CSA_811
- CSA_812
- Graph (FCGM)

> **NOTE** Additional types of entity Resolution can be configured. For more information, see the Entity Resolution Metadata section.

For more information on how to run ER in different workspaces, see the **Run ER in Different Workspaces** section in OFS Compliance Studio Installation Guide.

6.1.2 Preconfigured Rulesets for Matching, Merging, and Data Survival

The application provides preconfigured rulesets/rules for Matching, Merging, and Data Survival for the following Matching and Entity Resolution types:

- CSA_808
- CSA_812
- Graph (FCGM) (Matching only)

See the **Creating Rulesets** section in the OFS Compliance Studio User Guide for creating and configuring rulesets.

6.2 Creating Pre-Staging Tables in FSDF

Entity Resolution requires a set of pre-staging tables to be available in the OFSAA staging area and the pre-configured FSDF staging model.

The table definitions are available in terms of a data model file which can be uploaded to OFSAA with the help of AAI's Data model management.

> **NOTE** The `ER_datamodel.xml` file is applicable only for 8.1.2.0.0 version FSDF and 812 pipelined.

Following are the steps to be followed to do the data model upload.

1. Copy `ER_datamodel.xml` from `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/entity-resolution/datamodels/FSDF_8.1.2.0.1/ER_datamodel.xml`

   to `<AAI Application Server>/<FSDF_STG_INFODOM>/erwin/erwinXML`.
2. To upload the Data Model, perform the following:

   c. Model Upload Using **JSON/erwin XML**

   d. Select Upload Mode as **Sliced**

e. Select **Object Registration Mode** as **Incremental Object Registration**

f. Select **Upload File Type** as **XML**

g. Select the **erwin XML** or **Database XML** file for upload from the drop-down list.

Other options can be set to default and proceed to Upload.

For more information on uploading the Data Model, see the **Upload Business Model** section in the Oracle Financial Services Analytical Applications Infrastructure User Guide.

6.3 Updating Data Tables for ER with FSDF

For tables that are in scope for Entity Resolution, copies of the tables need to be created in CSA with the **suffix _PRE**. Data should be loaded into this using an ETL process before Entity Resolution is run. The resolved data is then output to the original CSA tables.

6.3.1 Insert Data into Pre-Staging Tables

Input tables for ER. Data will be processed, and resolved entities will be part of output tables.

> **NOTE** Ensure the pre-staging tables are available in FSDF for 808 and 811, 812. See Creating Pre-Staging Tables in FSDF section.

You can insert the records into Pre-staging tables every day using any one of the following methods:

- **Full Dataset/Full Load**: Insert all the records with the same **fic_mis_date** and process all the records on the same **fic_mis_date**.
- **Delta Dataset/Delta Load**: Insert only the modified and new records based on **fic_mis_date** and process existing identify new or modified records based on new **fic_mis_date**.

The **fic_mis_date** is the date on which the data is entered/loaded in the system.

For example,

- **Day 1**: 1000 records on 1st February (fic_mis_date)
- **Day 2**: 10 records added on 2nd February(fic_mis_date)

If a Full Dataset/Full load, **1000** records on **1st February** and all **1010** records are inserted and processed on **2nd February**.

If Delta load/Delta Dataset, **1000** records on **1st February** and additional **10** records are inserted and processed on **2nd February**.

NOTE	A full load needs to be run on the first day, and then on subsequent days, either full or delta data sets can be loaded into the **PRE** tables. Whether full or delta is run, the output tables will always contain full data for downstream applications to consume. This allows for the handling of deactivated parties due to matching and merging changes. If loading the **PRE** tables with delta only, records that should no longer be included will not be removed from the system. For this reason, a periodic full run may be required.

The following tables are pre-staging tables for FSDF 808:

- **STG_PARTY_MASTER_PRE**: This table contains Customer details, name, DOB, etc. This table contains a person or organization that is a party of financial institutions. Here party refers to the customer, issuer and guarantor, etc. This table will hold the master list of parties and details like party name, age, education, profession, gender etc.
- **STG_PARTY_DETAILS_PRE**: This table contains additional Party details and is an extension of the STG_PARTY_MASTER_PRE table.
- **STG_PARTY_EMAIL_ADDRESS_PRE**: A party can have multiple email addresses. This table identifies all the email addresses that are associated with a party. Email Address is linked to a party via the purpose type for which this email address is used in relation to a party. For example, the purpose could be a Personal Email Address, Business Email Address, etc.
- **STG_PARTY_ADDRESS_PRE**: A party can have multiple addresses. This table identifies all the addresses that are associated with a party. The address is linked to a party via the purpose type for which this address is used about a party. For example, the purpose could be Mailing Address, Business Address, Home Address, etc.

NOTE	There should not be double quotes (“”) special characters in any attributes. Load to Elastic Search will not consider records containing the double quotes in any of the columns. For example, #15, Ground Floor, **“**VK Circle**,”** 1st Main Road**,** Bangalore. VK Circle will not be considered as part of the address in the above address.

- **STG_PARTY_PHONE_PRE**: A party can have multiple phone numbers. This table identifies all the phone numbers that are associated with a party. The phone number is linked to a party via the purpose type for which this phone number is used in relation to a party. For example, Purpose could be Home Phone, Business Phone, Mobile Phone, etc.
- **STG_CUSTOMER_IDENTIFCTN_DOC_PRE**: This table stores the information regarding identification documents provided by customers. There should be a document associated with each Customer Identification Document record. Various documents submitted by the customer are identified by document type as BC- Certificate of Birth, BL- Business License, VR- Vehicle Registration Card or Title, VRC- Voter's Registration Card, etc.

The following tables are pre-staging tables for FSDF 811:

- STG_PARTY_MASTER_PRE
- STG_PARTY_DETAILS_PRE

- STG_PARTY_EMAIL_MAP_PRE
- STG_ADDRESS_MASTER_PRE
- STG_PARTY_ADDRESS_MAP_PRE: It stores the mapping ID for a specific customer ID from STG_ADDRESS_MASTER_PRE.
- STG_PARTY_PHONE_MAP_PRE
- STG_CUSTOMER_IDENTIFCTN_DOC_PRE

The following tables are pre-staging tables for FSDF 812:

- STG_PARTY_MASTER_PRE
- STG_PARTY_DETAILS_PRE
- STG_CUSTOMER_IDENTIFCTN_DOC_PRE
- STG_ADDRESS_MASTER_PRE
- STG_PARTY_ADDRESS_MAP_PRE
- STG_PARTY_PHONE_MAP_PRE
- STG_PARTY_EMAIL_MAP_PRE

6.3.2 Output Tables

The equivalent output tables exist in CSA according to the input tables for the FSDF (808, 811, and 812).

For example, if the input table is **STG_PARTY_MASTER_PRE,** the output table will be **STG_PARTY_MASTER.** It is the same for FSDF 808, 811, and 812.

After executing the Data survival Job, the output tables store the corresponding global party data.

NOTE	<ul><li>By default, the output tables are available in FSDF. The purpose of the tables is the same as the input tables.</li><li>Regardless of Full load or Delta load, the output tables contain the complete set of records with the current **fic_mis_date**. Such global parties can be removed from output tables where mappings have changed, and parties are deactivated.</li></ul>

The following are output tables for FSDF 808:

- STG_PARTY_MASTER
- STG_PARTY_DETAILS
- STG_PARTY_EMAIL_ADDRESS
- STG_PARTY_ADDRESS
- STG_PARTY_PHONE
- STG_CUSTOMER_IDENTIFCTN_DOC

- **FCC_ER_MAPPING:** It stores the mapping between Customer IDs in the input table **STG_PARTY_MASTER_PRE** and Global Party IDs in the output table **STG_PARTY_MASTER**.

The Table 15 describes column details in the FCC_ER_MAPPING.

Table 15: FCC_ER_MAPPING Details

Column Name	Description
V_GLOBAL_ID	It represents the global party id generated after Entity Resolution.
V_ENTITY_ID	It represents the original entity ids. For example, STG_PARTY_MASTER_PRE.V_PARTY_ID
F_LRI_FLAG	It indicates the state of a global id. The expected values are 'Y' or 'N'. 'Y'indicates active and 'N' indicates inactive.
D_CREATED_DATE	It stores the date and timestamp of a newly created Global Id from both ER batches and manual actions. **NOTE:** In case of **add** scenario, the **D_CREATED_DATE** column will be updated for the added entity in a global party. Existing entities will remain unchanged.
D_UPDATED_DATE	It stores the date and timestamp of an updated/deactivated Global Id from ER batches and manual actions. **NOTE:** In case of **split and merge,** the **D_UPDATED_DATE** column will be updated only for the deactivated global ids, and **D_CREATED_DATE** will be updated for the newly generated global ids.
V_ACTION	For information about V_ACTION column, see the Table 16.
V_PIPELINE_ID	It represents the implementation of Entity Resolution flow. For example, you have two pipeline ids for two versions of FSDF (i.e., 811 and 812).
V_COMMENT_ID	It stores the ID reference of the comments that are entered by a user while performing manual actions on a global party from **Manual Decision UI** and **Merge and Split UI**. This column will only store the **Id** and the respective comment will be stored in the **fcc_er_gp_comments** table.
V_MD_FLAG	It stores the state of the records upon which manual actions are taken **from Manual Decision UI** and **Merge and Split UI**. The expected values are: ● MA - Manual Approved / Manual Action ● PMA - Pending Manual Approval ● MR - Manual Rejection **NOTE:** The value in this column will be NULL for the records generated from Entity Resolution batches. The values will be populated for the entities upon which any manual action has been taken from **Merge and Split UI**.
N_RUN_SKEY	It signifies the execution identifier of an Entity Resolution batch. This identifier will be updated for all the impacted entities in an ER batch. For example: When a new global party is created, a new entity is added to an existing global party, an existing global party is split, existing global parties are merged or an existing global party is deactivated.

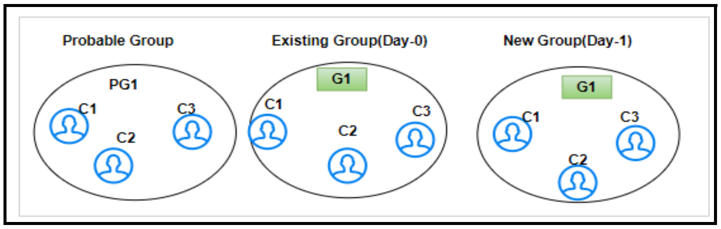
The Table 16 describes **V_ACTION** column in the **FCC_ER_MAPPING.**

Table 16: V_ACTION Details

<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td colspan="2"><b>Batch Execution</b></td></tr>
<tr><td>new global party</td><td>

On the first run of ER batches, the value of the V_action column will be a **new global party** for all the records. In subsequent batches, if there is no change in the existing entities, it will remain the same as new global party.

Figure 20: New Global Party

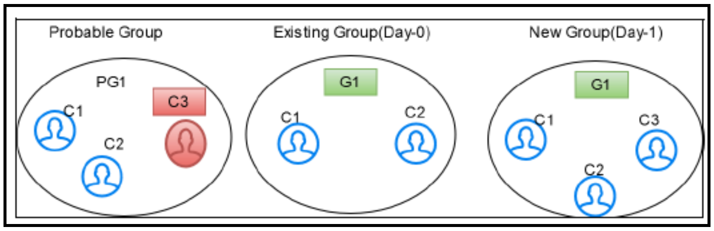


For example, G1 has C1, C2 and C3 entities. After the Day 1 batch execution, if there is no change in the existing group. Still, G1 has C1, C2 and C3 entities with the same global id.

</td></tr>
<tr><td>add</td><td>

If a new entity is available and matches the existing group, then it is defined as **add** in the V_ACTION column for a newly added entity. If a new entity matches the existing group, it will be added to the existing group and assigned the same global id.

Figure 21: Add



For example, G1 has C1 and C2 entities. After the Day 1 batch execution, if C3 entity matches with C1 or C2 then C3 will be added to the existing group G1 with the same global id.

</td></tr>
</table>

Table 16: V_ACTION Details

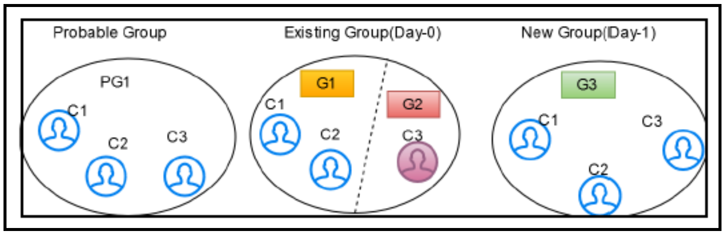
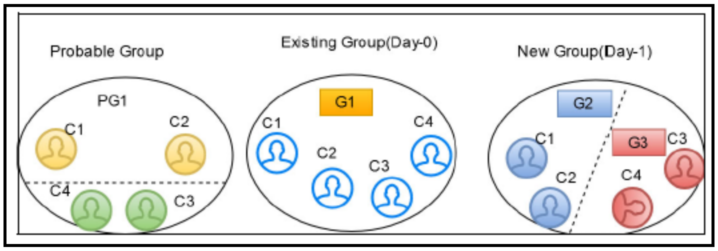
Value	Description
merge	If there is a data change in the entity of a different group and it merges with another group, then it is defined as **merge** in the V_ACTION column for the merged entities. The changed entities can be merged with an existing group with new global id is assigned and the previous global id will be de-activated. **Figure 22: Merge**  Probable Group Existing Group(Day-0) New Group(Day-1) PG1 C1 C2 C3 G1 G2 C1 C2 C3 G3 C1 C3 C2 For example, G1 has C1 and C2 entities and G2 has a C3 entity. After the Day 1 batch execution, if C3 entity matches with an existing group then C3 will be merged into the existing group with a new global id. The V_ACTION column for G3 will merge and G1 and G2 will be deactivated.
split	If there is a data change in the existing group entity which does not matches with other entities of an existing group; then it is defined as **split** in the V_ACTION column for the split entities. The changed entities can be split into a new group and a new global id is assigned to each. **Figure 23: Split**  Probable Group Existing Group(Day-0) New Group(Day-1) PG1 C1 C2 C4 C3 G1 C1 C4 C2 C3 G2 C1 C2 G3 C3 C4 For example, G1 has C1, C2, C3 and C4 entities. After the Day 1 batch execution, if C3 and C4 entities are not matched with the existing entities of the group then C3 and C4 will be split into a new group. G2 has C1 and C2 entities and G3 has C3 and C4 entities with a new global id assigned to each group. The V_ACTION column for G2 and G3 will split and G1 will be deactivated.

Table 16: V_ACTION Details

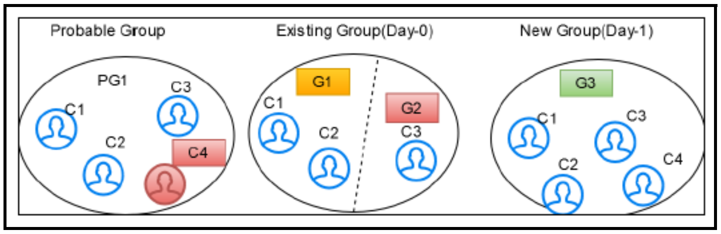
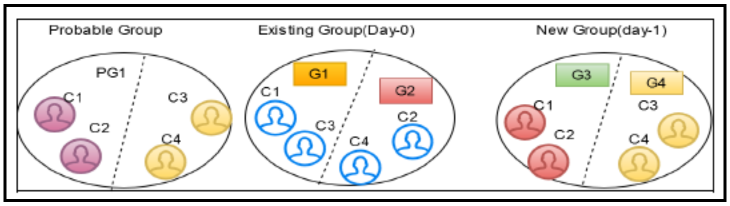
Value	Description
merge and add	If there is a data change in the existing group and a new entity is available, which also matches with the existing group; then it is defined as **merge and add** in the V_ACTION column for the updated and new entities. All the entities are grouped into a single group with a new global id. **Figure 24: Merge and Add**  Probable Group Existing Group(Day-0) New Group(Day-1) PG1 C1 C2 C3 C4 G1 C1 C2 G2 C3 G3 C1 C3 C2 C4 For example, G1 has C1 and C2 entities, G2 has C3 entity. After the Day 1 batch execution, if C4 entity is added newly and C3 entity got changed then common entities are merged into a single group and a new entity is added to the group with a new global id (G3 has C1, C2, C3, and C4 entities). The V_ACTION column for G3 will merge and add, G1 and G2 will be deactivated.
split and merge	If there is a data change in the entity of the first group that matches with another entity of the second group and also an entity from the second group matches with any entity of first group; then it is defined as **split and merge** in the V_ACTION column for affected entities. The changed entities can be split and merged into a new group with a new global id is assigned to each group. **Figure 25: Split and Merge**  Probable Group Existing Group(Day-0) New Group(day-1) PG1 C1 C2 C3 C4 G1 C1 C3 G2 C2 C4 G3 G4 C1 C3 C2 C4 For example, G1 has C1 and C3 entities and G2 has C2 and C4 entities. After the Day 1 batch execution, if C1 matches with C2 and C3 matches with C4 then C2 and C4 will be split separately and merged with C1 and C3 respectively. G3 has C1 and C2 entities and G4 has C3 and C4 entities with a new global id assigned to each group. The V_ACTION column for G3 and G4 will split and merge and G1 and G2 will be deactivated.

Table 16: V_ACTION Details

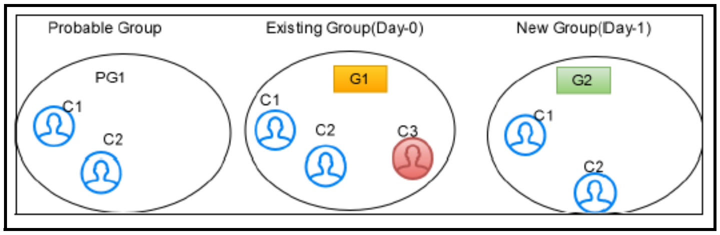
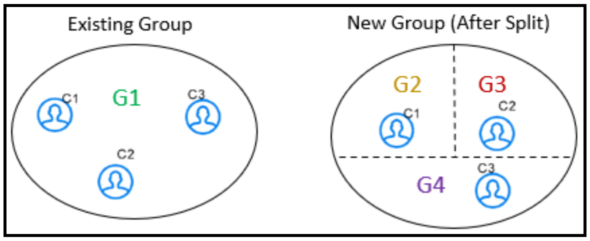
Value	Description
delete	During batch execution, if an entity is unavailable in the existing group, it is defined as **delete** in the V_ACTION column and a new global id is assigned to the remaining entities. **Figure 26: Delete**  Probable Group Existing Group(Day-0) New Group(Day-1) PG1 C1 C2 G1 C1 C2 C3 G2 C1 C2 For example, G1 has C1, C2, and C3 entities. After the Day 1 batch execution, if C3 is deleted from the existing group then G2 has C1 and C2 entities with a new global id. The V_ACTION column for G2 will delete and G1 will be deactivated.
Manual Action	
split	You can split the entities into different groups with new global ids assigned to each. **Figure 27: Split**  Existing Group New Group (After Split) C1 G1 C3 C2 G2 G3 C1 C2 G4 C3 For example, G1 has C1, C2, and C3 entities. After split, G2 has C1, G3 has C2 and G4 has C3 with new global ids assigned to each group. The V_ACTION column for G2, G3 and G4 will split and G1 will be deactivated.

Table 16: V_ACTION Details

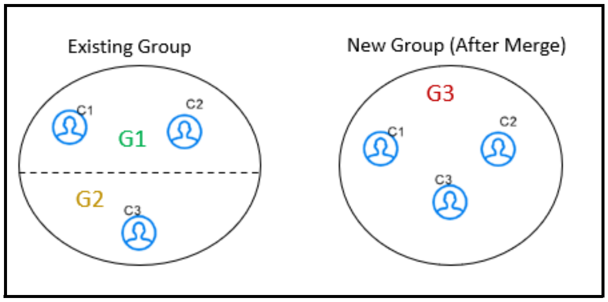
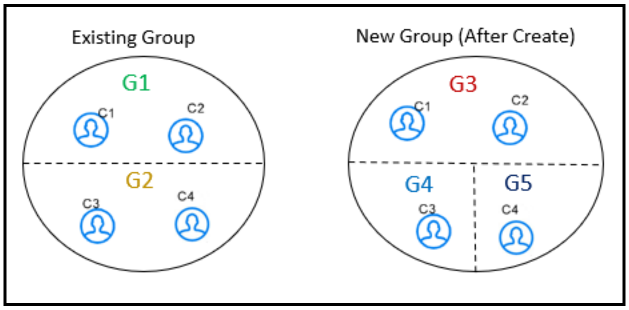
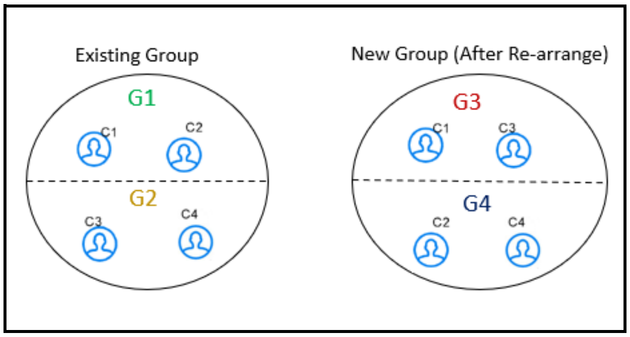
Value	Description
merge	You can merge the different entities into a single group with a new global id is assigned. **Figure 28: Merge**  Existing Group New Group (After Merge) G1 G2 G3 C1 C2 C3 For example, G1 has C1 and C2 entities, G2 has C3 entities. After merge, G3 has C1, C2, and C3 entities with a new global id. The V_ACTION column for G3 will merge and G1 will be deactivated.
create	You can create a new entity from the existing group with a new global id is assigned. **Figure 29: Create**  Existing Group New Group (After Create) G1 G2 G3 G4 G5 C1 C2 C3 C4 For example, G1 has C1 and C2 entities, G2 has C3 and C4 entities. After create, G3 has C1 and C2 entities, G4 has C3 entity and G5 has C4 entity with new global ids assigned to each group. The V_ACTION column for G3, G4 and G5 will create and G1 will be deactivated.

Table 16: V_ACTION Details

Value	Description
re-arrange	You can re-arrange the entities from another group with a new global id is assigned. **Figure 30: Re-arrange**  Existing Group G1 C1 C2 G2 C3 C4 New Group (After Re-arrange) G3 C1 C3 G4 C2 C4 For example, G1 has C1 and C2 entities, G2 has C3 and C4 entities. After re-arrange, G3 has C1 and C3 entities and G4 has C2 and C4 entities with new global ids assigned to each group. The V_ACTION column for G3 and G4 will re-arrange and G1 and G2 will be deactivated.

- **FCC_ER_OUTPUT:** It is a subset of all staging tables and stores specific column details from each staging output table.

The following are output tables for FSDF 811:

- STG_PARTY_MASTER
- STG_PARTY_DETAILS
- STG_PARTY_EMAIL_MAP
- STG_ADDRESS_MASTER
- STG_PARTY_ADDRESS_MAP
- STG_PARTY_PHONE_MAP
- STG_CUSTOMER_IDENTIFCTN_DOC
- FCC_ER_MAPPING
- FCC_ER_OUTPUT

The following tables are output tables for FSDF 812:

- STG_PARTY_MASTER
- STG_PARTY_DETAILS
- STG_PARTY_EMAIL_MAP
- STG_ADDRESS_MASTER
- STG_PARTY_ADDRESS_MAP
- STG_PARTY_PHONE_MAP
- STG_CUSTOMER_IDENTIFCTN_DOC

- FCC_ER_MAPPING
- FCC_ER_OUTPUT

6.4 Executing the ER Jobs

You can execute the following available jobs for Entity Resolution in a specified sequence:

1. Create Index and Load the Data (`ER_Create_And_Load_Data_Into_Index.sh`)
2. Perform Matching (`ER_Run_Bulk_Similarity_Job.sh`)
3. Data Survival (`ER_Run_Data_Survival_Engine.sh`)
4. Load Data in FCC_ER_OUTPUT Table (`ER_Run_Full_Data_Output.sh`)

> **NOTE** You can proceed with data movement from staging to FCDM during **Load Data in FCC_ER_OUTPUT Table** execution.

Before running the ER jobs, the user should ensure the following:

- Create an ER Schema
- Grant Permission to ER Schema
- Add ER Schema Wallet details
- Update `resource.xml` with ER Schema details

See the **Entity Resolution** section in the OFS Compliance Studio Installation Guide.

After installation, the user can follow the same steps in Configure the `resources.xml` for Multiple ER Schemas in OFS Compliance Studio Installation Guide to create additional ER schemas.

> **NOTE** You can use only one ER schema per **pipelineid** for each FSDF version, and the same ER schema cannot be used with other **pipelineid** for any ER job execution.

6.4.1 Create Index and Load the Data

> **NOTE** Ensure you have configured the **Logstash** parameter as **true** (`index.logstash-conf.apply`) in load-to-elastic-search `application.properties` to load the data from Database.

6.4.1.1 Job

`ER_Create_And_Load_Data_Into_Index.sh` performs the following:

- It creates all the output tables required at the different stages of Entity resolution tasks.
  - Input to this job will be pipeline id as an argument so that all the tables related to that pipeline ID will be created.
  - Index view table, Matching output table, Manual matches output table, Merge Map output table, Manual map merge output table, final dataset output tables. This task will create all these tables.
- It creates the index for the given Dataset and loads the data into the index table based on values provided in the **index.pipeline-id** argument.

6.4.1.2 Steps

1. Navigate to `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/bin`
2. Run the following command:

   ```
   ./ER_Create_And_Load_Data_Into_Index.sh "<pipelineid>" "<ERSchemaId>"
   "<Load Type>" "<FIC_MIS_DATE>" "<FSDFVersion>" "<Current_Batch>"
   "Source_Batch" "<Data_Origin>"
   ```

   For example, you can use the following commands for the different versions

   FSDF 808 version: `./ER_Create_And_Load_Data_Into_Index.sh "CSA_808" "ER1" "DeltaLoad"  "20220101" "808" "MAN" "MAN" "US"`

   ```
   FSDF 811 version: ./ER_Create_And_Load_Data_Into_Index.sh "CSA_811"
   "ER2" "FullLoad"  "20220101" "811" "MAN" "MAN" "US"
   ```

   ```
   FSDF 812 version: ./ER_Create_And_Load_Data_Into_Index.sh "CSA_812"
   "ER3" "DeltaLoad"  "20220101" "812" "MAN" "MAN" "US"
   ```

   The parameters are as follows:

   - **pipelineid**: ER Type has taken as Pipelined ID to execute. For example, CSA_808, CSA_812, CSA_811.
   - **ERSchemaId**: ER Schema Id is defined in the `resource.xml` file. For which schema that you want to run the Batch. This must be the same as specified in the `resource.xml` file.
   - **Load Type**: Load type. It can be either FullLoad or DeltaLoad.
     - **FullLoad**: Clear all the records from the history tables and match all the records based on the **fic_mis_date**.
     - **DeltaLoad**: Match the modified and new records with the current **fic_mis_date** against all the historical records.
   - **FIC_MIS_DATE**: The date on which the data is entered/loaded in the system in YYYYMMDD format.
   - **FSDFVersion**: The version of the staging tables.
   - **Current_Batch**: The processing group for which batch needs to be run. (Only one batch can run at a time for a processing group.)
   - **Source_Batch**: Future parameter. You can use the same value as the current batch for now.
   - **Data_Origin**: Origin of data.

6.4.1.3 Additional Configurations

To enhance the efficiency of history maintenance and delta processing, perform the following:

> **ATTENTION** The default values are based on hardware configurations (**Eight-core CPU** and **64 GB RAM**) and delta size (**ten million** records).

1. Log in to ER Schema.
2. Navigate to the **FCC_ER_CONFIG** table and configure the **V_PARAM_VALUE** value based on the DB performance.

   You can modify the following parameters in the table with **Pipeline_ID** as **CSA_812** before executing the job based on your volume of data:

- **PREV_CHUNKS**: The number of chunks of history tables during the last execution of the job. By default, it is set to **10**. You should not modify the value. This parameter value will be modified automatically when you modify the **TODAY_CHUNKS** value and execute the job successfully.
- **TODAY_CHUNKS**: The number of chunks of history tables for the current day/date. By default, it is set to **10**. You can modify this value to change the number of chunks to be processed in the respective history tables when the job execution time is longer.

NOTE	Here the chunk value is based on the volume of data being processed. It is recommended to increase the value to **15** when the volume of data being processed is more than **50** million records and monitor the performance.

- **MAX_JOBS**: Maximum number of jobs to schedule in the Database at a time. By default, it is set to **35**. You can modify this value to reduce job execution time.

NOTE	Increasing this value only when the Database is not shared for the other processes is recommended.

- **CHUNK_SIZE**: The number of records to process in one chunk. It is set to **2000000** (2 million records in each chunk) by default.

NOTE	It is recommended to retain the default value. You can decrease it to a lower value for better performance only when the server (where the Database is installed) has less than **eight** CPUs.

3. Save the changes.

6.4.1.3.1 Profiler Table

The table, **ER_PERFORMANCE_TIME_PROFILER** in ER schema, helps the user track the current status of the batch and debug performance issues.

The **ER_PERFORMANCE_TIME_PROFILER** table stores the queries that are executed during delta processing. Here are a few parameters that help to debug which query is failed:

- **V_TABLE_NAME**: It stores the table name for which the query was executed.
- **N_CHUNK**: It stores the chunk number that is executed.
- **D_STARTTIME**: It stores Database time when the query starts to execute.
- **D_ENDTIME**: It stores the Database time when the query got executed.
- **V_TOTAL_TIME**: It stores the duration of the query execution.
- **V_STATUS**: Current status of the query. The values are **START**, **RUNNING**, or **END**.
- **V_QUERY**: It stores the query that was executed.
- **N_RUN_SKEY**: It stores the **runSKey** value of the currently executing job.

To check the query status, perform the following:

1. Log in to ER Schema.
2. Run the following command:

```
SELECT * FROM ER_PERFORMANCE_TIME_PROFILER WHERE N_RUN_SKEY =
<CURRENT_RUNSKEY>
```

For example,

```
SELECT * FROM ER_PERFORMANCE_TIME_PROFILER WHERE N_RUN_SKEY = 200
```

3. Check **V_STATUS**. The status other than the **END** value indicates the failed query.

6.4.1.3.2 Cleanup Steps for Job Termination

Perform the clean-up steps when you manually terminate the **Create Index and Load Data** job. See Clean-up Steps When the Create Index and Load Data Job Terminated Manually.

6.4.2 Perform Matching

6.4.2.1 Job

The **`ER_Run_Bulk_Similarity_Job.sh`** triggers the matching engine to generate the matches in the match output table for rulesets saved against a pipeline-id argument for fetching rulesets.

6.4.2.2 Steps

> **NOTE** To re-run this job after a failure, the value of the **n_run_status** column in the **fcc_batch_run** table in Compliance Studio Schema should be changed to **2** for the respective **n_run_skey**.

1. Navigate to `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/bin`
2. Run the following command:

```
./ER_Run_Bulk_Similarity_Job.sh "<pipelineid>" "<ERSchemaId>""<Match
Type>" "<Current Batch>"
```

For example, you can use the following commands for the different versions

FSDF 808 version: `./ER_Run_Bulk_Similarity_Job.sh "CSA_808" “ER1” “delta” “MAN”`

FSDF 811 version: `./ER_Run_Bulk_Similarity_Job.sh "CSA_811" “ER2” “full” “MAN”`

FSDF 812 version: `./ER_Run_Bulk_Similarity_Job.sh "CSA_812" “ER3” “delta” “MAN”`

The parameters are as follows:

- **pipelineid**: ER Type has taken as Pipelined ID to execute. For example, CSA_808, CSA_812, CSA_811.
- **ERSchemaId**: ER Schema Id is defined in the `resource.xml` file. For which schema do you need to run the batch. This must be the same as specified in the `resource.xml` file.
- **Match Type**: It processes the records based on the dataset, either Full Load or Delta Load.

- **Current_Batch**: The processing group for which batch needs to be run. (Only one batch can run at a time for a processing group.)

> **NOTE** If the Bulk Similarity Edge job fails internally due to Incorrect schema details and then returns a success message. You can check the log file in `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/logs` for more details on the failure.

6.4.2.3 Matching Output

The results of the ER matching are stored in the following tables:

- **FCC_ER_MATCHING**: The results that exceed the automatic threshold limit are stored.
- **FCC_ER_MANUAL_MATCH**: The results between the automatic and manual thresholds are stored.

You can see the following details for the above tables:

- **SCORE**: Score of the match between Source and Target Entity
- **MATCH_DESCRIPTION:** Describes the attributes responsible for matching
- **SRC_DESC**: Describes attributes of Source considered for matching
- **TRG_DESC**: Describes attributes of Target considered for matching
- **V_PIPELINE_ID**: Describes the Pipeline Id of ER Type
- **N_RULESET_ID**: Describes the Ruleset responsible for matching
- **SRC_ORIGINAL_ID**: Describes the unique identifier for the Source entity
- **TRG_ORIGINAL_ID**: Describes the unique identifier for the Target entity

6.4.2.3.1 Cleanup Steps for Job Termination

Perform the clean-up steps when you manually terminate the **Bulk Similarity** job. See Clean-up Steps When the Bulk Similarity Job Terminated Manually section.

6.4.3 Data Survival

> **NOTE**
>
> - For executing the ER job for FSDF 808 version, disable the data survival Preconfigured rules for CSA_808 from the UI, and create a new rule to run the following batches.
> - Ensure only one preconfigured ruleset is enabled for Merging and Data Survival. See the Preconfigured Rulesets for Matching, Merging, and Data Survival section**.** The job will be failed with a unique constraint error if multiple rulesets are enabled.
> - If there is a unique constraint error in the **STG_CUSTOMER_IDENTIFCTN_DOC** table during the Data survival job, you should ignore the below error.

NOTE

```
2022-11-04 11:47:56,560 - global-
party.util.GlobalPartyUtils - 238 [ERROR]: Error
ORA-00001: unique constraint (ER10_0805_PERF.XPK-
STAGE_CUSTOMER_IDENTIFICATION_DOCUMENT_2) vio-
lated at row offset 10135

NoneType: None

2022-11-04 11:47:56,560 - global-
party.util.GlobalPartyUtils - 238 [ERROR]: Error
ORA-00001: unique constraint (ER10_0805_PERF.XPK-
STAGE_CUSTOMER_IDENTIFICATION_DOCUMENT_2) vio-
lated at row offset 10143

NoneType: None

2022-11-04 11:47:56,560 - global-
party.util.GlobalPartyUtils - 238 [ERROR]: Error
ORA-00001: unique constraint (ER10_0805_PERF.XPK-
STAGE_CUSTOMER_IDENTIFICATION_DOCUMENT_2) vio-
lated at row offset 10145

NoneType: None

2022-11-04 11:47:56,561 - global-
party.util.GlobalPartyUtils - 238 [ERROR]: Error
ORA-00001: unique constraint (ER10_0805_PERF.XPK-
STAGE_CUSTOMER_IDENTIFICATION_DOCUMENT_2) vio-
lated at row offset 10151

NoneType: None

2022-11-04 11:47:56,561 - global-
party.util.GlobalPartyUtils - 238 [ERROR]: Error
ORA-00001: unique constraint (ER10_0805_PERF.XPK-
STAGE_CUSTOMER_IDENTIFICATION_DOCUMENT_2) vio-
lated at row offset 10170

NoneType: None

2022-11-04 11:47:56,561 - global-
party.util.GlobalPartyUtils - 238 [ERROR]: Error
ORA-00001: unique constraint (ER10_0805_PERF.XPK-
STAGE_CUSTOMER_IDENTIFICATION_DOCUMENT_2) vio-
lated at row offset 10183
```

6.4.3.1 Job

The `ER_Run_Data_Survival_Engine.sh` job performs the following:

- **ER_Merge_Engine**: It triggers the merge engine, and records will be inserted in the mapping table based on the merge rules saved against the pipeline id argument.
- **ER_Data_Survival_Engine**: It triggers the data survival engine, and final outputs will be stored in tables based on the dataset survival rule stored against pipeline id.

6.4.3.2 Steps

NOTE To re-run this job after a failure, the value of the **n_run_status** column in the **fcc_batch_run** table in Compliance Studio Schema should be changed to **4** for the respective **n_run_skey**.

1. Navigate to `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/bin`
2. Run the following command:

```
./ER_Run_Data_Survival_Engine.sh "<pipelineid>" "<ERSchemaId>"
"<ERSchemaName>" "<Current Batch>" "<Match Type>" "<FIC_MIS_DATE>"
```

For example, you can use the following commands for the different versions

FSDF 808 version: `./ER_Run_Data_Survival_Engine.sh "CSA_808" “ER1” “ER1ABC” “MAN” "delta" "20151210"`

FSDF 811 version : `./ER_Run_Data_Survival_Engine.sh "CSA_811" “ER2” “ER2EFG” “MAN” "delta" "20151210"`

FSDF 812 version : `./ER_Run_Data_Survival_Engine.sh "CSA_812" “ER3” “ER3RST” “MAN” "delta" "20151210"`

The parameters are as follows:

- **pipelineid**: ER Type has taken as Pipelined ID to execute. For example, CSA_808, CSA_812, CSA_811.
- **ERSchemaId**: ER Schema Id is defined in the `resource.xml` file. For which schema do you need to run the batch. This must be the same as specified in the `resource.xml` file.
- **ERSchemaName**: Entity Resolution schema alias name.
- **Current_Batch**: The processing group for which batch needs to be run. (Only one batch can run at a time for a processing group.)
- **Match Type**: It processes the records based on the Full Load or Delta Load dataset.
- **FIC_MIS_DATE**: The date on which the data is entered/loaded in the system in `YYYYMMDD` format.

NOTE

- The user should not have **Type** “Distinct” and “All**”** together with other columns that return unique values in child tables.
- If the Batch, Backup, and Recovery processes fail when you execute the `ER_Run_Data_Survival_Engine.sh`, you need to re-run the same job again to ensure the Data is available in Archive only for the Mapping table (**FCC_ER_MAPPING**).
- To increase/decrease the execution efficiency according to the server size using **ER_THREADS** and **ER_BATCH_SIZE** parameters, perform the following:

1. Navigate to `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/bin`

NOTE

2. Open the `ER_Run_Data_Survival_Engine.sh` and set the following parameters:
   - `export ER_THREADS=<No of threads>`
   - `export ER_BATCH_SIZE=<Batch Size>`

   Example:
   - `export ER_THREADS=4`
   - `export ER_BATCH_SIZE=10000`

3. Validate to ensure Global party IDs are generated for the Entities in the following Staging Output tables after executing the job:
   - `STG_PARTY_MASTER`
   - `STG_PARTY_DETAILS`
   - `STG_PARTY_EMAIL_MAP`
   - `STG_PARTY_PHONE_MAP`
   - `STG_ADDRESS_MASTER`
   - `STG_PARTY_ADDRESS_MAP`
   - `STG_CUSTOMER_IDENTIFCTN_DOC`

NOTE Data Survival process expects the above STG tables to retain the snapshot of the previous **FIC_MIS_DATE** to complete the process successfully.

6.4.3.2.1 Cleanup Steps for Job termination

Perform the clean-up steps when you manually terminate the **Data Survival** job. See Clean-up Steps When the Data Survival Job Terminated Manually section.

6.4.4 Load Data in FCC_ER_OUTPUT Table

6.4.4.1 Job

The **`ER_Run_Full_Data_Output.sh`** job executes the SQL procedure that joins the following staging output tables and populates data for the split and merge UI:

- `STG_PARTY_MASTER`
- `STG_PARTY_DETAILS`
- `STG_PARTY_EMAIL_MAP`
- `STG_PARTY_PHONE_MAP`
- `STG_ADDRESS_MASTER`
- `STG_PARTY_ADDRESS_MAP`

- STG_CUSTOMER_IDENTIFCTN_DOC

> **NOTE** If you want to perform slicing for the initial input data to run **Day 0** batch, it is recommended to run `ER_Create_And_Load_Data_Into_Index.sh`, `ER_Run_Bulk_Similarity_Job.sh`, and `ER_Run_Data_Survival_Engine.sh` jobs for all slices. The Output Tables are expected to have the resolved entities at the end of this process. At this point, `ER_Run_Full_Data_Output.sh` job can be executed for bringing the entire data across all slices into the output table.

6.4.4.2 Steps

> **NOTE** To re-run this job after a failure, the value of the **n_run_status** column in the **fcc_batch_run** table in Compliance Studio Schema should be changed to **6** for the respective **n_run_skey**.

1. Navigate to `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/bin`
2. Run the following command:

```
./ER_Run_Full_Data_Output.sh "<pipelineid>" "<ERSchemaId>"
"<FIC_MIS_DATE>" "<Current Batch>"
```

For example, you can use the following commands for the different versions

FSDF 808 version: `./ER_Run_Full_Data_Output.sh "CSA_808" "ER1" "20151218" "MAN"`

FSDF 811 version: `./ER_Run_Full_Data_Output.sh "CSA_811" "ER2" "20151218" "MAN"`

FSDF812version: `./ER_Run_Full_Data_Output.sh"CSA_812""ER3 " "20151218" "MAN"`

The parameters are as follows:

- **pipelineid**: ER Type has taken as Pipelined ID to execute. For example, CSA_808, CSA_812, CSA_811.
- **ERSchemaId**: ER Schema Id is defined in the `resource.xml` file. For which schema do you need to run the batch. This must be the same as specified in the `resource.xml` file.
- **FIC_MIS_DATE**: The date on which the data is entered/loaded in the system.
- **Current_Batch**: The processing group for which batch needs to be run. (Only one batch can run at a time for a processing group.)

3. Validate specific column details are loaded in **FCC_ ER_OUTPUT** table from each staging output table for the Entities after executing the job.

6.4.4.2.1 Cleanup Steps for Job termination

Perform the clean-up steps when you manually terminate the **Load Data in the FCC_ER_OUTPUT** job. See Clean-up Steps When the Load Data in FCC_ER_OUTPUT Job Terminated Manually section.

6.4.5 Initial Run for High Volume Data

The initial run (Day 0) of Entity Resolution on a high volume of data is expected to take a longer time and more resources based on the performance. For an efficient initial run (Day 0), you can run the utility scrip to a faster turn-around time for individual batches as the load is moderately low. See Data Slicing Utility Script for more details.

6.4.6 Status Codes

The **fcc_batch_run** table in Compliance Studio Schema explains the status codes generated for ER jobs. See the status codes in **n_run_status** column for respective **n_run_skey** values.

Table 17 lists the ER job status codes:

Table 17: ER Job Status Codes

ER Job Name	During Execution	Success	Failure
ER_Create_And_Load_Data_Into_Index.sh	1	2	11
ER_Run_Bulk_Similarity_Job.sh	3	4	12
ER_Run_Data_Survival_Engine.sh	5	6	13
ER_Run_Full_Data_Output.sh	7	8	14

6.5 Persisting the Data

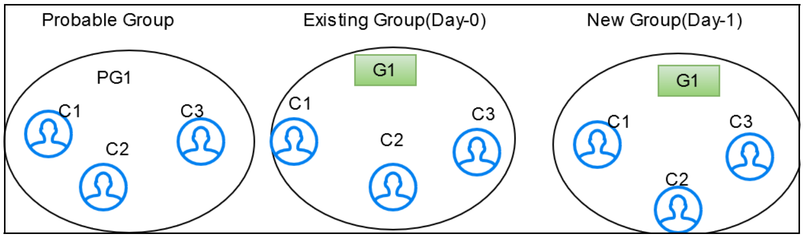
Probable groups are created for entities that match. Merge rules are applied to all entities within a probable group to define which entities should be grouped into a global party. Day-on-day changes to the underlying party records may impact the global party group of which they are apart. The following sections show where the match or merge changes may impact a global party and when the global party would be deactivated and new global parties would be created. This can occur when matching criteria change or when groups and manually linked or de-linked.

Note that a change in a non-matching attribute will not change the global party group but may change attributes on the global party record if it impacts the data survival mechanism.

6.5.1 No change

Existing group elements are a subset of probable group elements, and the number of elements is the same in both groups. All elements in the existing Group have the same global id. The existing global id is assigned to probable group elements.

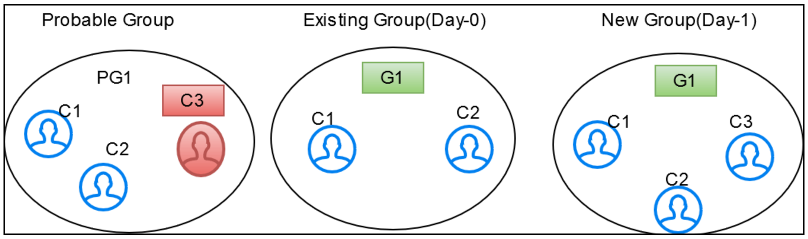
Figure 31: No change



6.5.2 Add

Existing group elements are a subset of probable group elements, and the number of elements in the probable Group is more than the existing Group. Extra elements in the probable Group don't have any global id assigned yet. New elements are added to the existing Group, and the same global id is assigned.

Figure 32: Add

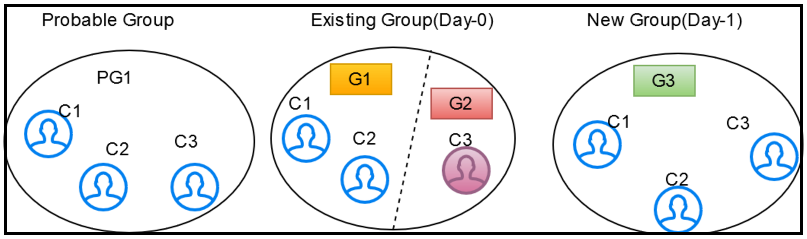


6.5.3 Merge

Existing group elements are a subset of probable group elements, and the number of elements is the same in both groups. Elements in the existing Group have different global ids assigned.

Elements are merged into a single group, and a new global id is assigned.

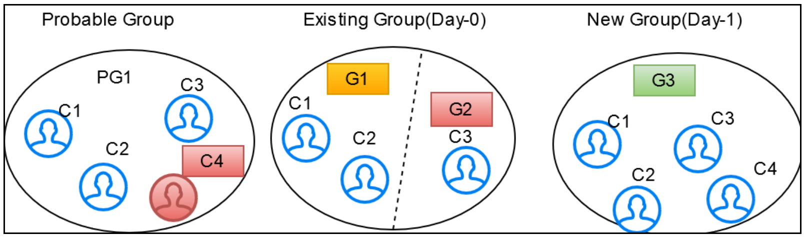
Figure 33: Merge



6.5.4 Merge and Add

Existing group elements are a subset of probable group elements, and the number of elements in the probable Group is more than the existing Group. Extra elements in the probable Group don't have any global id assigned yet, and standard elements have different global IDs assigned already. Common elements are merged into a single group, and new elements are added to the Group with a new global id.

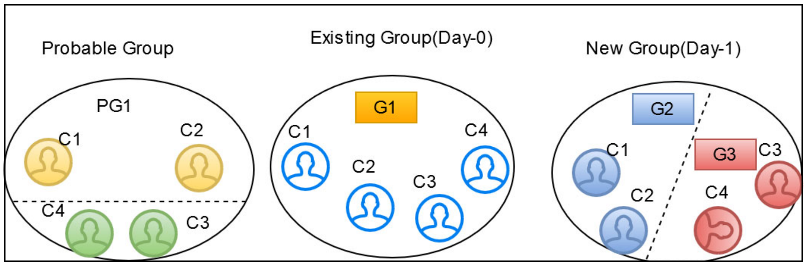
Figure 34: Merge and Add



6.5.5 Split

After applying merging rules criteria, if multiple groups are created for elements of a probable group, these elements are also a subset of existing group elements. The number of elements in both probable and existing groups is the same. A single global id is assigned to all elements in the existing Group, and then probable group elements are split-ted into different groups with new global ids assigned to each.

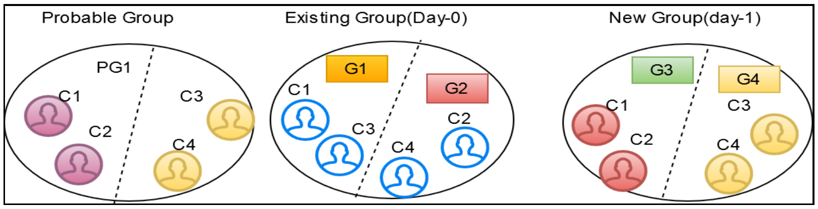
Figure 35: Split



6.5.6 Split and Merge

After applying merging rules criteria, if multiple groups are created for elements of a probable group, these elements are also a subset of existing group elements. The number of elements in both probable and existing groups is the same, and different global ids are assigned to elements in the existing Group, then probable group elements are split into different groups and merged, satisfying the same ruleset criteria with new global ids assigned to each.

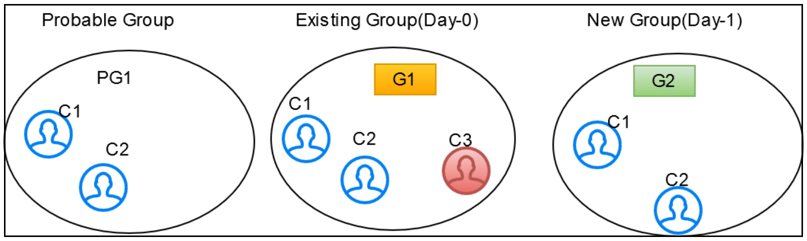
Figure 36: Split and Merge



6.5.7 Delete

If an element exists in the existing Group, but the same element doesn't belong to any probable group and doesn't exist in the customer/entity dataset, it is deleted from the existing Group, and a new global id is assigned to the Group.

Figure 37: Delete



6.6 Entity Resolution Metadata

Metadata tables manage the operation of the Entity Resolution jobs.

6.6.1 Default Data in the tables

The following are the complete set of tables that are used for the ER:

- **The following tables store the table structure definition for Party Master**:
  - **FCC_M_ER_TABLES**: This table contains information about different tables required by the product as part of an Entity Resolution process. The values in the column V_FSDF_VERSION differentiate FSDF versions to the tables belong to. This is used for creating Datasets and Data Surviving Rules.
  - **FCC_M_ER_TABLES_TL**: This table contains translative information for FCC_M_ER_TABLES, with multiple translations based on the Locale column.
  - **FCC_M_ER_COLUMNS**: This table contains information about columns a table has. It has mappings of columns and tables so that you can get the table's available columns information based on table Id. This is used for creating Datasets and Data Surviving Rules.
  - **FCC_M_ER_ATTRIBUTE**: This table contains information about columns. It has a column's information such as logical name and description. This is used for creating Datasets and Data Surviving Rules.
  - **FCC_M_ER_ATTRIBUTE_COLUMN_MAP**: This table contains mapping information of attributes and columns. It also stores information about the relationship between tables. This is used for creating Datasets and Data Surviving Rules.
  - **FCC_M_ER_ATTRIBUTE_TL**: This table contains translative information for table FCC_M_ER_ATTRIBUTE, which can have multiple translation information based on the Locale column.
- **The following tables store the Dataset definition for FSDF 808** and **812**:
  - **FCC_M_ER_DATASET**: This table contains information about Datasets. It has a master (parent) table information like STG_PARTY_MASTER_PRE (when resolving FSDF data), output table, and pipeline Id, and tables where the data will flow when the data survival job is run.
  - **FCC_M_ER_DATASET_GROUP**: This table contains information about a Group of other tables that are part input dataset. It has an input group table like STG_PARTY_ADDRESS_PRE and also stores the join condition with the Master table, STG_PARTY_MASTER_PRE.
  - **FCC_M_ER_DATASET_MAP**: This table contains information about the mapping table, which provides the relationship between the Master and Group tables. For example, STG_PARTY_ADDRESS_MAP_PRE stores the relationship between the STG_PARTY_MASTER_PRE and STG_PARTY_ADDRESS_PRE tables.
  - **FCC_M_ER_DATASET_TL**: This table contains translative information for table **FCC_M_ER_DATASET**, which can have multiple translations based on the Locale column.
- **The following tables store the Preconfigured Match and Merge Ruleset FSDF 808** and **812**:
  - **FCC_MATCH_RULESET**: This table contains the information of the Rulesets created in Matching Rules UI. It gives information like the Pipeline ID, Ruleset Name, and Ruleset Description and contains ruleset details in JSON format.

- **FCC_MERGE_RULESET**: This table contains the information of the Rulesets created in Merge Rules UI. It gives information like the Pipeline ID, Ruleset Name, and Ruleset Description and contains ruleset details in JSON format.

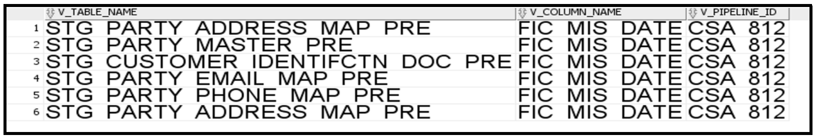
- **The following tables store the Dataset Survival Rule for FSDF 808** and **812**:
  - **FCC_DATASURV_RULES**: This table contains the information on the Rules created in Data Survival Rules UI. It gives information like the Pipeline ID, Ruleset Name, and Ruleset Description and contains ruleset details in JSON format. This table contains information only for the Master table.
  - **FCC_DATASURV_GROUPS**: This table contains data survival rules, such as rule id, UI JSON, and query JSON. UI JSON is used on the UI side, and query JSON is used as input JSON for the Data survival engine. This table contains information only for child tables.
  - **FCC_DATASURV_TYPE**: This table contains information about different Data Survival Algorithms, such as Longest, Latest, Most Common, etc. There is a Type drop-down on Data Survival UI to choose values (fetched from this table) for a particular column.

  Data survival rules of out-of-the-box ER pipeline survive the “Latest” data based on FIC_MIS_DATE. Since data for ER is always considered as a complete snapshot for the extraction date (FIC_MIS_DATE), the FIC_MIS_DATE will be standard across the entire snapshot. Hence ER internally considers the additionally maintained D_LAST_UPDATED_DATE column in H$ tables to find out the latest data for survival. This is achieved by an additional set of metadata maintained in the following tables:

  - **FCC_M_ER_PROCESSING_COLUMNS**: This table stores the table name, column name, and ER pipeline id.
  - **FCC_DS_REF_COLUMN_MAPPING**: This table stores the table name, reference column name (the standard column of the table, i.e., FIC_MIS_DATE), target column name (the actual column on which “Latest” should be considered, i.e., D_LAST_UPDATED_DATE), and ER pipeline id.

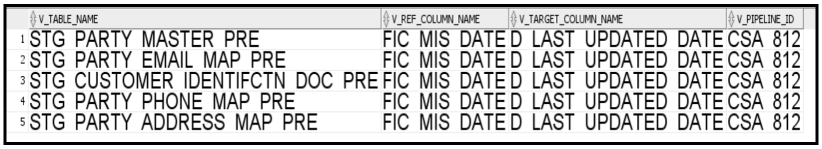
    For Example, the sample records for both tables are as follows:

Figure 38: Sample Record for FCC_M_ER_PROCESSING_COLUMNS



	V_TABLE_NAME	V_COLUMN_NAME	V_PIPELINE_ID
1	STG PARTY ADDRESS MAP PRE	FIC MIS DATE	CSA 812
2	STG PARTY MASTER PRE	FIC MIS DATE	CSA 812
3	STG CUSTOMER IDENTIFCTN DOC PRE	FIC MIS DATE	CSA 812
4	STG PARTY EMAIL MAP PRE	FIC MIS DATE	CSA 812
5	STG PARTY PHONE MAP PRE	FIC MIS DATE	CSA 812
6	STG PARTY ADDRESS MAP PRE	FIC MIS DATE	CSA 812

Figure 39: Sample Record for FCC_DS_REF_COLUMN_MAPPING



	V_TABLE_NAME	V_REF_COLUMN_NAME	V_TARGET_COLUMN_NAME	V_PIPELINE_ID
1	STG PARTY MASTER PRE	FIC MIS DATE	D LAST UPDATED DATE	CSA 812
2	STG PARTY EMAIL MAP PRE	FIC MIS DATE	D LAST UPDATED DATE	CSA 812
3	STG CUSTOMER IDENTIFCTN DOC PRE	FIC MIS DATE	D LAST UPDATED DATE	CSA 812
4	STG PARTY PHONE MAP PRE	FIC MIS DATE	D LAST UPDATED DATE	CSA 812
5	STG PARTY ADDRESS MAP PRE	FIC MIS DATE	D LAST UPDATED DATE	CSA 812

NOTE These metadata tables should be seeded with appropriate values in any similar use cases.

- **The following table stores the flattening data query for FSDF 808** and **812**:
  - **FCC_STUDIO_ER_QUERIES**: This table contains queries to fattening data from input tables for each pipeline id. The information in this table can be amended via an API if additional attributes need to be brought into matching.
- **The following tables to populate fields in Match and Merge Ruleset UI for FSDF 808** and **812**:
  - **FCC_ER_INDEX**: This table contains the index name on the ruleset UI screen in Source Index Name and Target Index Name Field.
  - **FCC_IDX_M_JSON_MAP**: This table contains the mapping of each index populated on elastic search, making the initial candidate selection for records to be scored by the matching engine. This is required for Match and Merge Rulesets mapping screen. You need to add custom attributes for mapping manually. For more information on how to map, see the Steps section.
  - **FCC_ER_ATTRIBUTES**: This table contains attributes matched in ruleset UI in source and target attribute for the respective index.
  - **FCC_IDX_M_LOOKUP**: This table contains the file name/index name of synonyms and Stopwords, which are used to improve the performance of Name/Address matching.
  - **FCC_IDX_M_LOOKUP_VALUES**: This table contains populated values for the above index names.
  - **FCC_ER_M_BKP_CONFIG**: This table contains the backup and failure recovery details.

6.6.2 Customize the Data in the Tables for ER types

Entity Resolution can be adapted for additional use cases by configuring the data in the metadata tables.

6.6.2.1 List of tables

- FCC_M_ER_DATASET
- FCC_M_ER_DATASET_GROUP
- FCC_M_ER_DATASET_MAP
- FCC_M_ER_DATASET_TL
- FCC_STUDIO_ER_QUERIES
- FCC_ER_INDEX
- FCC_IDX_M_JSON_MAP
- FCC_ER_ATTRIBUTES

6.6.2.2 Steps

Perform the following steps to customize the data using API:

1. Get the Datasets that exist in the system:
   a. Configure the hostname.
   b. Run the following command:

```
curl --location --request GET 'http://<HOSTNAME>:7051/datasurvival/
getDataSet' \
--header 'Content-Type: application/json'
```

```
For example,
```

```
curl --location --request GET 'http:// hostname.com:7051/datasurvival/
getDataSet' \
```

```
--header 'Content-Type: application/json'
```

> **NOTE** To modify the Dataset, you can provide the existing value for `datasetName` to edit the JSON file and modify the other parameters except for `datasetName` in the same file according to the requirement.

2. Enter the details of the Dataset in the Request JSON.
   a. Configure the hostname.
   b. Run the following command:

```
curl --location --request POST 'http://<HOSTNAME>:7051/datasurvival/
createdataset' \

--header 'Content-Type: application/json' \

--data-raw '{

    "fcc_m_er_dataset": {

        "tableId": "",

        "datasetName": "",

        "mapTable": "",

        "matchTable": "",

        "manualMatchTable": "",

        "manualMapTable": "",

        "viewDataset": "",

        "outputTable": "",

"pipelineId":"",

        "statusFl": "",

        "productPartFl": "",

        "code": ""

    },

    "fcc_m_er_dataset_tl": {

        "tlTableId": "",

        "locale": "en-US",

        "tlDdatasetName": "Customer811"

    },

    "fcc_m_er_dataset_group": [

        {
            "groupTableId": "",
            "mapTableId": "",
            "groupMapTableJoin": "",
            "outputTable": "",
            "statusFl": "",
            "productPartFl": "",
            "code": "",
"isParent":"Y"
        },
        {
            "groupTableId": "",
            "mapTableId": "",
            "groupMapTableJoin": "",
            "outputTable": "",
            "statusFl": "",
            "productPartFl": "",
            "code": "",
            "isParent":""
        },
        {
            "groupTableId": "",
            "mapTableId": "",
            "groupMapTableJoin": "",
            "outputTable": "",
            "statusFl": "",
            "productPartFl": "",
            "code": "",
"isParent":""
        },
        {
            "groupTableId": "",
            "mapTableId": "",
            "groupMapTableJoin": "",
            "outputTable": "",
            "statusFl": "",
            "productPartFl": "",
            "code": "",
"isParent":""
        },
        {
            "groupTableId": "",
            "mapTableId": "",
            "groupMapTableJoin": "",
            "outputTable": "",
            "statusFl": "",
            "productPartFl": "",
            "code": "",
"isParent":""
        }
    ],
    "fcc_m_er_dataset_map": [
        {
            "mapTableId": "",
            "datasetMapTableJoin": "",
            "outputTable": "",
            "statusFl": "Y",
            "productPartFl": "Y",
            "code": ""
        }
    ]
}'
```

For example,

```
curl --location --request POST 'http:// hostname.com:7051/
datasurvival/createdataset' \
--header 'Content-Type: application/json' \
--data-raw '{
    "fcc_m_er_dataset": {
        "tableId": "220",
        "datasetName": "Customer811",
        "mapTable": "FCC_ER_MAPPING_811",
        "matchTable": "FCC_ER_MATCHING_811",
        "manualMatchTable": "FCC_ER_MANUAL_MATCH_811",
        "manualMapTable": "FCC_ER_MANUAL_MAP_811",
        "viewDataset": "FCC_ER_VIEW_811",
        "outputTable": "STG_PARTY_MASTER",
"pipelineId":"CSA811",
        "statusFl": "",
        "productPartFl": "",
        "code": ""
    },
    "fcc_m_er_dataset_tl": {
        "tlTableId": "220",
        "locale": "en-US",
        "tlDdatasetName": "Customer811"
    },
    "fcc_m_er_dataset_group": [
        {
            "groupTableId": "221",
            "mapTableId": "",
            "groupMapTableJoin": "STG_PARTY_MASTER_PRE.V_PARTY_ID = 
STG_PARTY_DETAILS_PRE.V_PARTY_ID",
            "outputTable": "STG_PARTY_DETAILS",
            "statusFl": "",
            "productPartFl": "",
            "code": "",
"isParent":"Y"
        },
        {
            "groupTableId": "226",
            "mapTableId": "",
            "groupMapTableJoin": "STG_PARTY_MASTER_PRE.V_PARTY_ID = 
STG_CUSTOMER_IDENTIFCTN_DOC_PRE.V_CUST_REF_CODE",
            "outputTable": "STG_CUSTOMER_IDENTIFCTN_DOC",
            "statusFl": "",
            "productPartFl": "",
            "code": "",
            "isParent":""
        },
        {
            "groupTableId": "223",
            "mapTableId": "224",
           "groupMapTableJoin": "STG_ADDRESS_MASTER_PRE.V_ADDRESS_ID
= STG_PARTY_ADDRESS_MAP_PRE.V_ADDRESS_ID",
            "outputTable": "STG_ADDRESS_MASTER",
            "statusFl": "",
            "productPartFl": "",
            "code": "",
"isParent":""
        },
        {
            "groupTableId": "225",
            "mapTableId": "",
            "groupMapTableJoin": "STG_PARTY_DETAILS_PRE.V_PARTY_ID =
STG_PARTY_PHONE_MAP_PRE.V_PARTY_ID",
            "outputTable": "STG_PARTY_PHONE_MAP",
            "statusFl": "",
            "productPartFl": "",
            "code": "",
"isParent":""
        },
        {
            "groupTableId": "222",
            "mapTableId": "",
            "groupMapTableJoin": "STG_PARTY_DETAILS_PRE.V_PARTY_ID =
STG_PARTY_EMAIL_MAP_PRE.V_PARTY_ID",
            "outputTable": "STG_PARTY_EMAIL_MAP",
            "statusFl": "",
            "productPartFl": "",
            "code": "",
"isParent":""
        }
    ],
    "fcc_m_er_dataset_map": [
        {
            "mapTableId": "224",
            "datasetMapTableJoin": "STG_PARTY_DETAILS_PRE.V_PARTY_ID =
STG_PARTY_ADDRESS_MAP_PRE.V_PARTY_ID",
             "outputTable": "STG_PARTY_ADDRESS_MAP",
             "statusFl": "Y",
             "productPartFl": "Y",
             "code": ""
        }
    ]
}'
```

3. Delete the existing Dataset:
   a. Configure the hostname.
   b. Run the following command:

```
curl --location --request POST 'http://<HOSTNAME>:7051/datasurvival/
deleteDataSet' \
--header 'Content-Type: application/json' \
--data-raw '{
"dataSetId":""
"datasetName":""
}'
For example,
curl --location --request POST 'http:// hostname.com:7051/
datasurvival/deleteDataSet' \
--header 'Content-Type: application/json' \
--data-raw '{
"dataSetId":"273"
"datasetName":"Customer811"
}'
```

4. Get Dataset Hierarchy for table relation summary:
   a. Configure the hostname.
   b. Run the following command:

```
curl --location --request POST 'http://<HOSTNAME>:7051/datasurvival/
getDataSetHierarchySummary' \
--header 'Content-Type: application/json' \
--data-raw '{
    "dataSetId": "",
"datasetName": ""
}'
For example,
curl --location --request POST 'http:// hostname.com:7051/
datasurvival/getDataSetHierarchySummary' \
--header 'Content-Type: application/json' \
--data-raw '{
    "dataSetId": "273",
"datasetName": "Customer811"
}'
```

5. Get Dataset Hierarchy Tables' Data:
    a. Configure the hostname.
    b. Run the following command:

    ```
    curl --location --request POST 'http://<HOSTNAME>:7051/datasurvival/
    getDataSetHierarchy' \
    --header 'Content-Type: application/json' \
    --data-raw '{
        "dataSetId": "",
    "datasetName": ""
    }'
    For example,
    curl --location --request POST 'http:// hostname.com:7051/
    datasurvival/getDataSetHierarchy' \
    --header 'Content-Type: application/json' \
    --data-raw '{
        "dataSetId": "273",
    "datasetName": "Customer811"
    }'
    ```

6. To change any field name in the Elastic Search Index for the ER type:
    a. Modify the value in the QUERY column in the **FCC_STUDIO_ER_QUERIES** to bring the field name in the ES Index.
    b. Add the QUERY column values to the **V_IDX_JSON** column in the **FCC_STUDIO_ER_QUERIES**

    NOTE: Ensure the value is the same in both columns, QUERY, and V_IDX_JSON.

7. To populate the source and target index on Ruleset UI:
    a. Add a new record in the table, FCC_ER_INDEX.
    b. Add source and target attributes on respective indexes in the table FCC_ER_ATTRIBUTES.

c. Create a new Ruleset for the customized ER type(s) in tables in the previous step. See the Creating Rulesets section in the OFS Compliance Studio User Guide for creating and configuring rulesets.

d. Execute the ER jobs with customized ER type(s). For more information on how to execute the jobs, see the Executing the ER Jobs section.

6.6.3 Populate the Metadata for Data Survival in Compliance Studio Schema

The **FCC_M_ER_ATTRIBUTE_PREC** table in Compliance Studio Schema stores information about the attribute column name, code of the attribute value, and the precedence value.

Table 18 structure with examples:

Table 18: Metadata

v_metadata_type	v_column_cd	n_precedence
Occupation	teacher	2
Geo-location	US	3

6.6.3.1 REST API to Load Metadata into Compliance Studio Schema

This is used to upload metadata and precedence and update the precedence for existing metadata types in the `FCC_M_ER_ATTRIBUTE_PREC` table.

URL: `http://<hostname>:7051/datasurvival/loadDataSurvMetadata`

Request Method: POST

Request Headers: Content-Type: application/json

Request body:

```
[{
        "vmetadataType": "Geo Risk",
        "vcolumnCd": "UK",
        "nprecedence": "6"
    },
    {
        "vmetadataType": "Geo Risk",
        "vcolumnCd": "US",
        "nprecedence": "5"
    },
  {
        "vmetadataType": "Geo Risk",
        "vcolumnCd": "FIN",
        "nprecedence": "3"
    }
]
```

6.6.3.2 REST API to Update Metadata Type

This is used to delete the existing set of metadata and update the metadata type and precedence with a new set of metadata.

URL: `http://<hostname>:7051/datasurvival/updateMetadataType`

Request Method: POST

Request Headers: Content-Type: application/json

Request body:

```
[{
        "vmetadataType": "Geo Risk",
        "vcolumnCd": "UK",
        "nprecedence": "6"
    },
    {
        "vmetadataType": "Geo Risk",
        "vcolumnCd": "US",
        "nprecedence": "5"
    }
]
```

6.6.3.3 REST API to Get Metadata Type and Precedence

This is used to get the records available in the precedence table.

URL: `http://<hostname>:7051/datasurvival/getAttributePrecMetadata`

Request Method: GET

Request Headers: Content-Type: application/json

6.6.3.4 REST API to Delete any Metadata Type

This is used to delete all records for a specific metadata type in the precedence table.

URL: `http://<hostname>:7051/datasurvival/deleteMetadataType?vMetadataType=<Metadata Type>`

For example,

```
http://testserver.oracle.com:7051/datasurvival/
deleteMetadataType?vMetadataType=Occupation
```

Request Method: POST

Request Headers: Content-Type: application/json

7 Configure ETL

Use this chapter to understand and perform configuration activities before running the extract, transform, and load (ETL) process. The ETL process loads business data from FCDM (BD and ECM), which can be used by any interpreter. FCDM and External data sources such as ICIJ are processed and loaded as a graph in Compliance Studio.

Topics:

7.1 Understand ETL

Use this section to understand how to generate a graph to move source data into the PGX server. The following sections provide more insight on data sources, jobs, rulesets, and workflows used in Compliance Studio to generate graphs.

7.1.1 Data Source

The Compliance Studio provides the following ready-to-use data sources:

- **FCDM**: The Financial Crime Data Model (FCDM) is the data source for Behavior Detection (BD) and Enterprise Case Management (ECM) Atomic schema tables.
- **ICIJ**: International Consortium of Investigative Journalists (ICIJ) is the data source for external entities like Panama Papers, Paradise Papers, etc. Where input data files are in the .CSV format, these files must be placed in Hadoop Distributed File System (HDFS).

7.1.2 Rulesets

Ruleset facilitates to identify the similarity between two entities within the data source, that is, FCDM to FCDM (customer to customer) or across the data source, that is, FCDM to ICIJ (FCDM entities, customer to customer, and customer to Panama papers) to derive a match and create similarity edges in the graph.

For creating and configuring rulesets, see the **Creating Rulesets** section in the OFS Compliance Studio User Guide.

7.1.3 Elasticsearch

Elasticsearch is a distributed search and analytical engine for all structured and unstructured data types. In Compliance Studio, Node tables are moved to Elasticsearch to get faster responses to identify the entity's relationships.

7.1.4 Indices

An index is a logical namespace that maps to one or more primary shards (a shard is a unit in which Elasticsearch distributes data around the cluster). As a part of the connector's job, the node entities from the graph's data source are populated as indices. The indices are used to query to generate similarity edges in the graph.

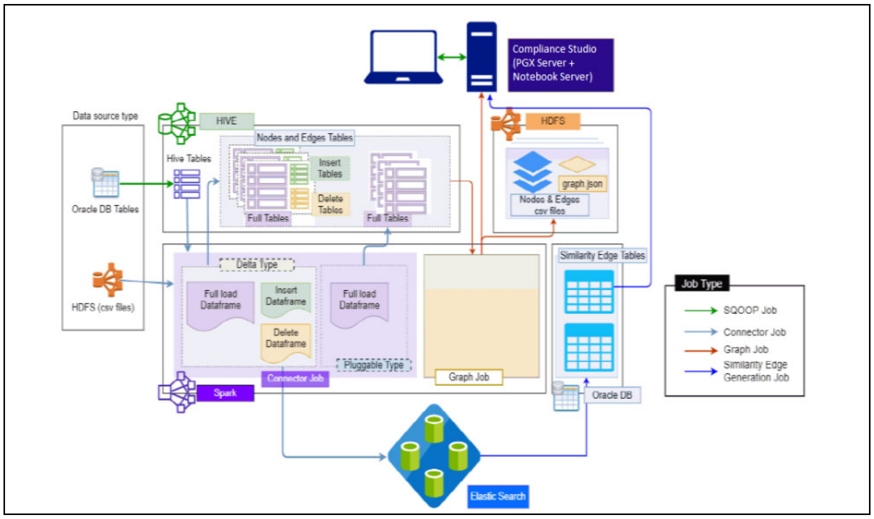
7.1.5 PGX

PGX is a toolkit for graph analysis, supporting both efficient graph algorithms and fast SQL-like graph pattern matching queries.

7.1.6 ETL and its Workflow

Compliance Studio's ETL is a process where business and external data can be processed to get a global graph, with entities resolved based on matching rules. This graph can be further queried for investigation.

Figure 40: ETL and its Workflow



ETL has two stages:

- Generation and maintenance of Graph
- Linking entities based on matching rules and updating

In the first stage, the business and external data are collected in HIVE Schema and then transformed based on the query and saved into HIVE tables. The Node type data are updated in elastic search for the second stage of ETL.

ETL classifies an entity into "delta type" or "pluggable type," Based on entity type, it updates the change, referred to as delta, into the graph. The ETL compares data from the previous batch for delta type and recognizes changes as an insert or delete.

Example: To understand the ETL delta, see the Example of ETL.

For pluggable type, transaction edges, the data are separated into multiple datasets based on a parameter like 'transaction date' and then are updated into a graph based on the valid transaction date range. ETL also removes older transactions and thus maintains graphs with the desired range of transactions.

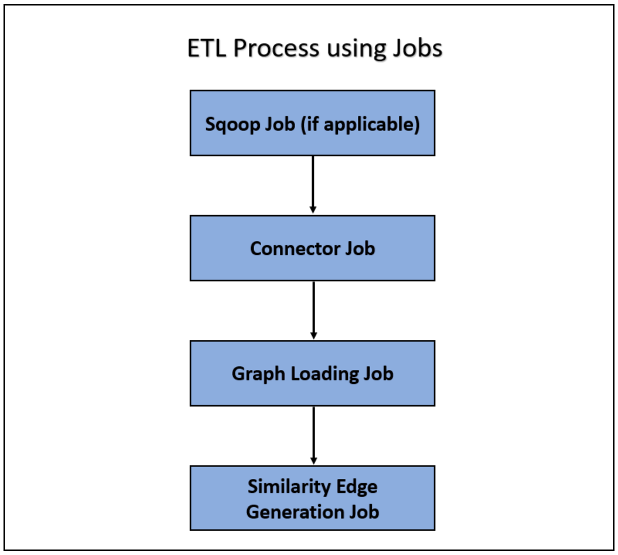
In the second stage, based on matching rules (ruleset), the nodes are resolved, and similarity edges are generated, among which edges having similarity scores more than automatic thresholds are directly updated into the graph. Other edges with a score between manual and automatic thresholds can be reviewed from Compliance Studio by users, and as soon as they are approved, these edge(s) are updated on the graph.

7.1.7 Jobs

The ETL process is split into four Jobs: Sqoop Job (if applicable), Connector Job, Graph Loading, and Similarity Edge Generation Job.

The following image illustrates the sequence of the ETL process.

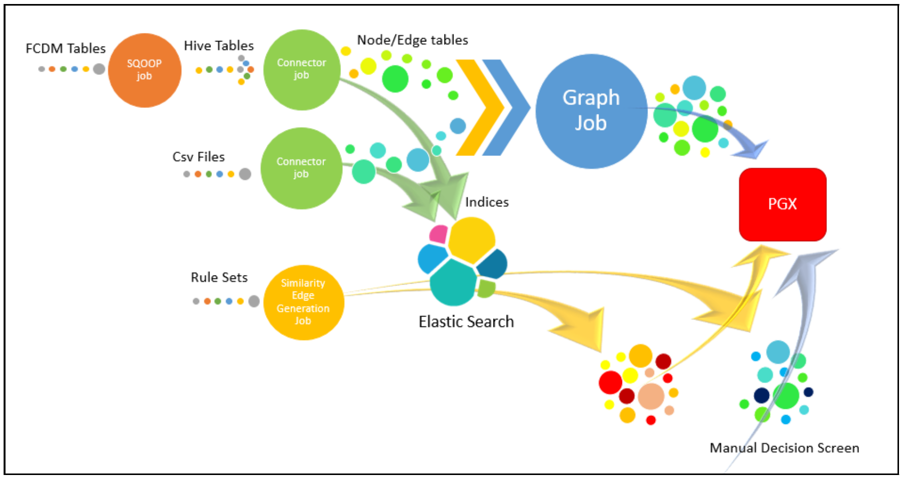
Figure 41: ETL Process using Jobs



- Sqoop Job: This task moves data from Behavior Detection (BD) and Enterprise Case Management (ECM) Atomic tables to Hive tables based on the date range. The Sqoop job creates and saves the import and export commands. It specifies parameters to identify and recall the saved job. This recalling or re-executing is used in the incremental import, which can import the incremental data from the RDBMS table to Hive.
- Connector Job: This task transforms data from Hive tables or the `.csv` files into nodes and edges format and identifies the changes in data loaded between previous and current batches. This task also pushes node tables into Elastic Search.
- Graph Job: This task generates the `.csv` files and configuration files for the graph, updates the changes into a graph, and manages transaction edges as per the date range.
- Similarity Edge Generation Job: This task generates the similarity edges based on a ruleset in the Compliance Studio application and adds similarity edges for breaching the automatic threshold into the graph directly.

The following image illustrates the workflow of ETL for specific data sources such as, Ready-to-use data sources, FCDM, and ICIJ:

Figure 42: ETL workflow



For more information on job execution, see the Run ETL section.

To understand how to move source data to the PGX server using connector jobs to create graphs in FCDM and ICIJ workflows as tabulated in the Table 19.

Table 19: FCDM and ICIJ workflows

Jobs	FCDM Workflow	ICIJ Workflow
Sqoop Job	Moves data from FCDM (BD or ECM) source into Hive tables.	Not applicable
Connector job	• Transforms data into nodes and edges tables using ready-to-use queries • Identifies incremental and updated data • Pushes nodes into Elastic-search as indices	• Reads the `.csv` files and transforms the data into nodes and edges tables using ready-to-use queries • Identifies incremental and updated data • Pushes nodes into Elastic-search as indices
Graph Loading Job	• Generates the `.csv` files and configuration files to load graph into the PGX server • Loads the delta graph changes	• Generates the `.csv` files and configuration files to load graph into the PGX server • Loads the changes directly into the PGX server from subsequent batches • Loads the delta graph changes
Similarity edge Generation Job	• Generates similarity edges • Pushes automatic matches of similarity edges into graph	• Generates similarity edges • Pushes automatic section of similarity edges into graph

After running the ETL process, global graphs are generated. For more information on the ready-to-use Graph Model, see Graph Model.

For more information on job execution, see Run ETL.

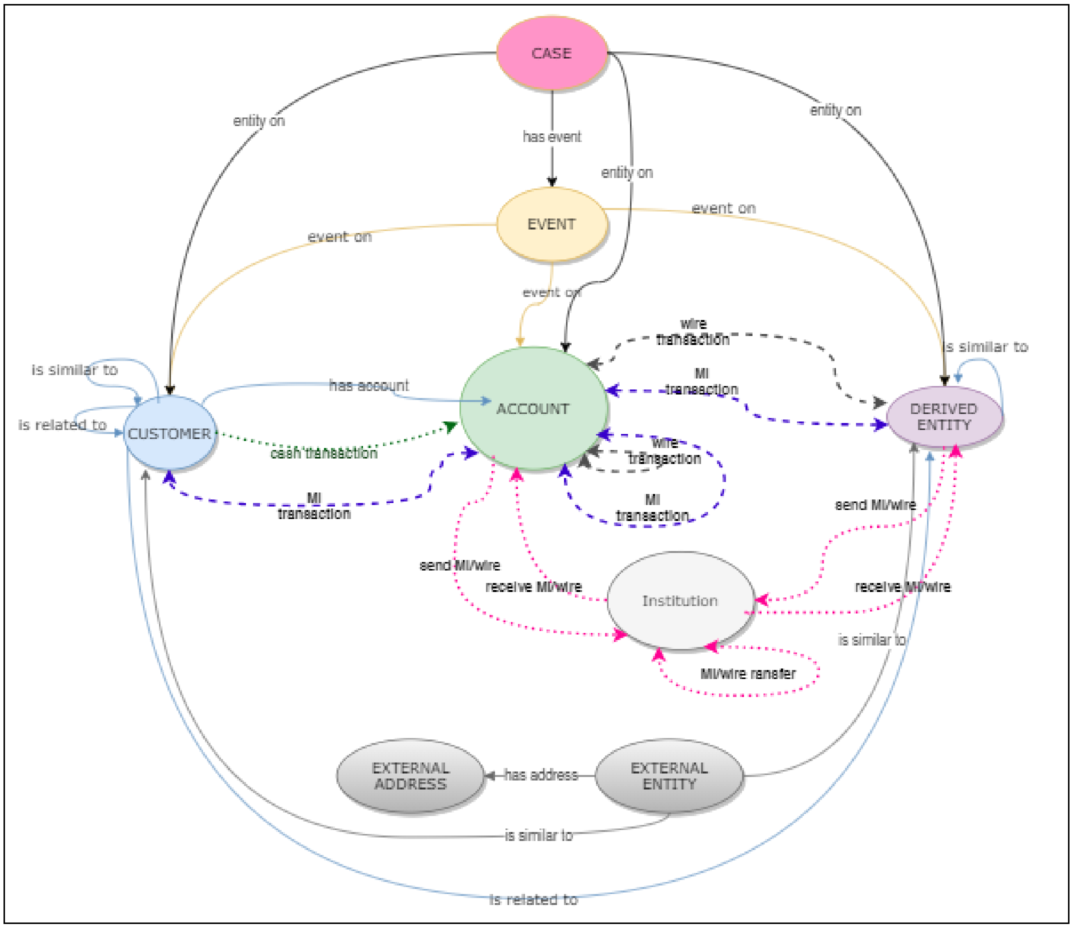
7.1.8 Graph Model

The Oracle Financial Crime Graph Model serves as a window into the financial crimes data lake. It collates disparate data sets into an enterprise-wide global graph, enabling an entirely new set of financial crime use cases. The Graph model enables to accelerate financial crime investigation use cases. The graph model expresses the conditional dependence structure between nodes and edges.

For information on Graph Data Model, see *Graph Data Model*.

The following image illustrates the Graph Model.

Figure 43: Graph Model



For information on the node and edge properties of the Oracle Financial Crime Graph Model, see the Data Model Guide.

7.2 Configure the SSH Connection

NOTE After an SSH connection has been created, add the following `kinit` command in the `.profile` or `.bash_profile` of BigData_Server.

```
kinit -V -k -t <Absolute path of Keytab file> <kerberos
principal>
```

To configure the SSH connection, run the following commands in the Windows command prompt:

1. `Run ssh-keygen`

   `Generating public/private rsa key pair`
2. `Enter file in which to save the key (<Linux_Home>/.ssh/id_rsa)`:[Press Enter]
3. `Enter passphrase (empty for no passphrase)`: [Press Enter]
4. `Enter same passphrase again`: [Press Enter]
5. `ssh-copy-id -i ~/.ssh/id_rsa.pub <BigData_Server>`
6. `ssh <BigData_Server>`

7.3 Configure Schema Creation

Schema creation is a one-time activity that replicates the table structure from the Financial Crime Data Model (FCDM) Atomic schema to the Hive Atomic schema.

Topics:

- Configure Schema Creation from Compliance Studio Server
- Configure Schema Creation from OFSAA Server

7.3.1 Configure Schema Creation from Compliance Studio Server

To configure Schema Creation from the Compliance Studio server, follow these steps:

1. Set `FIC_DB_HOME` path to `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb`.

   > **NOTE** The `$FIC_DB_HOME` path can be set from the `.profile` file.

2. Create a Hive Schema with the name mentioned in the `HIVE_SCHEMA` parameter in the `config.sh` file.

   For information on the `config.sh` file, see Configure the config.sh file in OFS Compliance Studio Installation Guide.
3. Execute the following shell script in the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/bin` directory to create tables in Hive Schema:

   `"./FCCM_Studio_SchemaCreation.sh HIVE"`
4. Check the `<Compliance_Studio_Installaton_Path>/deployed/logs/batchservice.logs` for more information.

   > **NOTE** If the table is not created in hive schema, follow the steps in Configure Schema Creation from OFSAA Server in the OFS Compliance Studio Installation Guide.

7.3.2 Configure Schema Creation from OFSAA Server

To configure Schema Creation from the OFSAA server, follow these steps:

1. Copy all the jar files from the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/lib` directory and paste into the `<OFSAA_FIC_HOME_PATH>/ficdb/lib` directory.

2. Copy all the `.sh` files from the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/bin` directory and paste into the `<OFSAA_FIC_HOME_PATH>/ficdb/bin` directory.
3. Create a Hive Schema with the name mentioned in the `HIVE_SCHEMA` parameter in the `config.sh` file.

   For information on the `config.sh` file, OFS Compliance Studio Installation Guide.
4. Execute the following shell script in the `<OFSAA_FIC_HOME_PATH>/ficdb/bin` directory to create tables in Hive Schema:

   ```
   FCCM_Studio_SchemaCreation.sh HIVE
   ```
5. Check the `<Compliance_Studio_Installaton_Path>/deployed/logs/batchservice.logs` for more information.

   If the schema creation fails, login to the atomic schema of BD/ECM and run the following query:

   ```
   select * from fcc_orahive_datatypemapping;
   ```

   The `fcc_orahive_datatypemapping` table must have only 5 rows. If there are more than 5 rows, run the following query to delete the additional rows:

   ```
   select * from fcc_orahive_datatypemapping for update
   ```

   If the studio schema creation fails, login as a studio user and run the following query:

   ```
   select * from fcc_datastudio_schemaobjects
   ```
6. Run the following query to replace all `Y` values with '':

   ```
   update fcc_datastudio_schemaobjects set SCHEMA_OBJ_GENERATED=''
   ```

   After the schema creation is successful, the value of the `SCHEMA_OBJ_GENERATED` attribute changes to Y.

7.4 Configure the ICIJ Data

> **NOTE** It is applicable only for ETL.

To include ICIJ data into Global Graph.

To configure the International Consortium of Investigative Journalists (ICIJ) data, follow these steps:

- Clean the ICIJ Data
- Configure the FILEPATH for ICIJ

7.4.1 Clean the ICIJ Data

To clean the ICIJ data, follow these steps:

1. Download the four dataset directories from https://offshoreleaks.icij.org/pages/database.
2. Extract the four dataset directories and place the extracted directories in the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/icij_data_cleaning` directory.
3. Navigate to the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/icij_data_cleaning/bin` directory and execute the following command:

```
./clean.sh
```

NOTE Ensure that Python 3 is available in the machine before executing this command.

After successful execution of the command:

- The cleaned data is available for the sqoop job to load in Hive and HDFS.
- A directory named clean is created inside each dataset, where a clean version of each CSV file is created.

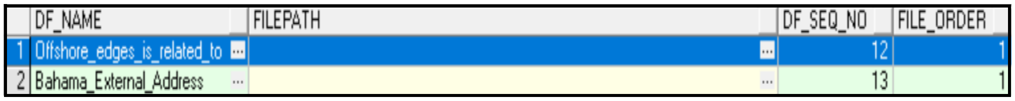
7.4.2 Configure the FILEPATH for ICIJ

NOTE The Compliance Studio graph model is configured to include ICIJ watchlist files.

To configure the `FILEPATH` for ICIJ, follow these steps:

1. Place the watchlist files in HDFS that are accessible by the user.
2. Update the `FILEPATH` of the watch list files in the `fcc_studio_etl_files` table. The FILEPATH refers to the complete hdfs filepath of the csv file. For mapping between DF_NAME and FILEPATH, see the following image:

Figure 44: fcc_studio_etl_files Table



	DF_NAME	FILEPATH	DF_SEQ_NO	FILE_ORDER
1	Offshore_edges_is_related_to		12	1
2	Bahama_External_Address		13	1

Table 20 provides the file path for the csv files:

Table 20: CSV Files Path

DF_NAME	File Path
Panama_External_Address	panama_papers.nodes.address.csv
Panama_External_Entity	panama_papers.nodes.entity.csv
Panama_External_Entity	panama_papers.nodes.intermediary.csv
Panama_External_Entity	panama_papers.nodes.officer.csv
Panama_edges_address_of	panama_papers.edges.csv
Panama_edges_is_related_to	panama_papers.edges.csv
Offshore_External_Address	offshore_leaks.nodes.address.csv
Offshore_External_Entity	offshore_leaks.nodes.entity.csv
Offshore_External_Entity	offshore_leaks.nodes.intermediary.csv
Offshore_External_Entity	offshore_leaks.nodes.officer.csv

Table 20: CSV Files Path

Offshore_edges_address_of	offshore_leaks.edges.csv
Offshore_edges_is_related_to	offshore_leaks.edges.csv
Bahama_External_Address	bahamas_leaks.nodes.address.csv
Bahama_External_Entity	bahamas_leaks.nodes.entity.csv
Bahama_External_Entity	bahamas_leaks.nodes.intermediary.csv
Bahama_External_Entity	bahamas_leaks.nodes.officer.csv
Bahama_edges_address_of	bahamas_leaks.edges.csv
Bahama_edges_is_related_to	bahamas_leaks.edges.csv
Paradise_External_Address	paradise_papers.nodes.address.csv
Paradise_External_Entity	paradise_papers.nodes.entity.csv
Paradise_External_Entity	paradise_papers.nodes.intermediary.csv
Paradise_External_Entity	paradise_papers.nodes.officer.csv
Paradise_External_Entity	paradise_papers.nodes.other.csv
Paradise_edges_is_related_to	paradise_papers.edges.csv
Paradise_edges_address_of	paradise_papers.edges.csv
Paradise_edges_is_linked_to	paradise_papers.edges.csv

7.5 Configure a Data Source

The data source configuration allows you to view the newly added edges or nodes in the graph. Define the source of the data, specify the order in which the files must be read, etc.

To configure a new data source for a graph, follow these steps:

1. Navigate to the `fcc_studio_etl_queries` table in Compliance Studio Schema. The FCDM-related nodes and edges are available in the table.
2. If you want to add additional nodes or edges, you can add a new entry in the `fcc_studio_etl_queries` table.
3. Enter the following details in the `fcc_studio_etl_queries` table to add a new node or edge as tabulated in the Table 21.

Table 21: Configure a Data Source

Column Name	Description	Applicable For
Type	Enter the column name. Enter the value as NODE or EDGE.	Applicable for node and edge queries.
DF_NAME	Enter the name for the node or edge.	Applicable for node and edge queries.
SOURCE	Enter the source of the data. For example, FCDM or ICIJ	Applicable for node and edge queries.

Table 21: Configure a Data Source

DATAFRAME	Enter the properties of the node or edge. NOTE: Enter this value only if the data source is Hive and not a `.csv` file.	Applicable for node and edge queries.

QUERY	If the source is Hive, provide the Hive query.
	If the source is a `.csv` file, provide the query in the following format:
	`spark.read.format("csv").option("header", "true")`
	`.option("mode", "DROPMALFORMED").load("(##FILEPATH##)").select("node_1","node_2","rel_type","SourceID")`
	`.withColumn("Label",lit("address of")).withColumnRenamed("node_1","from").withColumnRenamed`
	`("node_2","to").withColumnRenamed("rel_type","EDGE_TYPE").withColumnRenamed("SourceID","Source")`
	`.filter(col("EDGE_TYPE")==="registered_address").withColumn("node_ID",concat(lit("#NUMBER#"),col("node_ID")))`
	For more information, see Configure Spark Query Parameters.
	NOTE: Ensure that the source `.csv` file is UTF-8 compatible.
		Applicable for node and edge queries.

KEY_COLUMN_NAME	Set the value to the column name of your unique identifier, if the query is for node. For example: 'node_id'.	Applicable for node query.
SOURCE_NODE	Enter the DF_NAME of the node from which the edge starts.	Applicable for edge query.
DESTINATION_NODE	Enter the DF_NAME of the node from which the edge ends.	Applicable for edge query.
SOURCE_KEY_COLUMN_NAME	Set the value to the column name with key_column values of the Source Node. For example: 'from_id'	Applicable for edge query.

Table 21: Configure a Data Source

DESTINATION_KEY_COLUMN_NAME	Set the value to the column name that has key_column values of the Destination Node. For example: 'to_id'	Applicable for edge query.
ACTIVE	Enter the value. The expected values are 'Y' or 'N'. Set the value to Y to consider ETL and Graph loading.	Applicable for node and edge queries.

If the source is a `.csv` file, configure the file path in the `fcc_studio_etl_files` table.

> **NOTE** Ensure that the source `.csv` file is `UTF-8` compatible.

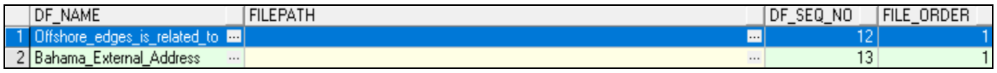
4. Enter the following details in the `c` table to add the file path tabulated in the Table 22:

Table 22: Adding File Path

Column Name	Description
DF_NAME	Enter the name of the node or edge.
DF_SEQ_NO	Enter the unique sequence ID for each file. For example, column number.
FILEPATH	Enter the path where the `.csv` files are stored. **NOTE**: If one data frame has multiple `.csv` files, make separate entries for all the files. For example: see fcc_studio_etl_files Table.
FILEORDER	If data must be imported from multiple files, specify the order in which the files must be read. For example, if the query for an entity uses multiple files, then the sequence must be provided to replace the file path with the correct file path.

The following image provides an example of fcc_studio_etl_files.

Figure 45: fcc_studio_etl_files Table



	DF_NAME	FILEPATH	DF_SEQ_NO	FILE_ORDER
1	Offshore_edges_is_related_to		12	1
2	Bahama_External_Address		13	1

7.5.1 Configure Spark Query Parameters

This section provides information on the Spark query parameters that are used during configuring a new data source for a graph or modifying existing data queries for ICIJ. This can be used for modifying the spark query of ICIJ. For example, it adds new attributes or a new data source.

> **NOTE** This activity is optional for the ETL process while adding or modifying data sources.

To configure Spark Query Parameter, follow these steps:

1. Navigate to the `fcc_studio_etl_queries` table in Compliance Studio Schema. The FCDM-related nodes and edges are available in the table.
2. Enter the following details in the fcc_studio_etl_files table to add the file path in the Table 23:

Table 23: fcc_studio_etl_files table

Query Parameter	Description
`spark.read.format("csv")`	Enter the input file format. For example: `.csv`.
`option("header", "true")`	Enter the presence of a header in the input file. ● True indicates that the header is available in the input file. ● False indicates that the header is absent in the input file.
`load("Path").`	● Load indicates to load the data from the mentioned file path. ● The path indicates the path where the files are placed. You can load to multiple paths using the following format: `("Path1","Path2",...)`
`select("Col1","Col2","Col3","Col4")`	Select the columns in the input file.
`withColumn("A",lit("Test1"))`	Add a new column with column name A and column value Test1.
`withColumnRenamed("A","B")`	Rename a column with a different name. For example, rename the column from A to B.
`filter(col("A")==="Test1")`	Enter the "Where" filter condition. Here, the value for column A is Test1.
`withColumn("B",concat(lit("Test1"),col("A")))`	Add a new column B, whose value is the concatenated value of Test1 and column A. For example, Test1=ABC ● Column A contains Country and Pin codes as the column values. ● Column B gets ABC Country and ABC Pin code as column values.

7.6 Configure Graph

The Compliance Studio provides an intuitive way for creating graphs used in notebooks, where you can load graphs from external sources or create custom graphs. Using PGX, you can load multiple graphs into a notebook and create PGQL queries against different graphs. The result obtained from running a paragraph in a notebook can be used as an input for other paragraphs in the notebook. The results of analytics algorithms are stored as transient properties of nodes and edges in the graph. Pattern matching can then be used against these properties.

The graph configuration can be defined through UI based configurator or a JSON configurator. Graph configurations give you easy access to graphs using PGX-ALGORITHM, PGX-JAVA, and PGQL interpreters.

Use this section to configure attributes, extra empty nodes and edges providers, and local date format for graphs.

Topics:

- Configure Attributes in Graph
- Configure Extra Empty Nodes and Edges Providers
- Additional Configuration (Local Date Format)

7.6.1 Configure Attributes in Graph

Use this section to configure the attributes of nodes and edges in the graph.

> **NOTE** In Compliance Studio, the heterogeneous graph does not support the dynamic addition of Nodes and Edges Provider in the graph. If extra nodes or edge providers are required, then you must add the entries to the `FCC_GRAPH_EMPTY_ENTITY_MAPPING` table.

All the attributes of nodes or edges must be present in the `FCC_GRAPH_COLUMN_NAME_MAPPING` table.

- COLUMN_NAME: Indicates the attributes name in queries.
- RENAMED_COLUMN_NAME: Indicates the required attribute name.
- COLUMN_DATA_TYPE: Indicates the PGX's data type of the attribute.

> **NOTE**
> - The accepted PGX's datatype formats are Boolean, integer, float, long, double, string, date, local_date, time, timestamp, time_with_timezone, timestamp_with_timezone, and point2d.
> - The date is deprecated; hence, you can use one of the following:
>   - local_date
>   - time
>   - timestamp
>   - time_with_timezone
>   - timestamp_with_timezone

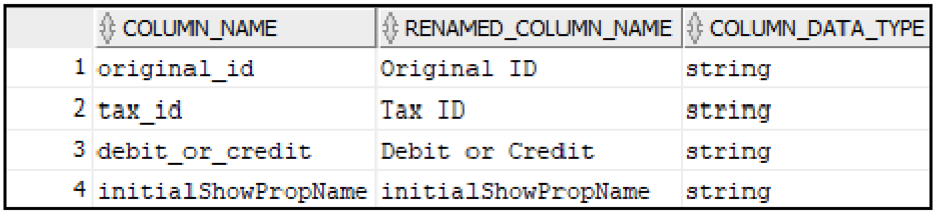
For example, if the values are as follows:

- **COLUMN_NAME**: sample_attribute
- **RENAMED_COLUMN_NAME**: Sample_AttributeName
- **COLUMN_DATA_TYPE**: string

Then the attribute name shown in the graph is, Sample_AttributeName.

The following image provides you with an example.

Figure 46: FCC_GRAPH_COLUMN_NAME_MAPPING Table



	COLUMN_NAME	RENAMED_COLUMN_NAME	COLUMN_DATA_TYPE
1	original_id	Original ID	string
2	tax_id	Tax ID	string
3	debit_or_credit	Debit or Credit	string
4	initialShowPropName	initialShowPropName	string

7.6.2 Configure Extra Empty Nodes and Edges Providers

In Compliance Studio, the heterogeneous graph does not support the dynamic addition of Nodes and Edges Provider in the graph. If extra nodes or edge providers are required, then you must add the entries to the `FCC_GRAPH_EMPTY_ENTITY_MAPPING` table.

Where,

- **TYPE**: Indicates the type of empty entity provider to be added. Expected value: “NODE" or "EDGE"
- **NAME**: Indicates the name of the entity provider.
- **COLUMN_MAPPING**: Indicates the attributes required for the entity with its data type. The value must be a comma-separated paired value of the column name and its type.
- For example: column1:string,column2:long

> **NOTE**
>
> - In the case of NODE, do not specify key_column for the node. In the case of EDGE, do not specify the source and destination **key_columns**.
> - The accepted PGX's datatype formats are Boolean, integer, float, long, double, string, date, local_date, time, timestamp, time_with_timezone, timestamp_with_timezone, and point2d.
> - The date is deprecated; hence, you can use one of the following instead:
>   - local_date
>   - time
>   - timestamp
>   - time_with_timezone
>   - timestamp_with_timezone

- Example 1:
  - TYPE: NODE
  - NAME: extra_node
  - COLUMN_MAPPING: name:string,phone_number:integer

  An extra vertex provider with the name "extra_node" is added with the attributes, Name and Phone Number, datatype, string, and integer, respectively.

- Example 2:
  - TYPE: EDGE
  - NAME: extra_edge
  - COLUMN_MAPPING: name:string,risk:long,edge_type:string

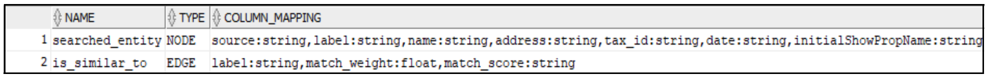
  Extra edges are formed between every node provider, including itself with the name as "`<source_node_provider>_extra_edge_<destination_node_provider>`", with the attributes, Name, Risk and Edge Type, datatype, string, long, and string, respectively.

To configure extra empty nodes and edges providers:

1. Navigate to Studio schema and go to the `FCC_GRAPH_EMPTY_ENTITY_MAPPING` table.
2. Add the entries to the `FCC_GRAPH_EMPTY_ENTITY_MAPPING` table.

   The following image provides you with an example.

Figure 47: FCC_GRAPH_EMPTY_ENTITY_MAPPING Table



	NAME	TYPE	COLUMN_MAPPING
1	searched_entity	NODE	source:string,label:string,name:string,address:string,tax_id:string,date:string,initialShowPropName:string
2	is_similar_to	EDGE	label:string,match_weight:float,match_score:string

7.6.3 Additional Configuration (Local Date Format)

Use this section to configure the local date format.

To configure the local date format, follow these steps:

1. Navigate to Studio schema and go to the `FCC_DATASTUDIO_CONFIG` table.
2. For the ready-to-use graph's configuration, the following parameters are set in the `FCC_DATASTUDIO_CONFIG` table:

   a. **local_date_format**: The default value: [M/D/YYYY, M-D-YYYY, D/M/YYYY, D-M-YYYY, YYYY-MM-DD, YYYY/MM/DD, YYYY-D-M, or YYYY/D/M].

   > **NOTE** The date format option can be used only to view the data type of an attribute on the graph in the configured format.

   b. **vertex_id_type**: The default value is "long" as per the ready-to-use queries.

   This parameter represents the datatype of the vertex_id column or key_column of node providers.

   > **NOTE** This data type should be consistent across all nodes.

7.7 Apply Graph Fine-Grained Access Control

The Graph Fine-Grained Access Control and Redaction changes are applied to the Compliance Studio to redact the sensitive data in the Graph and provide role-based access control, which restricts the graph access at the Business domain and Jurisdiction level of the user.

DSREDACT role enables data redaction feature for any user. In case you are using OFSAA for user creation, then map the user you want redact to be applied with the DSRedact role.

To apply Graph Fine-Grained Access Control, follow these steps:

1. For OFSAA user creation, navigate to OFSAA Identity Management page and map the user to the DSREDACT role.

   For SAML user creation, map the required user group to DSREDACT
2. Data redaction and fine grain access (Business Domain and Jurisdiction filters) on the Graph are configurable.
   - fcc_studio_redaction_mapping table is used to manage to turn On and Off Data redaction and fine grain access features.
   - If the Graph data is to be filtered based on the Business domain and Jurisdiction associated with the user. Then set the “Jur_BusDmn_Rule” role in the table to 'Y', else set it to 'N'
   - To enable the redaction feature on the graph, set the “DSREDACT” role in the table to 'Y' or 'N'.
3. There are ready-to-use graph properties specified for data redaction that are available in the FCC_STUDIO_REDACTION_RULE table. To add new properties for redaction, you can specify details in the FCC_STUDIO_REDACTION_RULE table.

   The following are the column names:
   - RULE_SEQ_ID: Unique sequence ID
   - LABEL: Node or Edge label name
   - PROPERTY: Property that you want to redact
   - TYPE: Node or Edge based on the property's expected value.
4. Navigate to Studio installed server and set a variable with the name FIC_DB_HOME as
5. `Export FIC_DB_HOME=<COMPLIANCE_STUDIO_HOME>/deployed/ficdb`
6. Navigate to the `<FIC_DB_HOME>/bin` directory.
7. Run the `FCCM_Studio_ApplyGraphRedaction.sh` file. The Graph Fine-Grained Access Control changes are applied.

> **NOTE** Whenever you enable or disable the jurisdiction filter, `FCCM_Studio_ApplyGraphRedaction.sh` has to be executed.

8 Execute ETL

Use this chapter to prepare and perform the batches to execute the ETL process. You can also verify the status of all tasks at the end of batch execution. You can verify both the overall status of the batch as well as individual task status. Execute the ETL by preparing and running the batches. You can also verify the status of all tasks at the end of batch execution. You can verify both the overall status of the batch as well as individual task status.

Topics:

- Prepare the Batches
- Perform the Batches
- Verify Batch Execution

8.1 Prepare the Batches

Use this section to prepare batches to execute the ETL. Batches enable you to load graphs, run notebooks, and move data from Oracle Database or Big Data to Compliance Studio. Batches are prepared based on the Realms you are using.

8.1.1 Prepare Batches for FCCM Realm

To prepare the batches for FCCM Realm, follow these steps:

1. Copy all the jars from the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/lib directory to the <FIC_HOME of OFSAA_Installed_Path>/ficdb/lib` directory .
2. Copy the `NBExecutor.txt` file from the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/bin directory to the <FIC_HOME of OFSAA_Installed_Path>/deployed/ficdb/bin` directory .
3. Navigate to the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/bin` directory.
4. Run the `FCCM_Studio_Set_UserPass.sh` command as follows:
   - `FCCM_Studio_Set_UserPass.sh --username "Username" --password "Password"`

   Or

   - `FCCM_Studio_Set_UserPass.sh -u "USERNAME" -p "PASSWORD"`

   The `FCC_Studio_SecretKey.properties` and `NBExecutor.txt` files are created in the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/conf` directory.

> **NOTE**
>
> - Ensure that the `Compliance_Studio_SecretKey.properties` and `NBExecutor.txt` files are present in the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/conf` directory before executing a notebook batch.
> - If only `NBExecutor.txt` file is present in the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/conf` directory, then re-execute the `FCCM_Studio_Set_UserPass.sh` command with username and password to create a new `Compliance_Studio_SecretKey.properties` file and update the `NBExecutor.txt` file.

8.1.2 Prepare Batches for SAML Realm

To prepare the batches for SamlRealm, you must generate an API token and configure the `NBExecutor.properties`.

To generate the API token, follow these steps:

1. Navigate to the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/bin` directory.
2. Run the shell script: `fic_db_home/FCCM_Studio_Generate_APIToken.sh.`
3. Run the script: `fic_db_home/FCCM_Studio_Generate_APIToken.sh APPNAME.`

   For example, use the `./FCCM_Studio_Generate_APIToken.sh BATCH_USER.`

   In the `NBExecutor.properties` file, specify the following details:

   - saml=true
   - username=<BATCH_USERNAME>
   - password=<BATCHUSER_PASSWORD>
   - apiToken=<API_TOKEN>

> **NOTE** `BATCH_USERNAME` and `BATCHUSER_PASSWORD` can be NULL.

8.2 Perform the Batches

Batches are performed to execute the ETL process. The batches contain Sqoop Job, Connector Job, Graph Job, and Similarity Edge Generation Job in OFSAAI. You can also execute the ETL process by running the scripts without configuring the batches.

You can perform batches in the following ways:

- **Perform batches using OFSAAI**: For more information, see Create and Execute a Run Executable for Scenario Notebooks section.

- **Perform batches using shell script**: For more information, see relevant jobs in the Run ETL section.

8.2.1 Run ETL

NOTE It is recommended to open the Putty session in **bash** mode instead of **ksh** mode.

To run the ETL, you must perform these jobs:

- Sqoop Job
- Connector Job
- Graph Job
- Verify Similarity Edge Generation Job

NOTE You must not trigger the same ETL job twice until it is completed.

8.2.2 Sqoop Job

Sqoop is a tool designed to efficiently transfer bulk data between Hadoop and structured data such as relational databases. Sqoop job creates and saves the import and export commands. It specifies parameters to identify and recall the saved job. This re-calling or re-executing is used in the incremental import, which can import the updated rows from the RDBMS table to HDFS.

NOTE

- This section is applicable for Compliance Studio with non-OFSAA.
- Before performing a Sqoop job, verify the Schema creation (it replicates the table structure of FCDM tables from atomic schema into HIVE schema).

The Sqoop Job moves data from BD/ECM Atomic tables to Hive tables based on the date range. This task can be skipped in the graph if FCDM data is not required.

To execute the Sqoop job, follow these steps:

1. Navigate to the `FIC_DB_HOME/bin` directory.
2. If this is your first Sqoop job, execute the following command.

   ```
   ./FCCM_Studio_SchemaCreation.sh HIVE
   ```

3. The Sqoop job can be scheduled or executed using the following command:

   NOTE This example is applicable to shell script.

   ```
   ./FCCM_Studio_ETL_SqoopJob.sh <FROM_FIC_MIS_DATE> <TO_FIC_MIS_DATE>
   SNAPSHOT_DT<=SNAPSHOT_DATE> <Batch_ID>
   ```

   For example:

   ```
   ./FCCM_Studio_ETL_SqoopJob.sh "20151201" "20200412"
   "SNAPSHOT_DT=20200415" "BatchID_001"
   ```

Where:

- `FROM_FIC_MIS_DATE` is 20151201
- `TO_FIC_MIS_DATE` is 20200412
- `SNAPSHOT_DT` is 20200415
- `Batch_ID` is BatchID_001

NOTE The date format is “**YYYYMMDD**”.

If the date parameters are passed as null, then the values of these parameters are calculated based on `ETL_PROCESSING_RANGE,` and the date’s value is as follows:

- `Snapshot_dt` is considered as the current date.
- `To_fic_mis_date` is considered as yesterday’s date.
- `From_fic_mis_date` is considered as a date which is `etl_processing_range` behind `to_fic_mis_date`.

For example:

```
./FCCM_Studio_ETL_SqoopJob.sh "null" "null" "SNAPSHOT_DT=null"
"BatchID_001"
```

If the ETL processing range: 2Y, 3M, 10D (2 years, 3 months, 10 days) and Present Date: 20200815, then:

- `Snapshot_dt` is 20200815
- `To_fic_mis_date` is 20200814
- `From_fic_mis_date` is 20180504

8.2.3 Connector Job

The connector job transforms the data from the Hive table or the `.csv` files based on the data source into the node and edge format and recognizes the changes in data for the graph.

To execute the connector job, follow these steps:

1. Navigate to the `FIC_DB_HOME/bin` directory.
2. Execute the following command:

```
./FCCM_Studio_ETL_Connector.sh <Source> SNAPSHOT_DT=<SNAPSHOT_DATE>
```

NOTE The date format is “**YYYYMMDD**”.

For example,

```
./FCCM_Studio_ETL_Connector.sh FCDM SNAPSHOT_DT=20200415
```

Where:

- Source: FCDM
- Snapshot DT: 20200415

Compliance Studio has a ready-to-use configuration for FCDM and ICIJ data sources as per the graph model. If the date parameter is passed as null, then the snapshot date is taken from the previous Sqoop job; if present, otherwise, it is the current day.

For example:

```
./FCCM_Studio_ETL_Connector.sh FCDM SNAPSHOT_DT=null
```

The snapshot date is 20200815 (see the example from Sqoop Job)

- For ready-to-use, run the following command for FCDM.

  ```
  ./FCCM_Studio_ETL_Connector.sh FCDM SNAPSHOT_DT=20200415
  ```

- For ready-to-use, run the following command for ICIJ.

  ```
  ./FCCM_Studio_ETL_Connector.sh ICIJ SNAPSHOT_DT=20200415.
  ```

NOTE When the connector snapshot date is *'Null'*, then it takes a snapshot of the date of the last run Sqoop job.

8.2.4 Graph Job

The Graph Job task generates the JSON files for the PGX server to load with other `.csv` files for all the sources and updates the changes into the PGX server.

To execute the Graph job, follow these steps:

1. Navigate to the `FIC_DB_HOME/bin` directory.
2. Execute the following command:

   ```
   ./FCCM_Studio_ETL_Graph.sh.
   ```

   After the first execution of this task, start the PGX server to load the graph, which can be queried and viewed in the Compliance Studio Notebook.

8.3 Verify Batch Execution

Use this section to verify the status of all tasks at the end of batch execution. You can verify both the overall status of the batch and individual task status.

Topics:

- Verify Sqoop Job
- Verify Connector Job
- Verify Graph Job
- ML Name and Address Incremental Training API
- Verify Oracle Schema Tables

> **NOTE** The following modes are available for the current release:
>
> - info
> - debug
>
> By default, logs will be generated on info mode `<Root level="`**`info`**`">`. In the current release, the functionality of **debug** mode is similar to **info** mode.
>
> To change the mode, perform the following:
>
> 1. Navigate to `<Compliance_Studio_Home>/OFS_COMPLIANCE_STUDIO/deployed/batchservice/conf/log4j2.xml`.
> 2. Change the mode in following parameters:
>
>   `<Root level="<`**`mode`**`>">`
>
>   **For example,** `<Root level="debug">`.

8.3.1 Verify Sqoop Job

Use this section to verify logs and Hive tables' sqoop job.

8.3.1.1 Verify Logs

To verify logs in Sqoop Job, follow these steps:

1. Navigate to the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/logs/` directory.
2. Open the `batchservice.log` file. The overall status and individual status of each moved table are displayed. Also, errors are displayed if any individual table has failed.

Based on this, you can fix it accordingly or contact My Oracle Support in case of any errors.

8.3.1.2 Verify Hive Tables

To verify the Hive table in Sqoop Job, follow these steps:

1. Connect to the Hive Schema.
2. Verify if data was moved into the respective tables (based on logs) for the snapshot date of the batch.

8.3.2 Verify Connector Job

Use this section to verify logs and Hive tables for the connector job.

8.3.2.1 Verify Logs

To verify logs in Connector Job, follow these steps:

1. Navigate to `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/logs/` >
2. Open the `batchservice.log` file. The overall status and status of each entity is displayed. Also, errors are displayed if any entity has failed.

Based on this, you can fix it accordingly or contact My Oracle Support in case of any errors.

8.3.2.2 Verify Hive Tables

To verify the Hive table in Connector Job, follow these steps:

1. Connect to the Hive Schema
2. Verify if table names: `<Source>_<entity_name>` (example: `fcdm_customer`) are present and populated or not based on the log.

8.3.2.3 Verifying Indices in Elasticsearch

To verify indices in the elastic search, follow these steps:

1. Enter the URL in the following format into the browser:

   ```
   http://<Elastic_Search_Hostname>:<Elastic_Search_Port>/_cat/
   indices
   ```

   All the indices must be displayed with the same snapshot date with which the job is triggered.

2. Format: `<Index name>_<Snapshot Date>`

   For example:

   - `fcdm_customer_2020-03-01`
   - `icij_bahama_external_address_2020-03-01`

8.3.3 Verify Graph Job

Use this section to verify logs and Hive tables for the graph job.

8.3.3.1 Verify Logs

To verify logs in the Graph Job, follow these steps:

1. Navigate to the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/logs/` > directory.
2. Open the `batchservice.log` file. The overall status and status of each entity is displayed. Also, errors are displayed if any entity has failed.

Based on this, you can fix it accordingly or contact My Oracle Support in case of any errors.

For example, if the fix requires a query change or configuration changes, follow the cleanup steps and re-run the tasks after fixing it.

8.3.4 Verify Similarity Edge Generation Job

Use this section to verify logs and Hive tables for the Similarity Edge Generation job.

8.3.4.1 Verify Logs

To verify logs in the Graph Job, follow these steps:

1. Navigate to the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/logs/` > directory.
2. Open the `batchservice.log` and `entity-resolution.log` file. The overall status and status of each ruleset is displayed. Also, errors are displayed if any entity has failed.

Based on this, you can fix it accordingly or contact My Oracle Support in case of any errors.

8.3.5 Verify Oracle Schema Tables

To verify logs in the Graph Job, follow these steps:

1. Navigate to the Oracle Studio schema.
2. Verify if the similarity edges are formed for the following tables:
   - `fcc_er_matched_edges`
   - `fcc_er_matched_edges_manual`

> **NOTE** Target match results are always displayed as a name in the **Description** column in the following tables for matching with the name, Concat name, and alias in the current release:
>
> - fcc_er_matched_edges
> - fcc_er_matched_edges_manual
>
> For example,
>
> **Source**: Mickey M
>
> **Target**: Mickey Mouse
>
> The following value will be displayed in the **Description** column:
>
> Matched on concat_name-95.00, name-95.00.
>
> Where the concat_name is Mickey M, and the name is Mickey Mouse.

8.3.6 Clean up for ETL

If any ETL jobs are failed, and you want to re-run the job, you must clean up the ETL.

To clean up the ETL, follow these steps:

1. Navigate to Compliance Studio schema based on the source (for example, FCDM, ICIJ), and delete the following tables.
   - fcc_studio_graph_entity_provider
   - fcc_studio_graph_plug_edge_status
2. Drop the tables created in Hive schema.
   - If you want to clean up the ICIJ job, then drop ICIJ-related tables. For example, icij_paradise_external_entity.
   - If you want to clean up the FCDM job, then drop FCDM-related tables. For example, fcdm_customer.
3. Truncate the tables created after the schema creation job.

   For example, cust, acct, wire_trxn, fcc_studio_nodeedge_lookup, etc.
4. Delete the folder where `graph.json` and `.csv` files are created that is, **HDFS_GRAPH_FILES_PATH** (see `config.sh` under the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/bin` path.

9 Configuring Synonym and Stopword

Synonyms are alternative names that should be treated the same, maybe nicknames (Bob, Robert) or spelling variations (1st, First). Synonyms are applied to elastic search candidate selection and to match scoring to improve matching.

Stopwords are standard terms (like Mr, Ltd) that do not provide value in matching and should be excluded from candidate selection and excluded or given less precedence in matching. Based on data requirements, pre-defined lists of synonyms and Stopwords are provided but configured as required.

Synonyms and Stopwords are applied to candidate selection in Elastic Search (via text files) and match scoring (via database tables).

Topics:

- Elastic Search Cluster
- Database

9.1 Elastic Search Cluster

To enable Synonym and Stopword with the Elastic Search service, follow these steps:

1. Navigate to the `<Elastic_Search_Installed_Path>/config` directory.
2. Create a directory named analysis using the following command:

   ```
   mkdir analysis
   ```

3. Place your Stopword and Synonym files in the newly created analysis directory.

   Some examples are provided here:

   > **NOTE**
   > - User can decide to provide any data in the Stopword or Synonym files.
   > - Each Stopword must be provided in a separate line.
   > - All related synonyms must be provided in the same line, separated by a comma.
   > - All the synonyms must be provided in the same line and ensure that there are no repetitions of the synonym. For example, rob, robi, robie, roby, robbi.

   - `Name_synonyms.txt`: Contains name synonyms like bob, bobby, etc.
   - `Country.txt:` Contains country synonyms like US, United States, America, etc.
   - `Organisation_suffix.txt`: Contains organization suffices that are used as Stopwords.
   - `Title.txt:` Contains title Stopwords used as the title for the name.

   > **NOTE** The `Title.txt` should contain Name Tiles and organization Stopwords.

   - `Gender.txt:` Contains gender-related synonyms.
   - `Organisation_strip.txt`: Contains organization Stopwords.
   - `Namestop.txt:` Contains Individual name strip words.

- `Cardinal_ordinal.txt`: Contains organization numbers.
- `Organisational_level2.txt`: Contains organization level 2 files that do have any values. Currently, it is empty.
- `Organisational_stopwords.txt`: Contains organization Stopwords.
- `Oraganisational_businesswords.txt`: Contains organization business words.

9.2 Database

NOTE	• You must update the text files and DB table entries whenever you add/edit/delete Stopwords and synonyms. • You must enable the flag to **Y** (that is **true**) in the **fcc_idx_m_lookup** for that particular Lookup ID to reflect the changes in Matching Service in UI. • The business words are not considered Stopwords or synonyms • Business words will be considered for scoring but with less weightage.

You can add/modify/delete Stopwords or synonyms in the database tables are as follows:

- **fcc_idx_m_lookup**: Contains Stopwords and synonyms.
- **fcc_idx_m_lookup_values**: Contains the actual values according to lookup ID.

To modify the Stopword or Synonym in the database, perform the following:

1. Get the **N_LOOKUP_ID** and file name from **fcc_idx_m_lookup** table.
2. Modify the value in **V_LOOKUP_VALUES** columns in the **fcc_idx_m_lookup_values** for corresponding N**_LOOKUP_ID.**
3. Change the flag from **N** to **Y** in **F_IS_RECENTLY_CHANGED** for corresponding **N_LOOKUP_ID** in the **fcc_idx_m_lookup** table
4. Navigate to `<Elastic_Search_Installed_Path>/config/analysis`
5. Update the same values in the respective lookup file.

10 ML Name and Address Model Training

Compliance Studio provides a matching capability to support similarity edge creation between entities (Customer, Derived Entity, and External Entities) In the Global Graph and grouping of records in the Entity Resolution process.

Following ML scoring methods are supported:

- ML-Boosted Name (only for Name attribute)
- ML-Boosted Address (only for Address attribute)

Compliance Studio has pre-trained models based on a default set of features for matching names/ addresses. Over time, ML Models need to be retrained in order to provide better results for Name/ Address matching. They will also need to be retrained if features are added to or removed from the default settings for accuracy or performance reasons.

Following are the additional steps that need to be performed if using ML Boosted Name/Address matching are applicable only for On-premise Compliance Studio installation.

1. Navigate to the `FIC_DB_HOME/bin` directory.

> **NOTE** You must ensure that the pre-trained model is marked as a champion model and that is `IS_CHAMPION` column in the `FCC_ER_ML_MODEL_NAME_BOOSTED` table is set to **Y**.
>
> Data will be inserted into the FCC_ER_ML_MODEL_NAME_BOOSTED table for name and address champion models after training.

2. Execute the following command:

```
./FCCM_Studio_ETL_BulkSimilarityEdgeGeneration.sh
```

Topics:

- Changing Default Features and Custom Model Training
- ML Name and Address Incremental Training API

10.1 Changing Default Features and Custom Model Training

Users can optionally give a JSON with custom features in MLBoosted training notebooks when training a new model from scratch. The model will be trained using SANE's default features if this JSON is not provided.

The configuration JSON should have a "features" key, which will contain the feature categories that will be used:

- Full Record Features (operating on the full name or address)
- Separated Records Features (operating on the words in the name or address)
- Character Features (operating on the characters in the name or address)
- Initial Features (operating on the initial of the name only and not applicable for address)

For each category, an array should be provided. The array should contain the features the end-user wants to train the model. Users are allowed to create any "class_n_k" feature they want. See the OFS Compliance Studio Matching Guide for more details.

The following configuration, JSON, contains the defaults feature configuration for name matching:

```
    {


        "features":

        {

            "full_record_features":["sorensendice_2_1", " strlen_0_0", "
strlen_1_0", "wordslen_0_0", "wordslen_1_0"],


            "separated_records_features":["jarowinkler_0_0"],


            "character_features":["sorensendice_1_0"],


            "initial_features":["sorensendice_1_0"]

        }

    }
```

The below configuration JSON contains the defaults feature configuration for address matching:

```
    {

        "features":

        {

            "full_record_features":["sorensendice_2_0", " sorensendice_2_1",
" sorensendice_3_1", "sorensendice_3_2", "entropyncd_3_0"],


            "separated_records_features":["jarowinkler_1_0"],


            "character_features":["sorensendice_2_0"]

        }

    }
```

NOTE If end users do not want to use features for a particular category, they can either not use the key value of the category or use as a value an empty array.

Example: where the category is **not** provided in the JSON:

```
        {
            "features":
            {
                "full_record_features":["sorensendice_2_0", "
sorensendice_2_1", " sorensendice_3_1", "sorensendice_3_2",
"entropyncd_3_0"],

                "character_features":["sorensendice_2_0"]
            }
        }
```

An example where a category has an empty array:

```
        {
            "features":
            {
                "full_record_features":["sorensendice_2_0", "
sorensendice_2_1", " sorensendice_3_1", "sorensendice_3_2",
"entropyncd_3_0"],

                "separated_records_features":[],

                "character_features":["sorensendice_2_0"]
            }        }
```

10.1.1 Storing the Feature Configuration for Training

When a model is created from scratch, its configuration JSON is stored alongside the model binary (either in file format or the database). When a model is loaded for incremental training and/or inference, the model's configuration JSON is used in order to use the proper features for the model.

The configuration JSON has an extra key, "sane_release," which is added from the codebase and contains the version of the name or address matching library (in the fcc-python-SANE virtual environment) which the model was trained with. For instance:

```
        {
            "sane_version": "0.2.3",
            "features":
            {
                "full_record_features":["sorensendice_2_0", "
sorensendice_2_1", " sorensendice_3_1", "sorensendice_3_2",
"entropyncd_3_0"],

                "separated_records_features":[],

                "character_features":["sorensendice_2_0"]
            }
        }
```

This is a configuration JSON of a model trained with the SANE release "0.2.3 "(address matching and name matching libraries) and some user-specified features.

10.2 ML Name and Address Incremental Training API

Over time, ML Models need to be trained in order to give better results for Name/Address matching. The incremental Training API is used to train the model periodically. In order to do so, the Model expects historical data to be available for training.

Prerequisite:

You must perform the following configuration based on the realm:

- **FCCM Realm**: Carry out **step 4** in the Prepare Batches for FCCM Realm.
- **SAMLRealm**: Carry out **step 3** in the Prepare Batches for SAML Realm.

NOTE

- For Incremental training, the user should not use the same data in the `FCC_ER_ML_TRAINING_DATA_NAME_BOOSTED` table for name and train the Model multiple times to avoid overfitting the model. Also, make sure the table has enough data, and this data is enhanced frequently.
- `FCC_ER_ML_TRAINING_DATA_ADDRESS_BOOSTED` table for address and train the Model multiple times to avoid overfitting the model. Also, ensure the table has enough data, and this data is enhanced frequently.
- To train and arrive at a good ML Model, there should be an equal number of Matches (**1**) and Non-matches (**0**) in the training table `FCC_ER_ML_TRAINING_DATA_NAME_BOOSTED`.

To populate data in the Name and Address tables, perform the following steps:

1. Log in to the Compliance Studio Schema.
2. Edit the table for name or address, `FCC_ER_ML_TRAINING_DATA_NAME_BOOSTED` or `FCC_ER_ML_TRAINING_DATA_ADDRESS_BOOSTED` respectively.
3. To populate the training data manually and arrive at a good ML Model, run the following SQL query to insert the records in the name/address table.

- For name:

  ```
  insert into FCC_ER_ML_TRAINING_DATA_NAME_BOOSTED (pairid, name1,
  name2, match, created, removed, source, edge_id)
  values (<number>, '<name1>', '<name2>', <Match type>, to_date('<Date
  and time>'), null, null, null);
  ```

- For address:

  ```
  insert into FCC_ER_ML_TRAINING_DATA_ADDRESS_BOOSTED (pairid,
  address1, address2, match, created, removed, source, edge_id)
  values (<number>, '<address1>', '<address2>', <Match type>,
  to_date('<Date and time>'), null, null, null);
  ```

Example:

```
insert into FCC_ER_ML_TRAINING_DATA_NAME_BOOSTED (pairid, name1, name2,
match, created, removed, source, edge_id)
values (1, 'Vitorino Mazzarella', 'Vitorin Mazarela', 1, to_date('13-11-
2000 12:00:00', 'dd-mm-yyyy hh24:mi:ss'), null, null, null);
```

The example Table 24 structure for name is as follows, similarly for address:

Table 24: Training Data

Paired	Name1	Name2	Match	Created	Removed	Source	EdgeID
1	NICHOLAS INC	NICK	1	15-02-2021	--	AUTOAPPROVED	841120030189
3	THOMAS JOY	JONS	0	26-12-2021	26-12-2021	MANUAL EDIT	1100322012647

Following are the parameter details:

- **pairid**: This unique id.
- **name1/address1**: It represents the source customer's name/address.
- **name2/address2**: It represents the target customer's name/address.
- **match**: It accepts the following values:
  - **1**: for matched records
  - **0**: for non-matched records
- **created**: When a model is updated, the training process selects only rows that were added only after the present model was created (updated), plus some older sampled training data. It is used for incremental updates

  Format: dd-mm-yyyy hh24:mi:ss.

- **removed**: For full retraining, we will pick all records that are not decommissioned (Removed is empty). The decommissioned records are kept for reference, so we can rebuild a model for any point in time, if necessary. We may remove decommissioned records after a period to keep the learning database a reasonable size.

- **Decommission old**: After training, you should be able to see the edges which are decommissioned with the current training date >12 and <36 months.

NOTE Ensure LAST_TRAINING_NAME_DATE and LAST_TRAINING_ADDRESS_DATE columns are updated with the date when incremental training is executed in table FCC_ER_LAST_BATCH_CONFIG.

- **source**: It accepts the following values:
  - **AUTOAPPROVED**: It indicates that name/address are matched. Names/addresses are considerably similar to get a high score based on the following formula in the table.
  - **MANUAL EDIT**: It indicates name/address pairs that are added by the user manually. Insert an equal number of records in the training table for Matches (**1**) and Non-matches (**0**) with the source as MANUAL EDIT.
  - **NEGATIVE EXAMPLES**: Negative examples are generated automatically to supplement the training set with a balanced number of matches and not-matches after the training based on the following formula in the table:
  - **Old Sample**: The user should be able to see the edges that are >11 Months and not decommissioned based on the following formula in the table.

NOTE The records will be generated for old samples and negative examples only when you execute the incremental training the `FCCM_Studio_ML_Model_Training.sh` job 2nd time.

- **edge_id**: Any unique number.

  Table 25 lists all ratios to train a Model for different sources.

Table 25: Ratio of the training model for different source

Source	Ratio
Auto approved	Recommended number of records should be 10% of total matches and non-matches provided manually.
Negative examples	The system will add the appropriate number of examples and equally distribute between positive and negative examples. Ratio: #(negative examples) = #(auto approved) + #(manual edit approvals) - #(manual edit rejection)
Old Sample	10 % (matched edges>last config date) + 10 % (not matched edges>last config date). pick the same number of edges randomly(equal number of matched and not matched) from edges that are <=last config date

4. To trigger the **Incremental Training API**, perform the following steps:

a. Navigate to the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/bin` directory.

b. Run the following command:

```
export FIC_DB_HOME=<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed /
ficdb
```

The following steps are applicable only to the SAML realm:

i. `NBExecutor.properties` file should contain UserName and base 64 Encoded password.

ii. Run the following command in order to populate User Name and Password,

```
./FCCM_Studio_Set_UserPass.sh --username "<Username that is used
for SAML>" --password "<Password that is used for SAML>"
```

Or

```
FCCM_Studio_Set_UserPass.sh -u "USERNAME" -p "PASSWORD"
```

iii. In `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/conf` directory, verify the `FCC_Studio_SecretKey.properties` file that is created. Also, the `NBExecutor.properties` file is updated.

The `NBExecutor.properties` file should look like the following:

```
saml=true

username=MMGUSER

password=x0oB0FwPPf4un+FQh6gQEw==

apiToken=eyJhbGciOiJSUzI1NiJ9.eyJ1c2VyIjoiQkRfVVNFUiJ9.Uykh5Uh-
k9EzfMpMeLJIF-
```

5. To set the Training timeout, perform the following:

a. Navigate to `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/entity-resolution/conf/application.yml.`

b. Set the `trainingTimeOut:<number of seconds>` based size of training data.

For example, to configure for 2 minutes

```
trainingTimeOut:120
```

By default, it is set to **60** seconds.

6. To import the ML training notebook to Compliance studio UI, perform the following:

a. Navigate to `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/bin`

b. Run the following batch:

```
./Import_Training_Models.sh
```

NOTE: Batch execution status always displays success in case of success or failure. After successful execution, the ML_Boosted Objective is available in UI. For more details, see **Frequently Asked Questions (FAQs) and Error Dictionary** in OFS Compliance Studio Installation Guide.

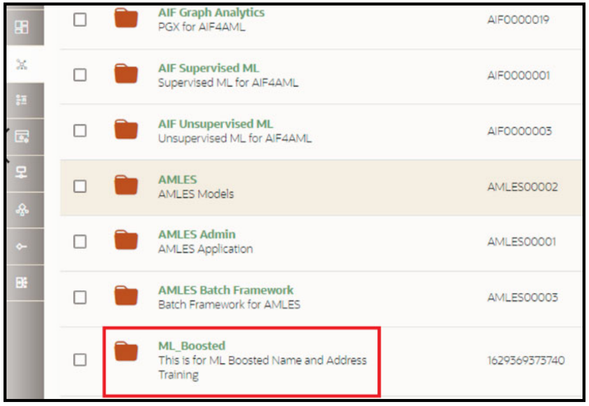
c. To verify the ML_Boosted notebook available in Compliance Studio UI:

i. Login to the Compliance Studio application.

ii. Launch CS production workspace.

iii. Click **Advanced Model Management** to display the Model Management window.

Figure 48: Model Management



□	AIF Graph Analytics PGX for AIF4AML	AIF0000019
□	AIF Supervised ML Supervised ML for AIF4AML	AIF0000001
□	AIF Unsupervised ML Unsupervised ML for AIF4AML	AIF0000003
□	AMLES AMLES Models	AMLES00002
□	AMLES Admin AMLES Application	AMLES00001
□	AMLES Batch Framework Batch Framework for AMLES	AMLES00003
□	ML_Boosted This is for ML Boosted Name and Address Training	1629369373740

7. Execute the following command:

```
./FCCM_Studio_ML_Model_Training.sh "<Type>" "<ModelId>"
"<ModelDescription>"
```

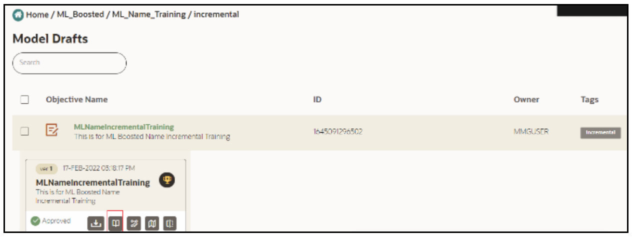
- Type: It should be either **name** or **address**
- ModelId: It should be any unique number.
- ModelDescription: It can be any string that describes the model.

NOTE Ensure that you do not provide spaces between the text for the ModelId and ModelDescription.

8. Verify the following to ensure the training is successful:

NameMatchingIncrementalTraining/AddressMatchingIncrementalTraining notebook is executed in Compliance Studio as shown in the following figure, and the final accuracy of the trained model is displayed in the "Publish new champion model" paragraph.

Figure 49: Model Drafts



Home / ML_Boosted / ML_Name_Training / incremental

Model Drafts

□	Objective Name	ID	Owner	Tags
□	MLNameIncrementalTraining This is for ML Boosted Name Incremental Training	1645091296502	MMGUSER	Incremental

ver 1 17-FEB-2022 03:18:17 PM
MLNameIncrementalTraining
This is for ML Boosted Name Incremental Training
Approved

- Based on the Model accuracy, a new record is inserted in the FCC_ER_ML_MODEL_NAME_BOOSTED table. The same table is used for both name and address training. The newly trained Model will be marked as the champion model

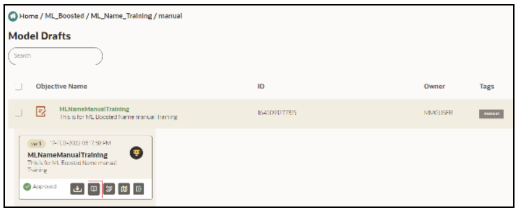
(`IS_CHAMPION` is **Y**) only if the accuracy is more than **0.77** and the accuracy of the current model is higher than the previous champion model.

NOTE

- A view, **FCC_ER_ML_TRAINING_DATA_TMP_VIEW,** will be created, and training data will be populated automatically after executing the `FCCM_Studio_ML_Model_Training.sh` job in the following ways:
  - Matches generated from the Similarity Edge job are greater than the **Auto** threshold set on **Rule Set** page. These matches are considered as **1**.
  - Negative example, OLD samples, Decommissioning records will be processed automatically. See formula table.
  - The model trained on training data is internally validated against the validation dataset available in **FCC_ER_ML_VALIDATION_DATA_NAME_-BOOSTED**. table. You need to update or populate the **FCC_ER_ML_VALIDATION_DATA_NAME_BOOSTED** table with the actual matching results for better validation.
  - The trained model on training data is internally validated against the validation dataset available in the FCC_ER_ML_VALIDATION_DATA_ADDRESS_BOOSTED table for address. You need to update or populate the **FCC_ER_ML_VALIDATION_DATA_ADDRESS_BOOSTED** table with the actual matching results for better validation.
- The same view, **FCC_ER_ML_TRAINING_DATA_TMP_VIEW,** can be replicated to another view for manual training (this is optional).

- **NameMatchingManualTraining/AddressMatchingManualTraining** notebook is executed in Compliance Studio as shown in the following figure. You can create a new view with training data or reuse replicated view (that is created during incremental training) for training data in this notebook and create a model. In order to execute the training, you have to provide the following inputs in this notebook:

Figure 50: Model Drafts



Home / ML_Boosted / ML_Name_Training / manual

Model Drafts

	Objective Name	ID	Owner	Tags
□	MLNameManualTraining This is for ML Boosted Name manual Training	1645091277515	MMGUSER	manual

ver 1 17-FEB-2022 03:17:58 PM

MLNameManualTraining

This is for ML Boosted Name manual Training

Approved

— **Database DSN alias** in the **Initialization** paragraph: Studio schema alias name that is provided.

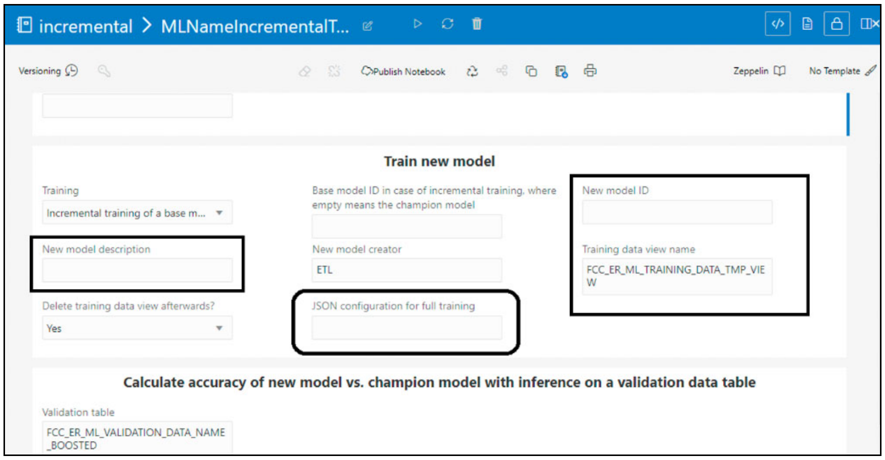
— The same view can be used for manual training

— **New model ID**, **New model description**, **and Training data view name** (the view you want to use to create a model with the training data) in the **Train new model** paragraph.

For example, the data that you use for manual training is recorded in the manual data table, and those validated data are put into a view (**FCC_ER_ML_TRAINING_DATA_TMP_VIEW**). Based on this view's accuracy, a new model is created and becomes the "Champion Model" because it has higher accuracy than the incremental training data.

The following figure illustrates the model training data:

Figure 51: Model training data



Train new model

Training: Incremental training of a base m...

Base model ID in case of incremental training, where empty means the champion model:

New model ID:

New model description:

New model creator: ETL

Training data view name: FCC_ER_ML_TRAINING_DATA_TMP_VIEW

Delete training data view afterwards?: Yes

JSON configuration for full training:

Calculate accuracy of new model vs. champion model with inference on a validation data table

Validation table: FCC_ER_ML_VALIDATION_DATA_NAME_BOOSTED

NOTE The field **JSON configuration for full training** is empty by default but can give JSON string full training.

9. The python processes are generated while executing incremental ML Training notebooks.

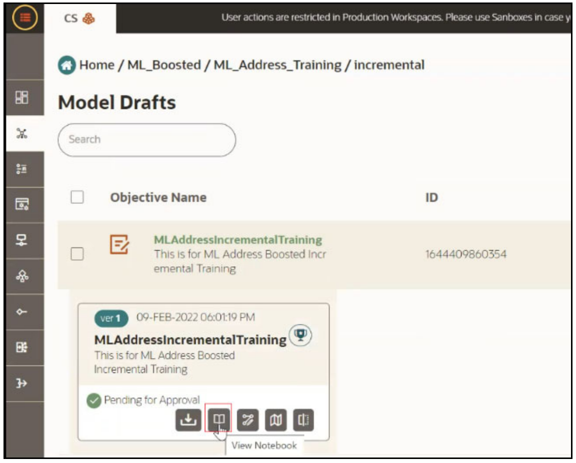
   After completing the execution of these ML Training notebooks, you can free up memory that is used by python processes.

   To free up memory, perform the following:

   a. Navigate to **Advanced Model Management** > **ML Boosted** > **ML_Address_Training** > **incremental**. The ML Address Incremental object is displayed.

      For Name, select the **ML_Name_Training.**

      Figure 52: ML Training Incremental object

      User actions are restricted in Production Workspaces. Please use Sandboxes in case y

      Home / ML_Boosted / ML_Address_Training / incremental

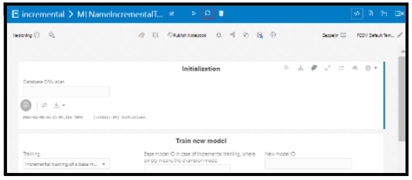
      Model Drafts

	Objective Name	ID
□	MLAddressIncrementalTraining This is for ML Address Boosted Incremental Training	1644409860354

      ver 1 09-FEB-2022 06:01:19 PM
      MLAddressIncrementalTraining
      This is for ML Address Boosted Incremental Training
      Pending for Approval

   b. On the **MLAddressIncrementalTraining** notebook, Click **View Notebook**. The notebook is displayed.

Figure 53: ML Address Incremental Training


incremental > MI NameIncrementalT...
Versioning
Publish Notebook
Zeppelin
FCCM Default Tem...
Initialization
Database DSN alias
Train new model
Base model ID in case of incremental training, where empty means the champion model
New model ID

Training: Incremental training of a base m...

c. Click to invalidate the session.

> **NOTE** If any of the Notebooks are no longer needed, you can invalidate the session of that notebook and free up the system resources.

Topics:

- Tables Used for Training the data for Name
- Tables Used for Training the data for Address

10.2.1 Tables Used for Training the data for Name

Table 26 that is used for training the data for address.

Table 26: Data Tables

Purpose	Data Tables
For Last Training Date or ETL Date	`FCC_ER_LAST_BATCH_CONFIG`
For Training Data Gathering	`FCC_ER_ML_TRAINING_DATA_NAME_BOOSTED`
For Validation Data table	`FCC_ER_ML_VALIDATION_DATA_NAME_BOOSTED`
Models Stored Table	`FCC_ER_ML_MODEL_NAME_BOOSTED`
View Name on top of Training Table	`FCC_ER_ML_TRAINING_DATA_TMP_VIEW`

10.2.2 Tables Used for Training the data for Address

Table 27 that is used for training the data for address.

Table 27: Data Tables

Purpose	Data Tables
For Last Training Date or ETL Date	`FCC_ER_LAST_BATCH_CONFIG`
For Training Data Gathering	FCC_ER_ML_TRAINING_DATA_ADDRESS_BOOSTED
For Validation Data table	FCC_ER_ML_VALIDATION_DATA_ADDRESS_BOOSTED
Models Stored Table	FCC_ER_ML_MODEL_NAME_BOOSTED **NOTE**: This tables stores both Name and Address.
View Name on top of Training Table	FCC_ER_ML_TRAINING_DATA_ADDRESS_TMP_VIEW

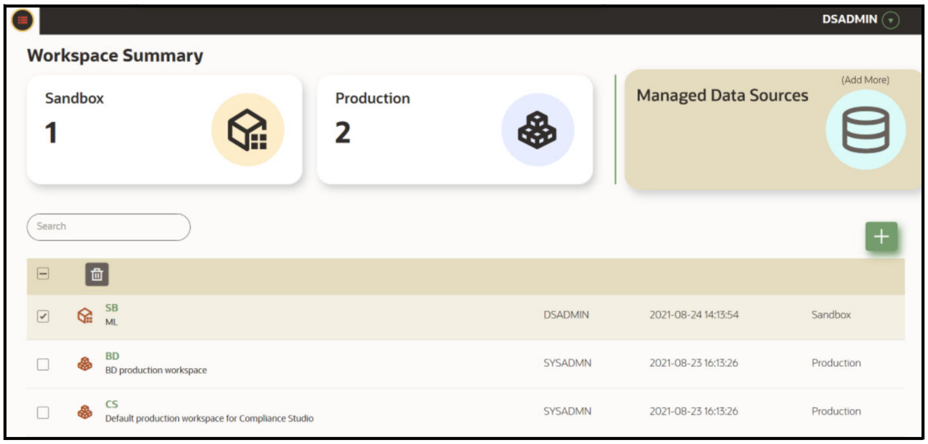
11 ML for AML (ML4AML)

Topics:

- Creating Data Source
- Creating a Sandbox Workspace
- Populating the Sandbox Workspace
- Importing Workspace Metadata for ML4AML
- Launch the Sandbox Workspace
- Model Groups
- Batch Framework
- Data Movement
- ECM Connector Batch

11.1 Creating Data Source

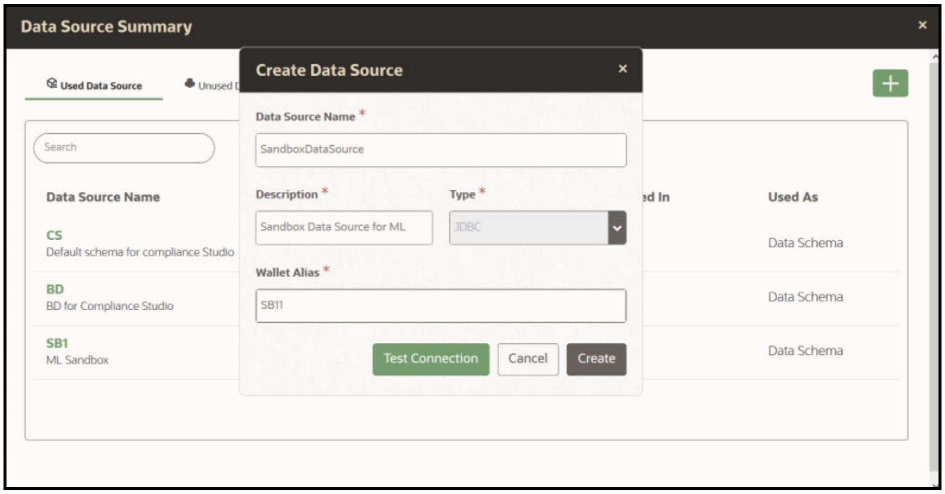
Figure 54: Workspace Summary



☑	SB ML	DSADMIN	2021-08-24 14:13:54	Sandbox
□	BD BD production workspace	SYSADMN	2021-08-23 16:13:26	Production
□	CS Default production workspace for Compliance Studio	SYSADMN	2021-08-23 16:13:26	Production

- Click on **Managed Data Sources** on Compliance Studio Home Page (Workspace Summary).
- Click on **+** button to create the data source for the sandbox workspace.
- Provide Data source details like Name & Description.
- **Wallet Alias**: Make sure wallet alias has been created/added for the schema and used as sandbox workspace.
- Refer to **Oracle Wallet documentation** to create/manage wallets.
- Refer to **Compliance Studio Installation Guide** to locate the wallet location.
- Click **Test Connection** to verify its connectivity.
- Click **Create** to create/add a new data source.
- This newly created data source will be used/selected while creating a sandbox workspace.

Figure 55: Data Source Summary



Data Source Name *: SandboxDataSource

Description *: Sandbox Data Source for ML

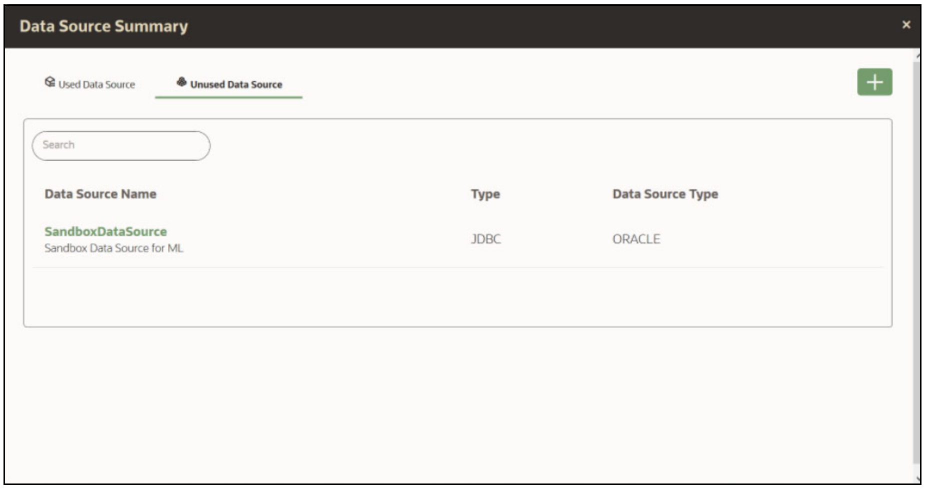
Type *: JDBC

Wallet Alias *: SB11

Data Source Name	ed In	Used As
CS Default schema for compliance Studio		Data Schema
BD BD for Compliance Studio		Data Schema
SB1 ML Sandbox		Data Schema

- Newly created data sources will be visible under **Unused data sources** until they create a workspace (Sandbox / Production).

Figure 56: Data Source Summary



Data Source Name	Type	Data Source Type
SandboxDataSource Sandbox Data Source for ML	JDBC	ORACLE

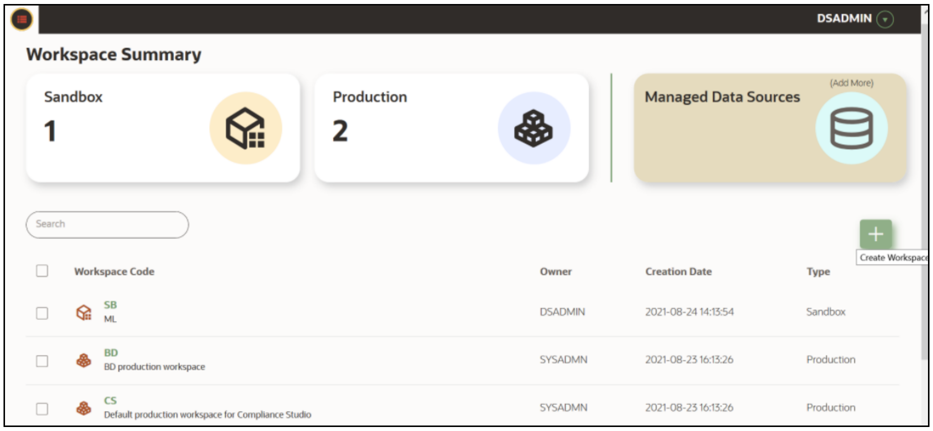
11.2 Creating a Sandbox Workspace

Topics:

- Basic Details
- Workspace Schema
- Data Sourcing
- Metadata Sourcing
- Validate Workspace
- Summary

After selecting Add **+** in the **Workspace Summary page (CS Home Page)**, the Workspace Creation window is displayed.

Figure 57: Workspace Summary page



Workspace Summary

Sandbox 1

Production 2

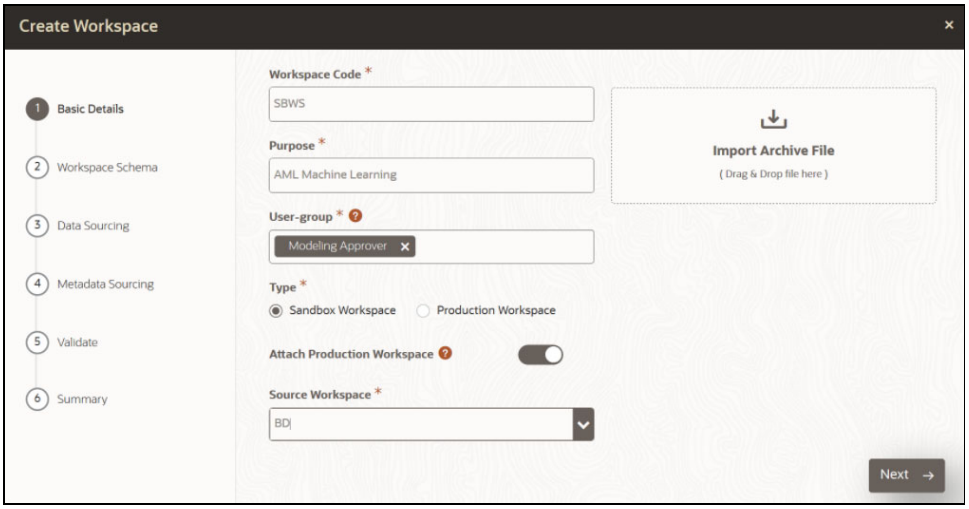
Managed Data Sources (Add More)

Workspace Code	Owner	Creation Date	Type
SB ML	DSADMIN	2021-08-24 14:13:54	Sandbox
BD BD production workspace	SYSADMN	2021-08-23 16:13:26	Production
CS Default production workspace for Compliance Studio	SYSADMN	2021-08-23 16:13:26	Production

11.2.1 Basic Details

- Provide the requested details for Workspace Code and Purpose.
- Select the type as **Sandbox Workspace**.
- **Enable** the **Attach Production Workspace** button.
- Choose **BD** as Source Workspace (Production workspace).

Figure 58: Create Workspace



Create Workspace

Workspace Code * SBWS

Purpose * AML Machine Learning

User-group * Modeling Approver ×

Type * ● Sandbox Workspace ○ Production Workspace

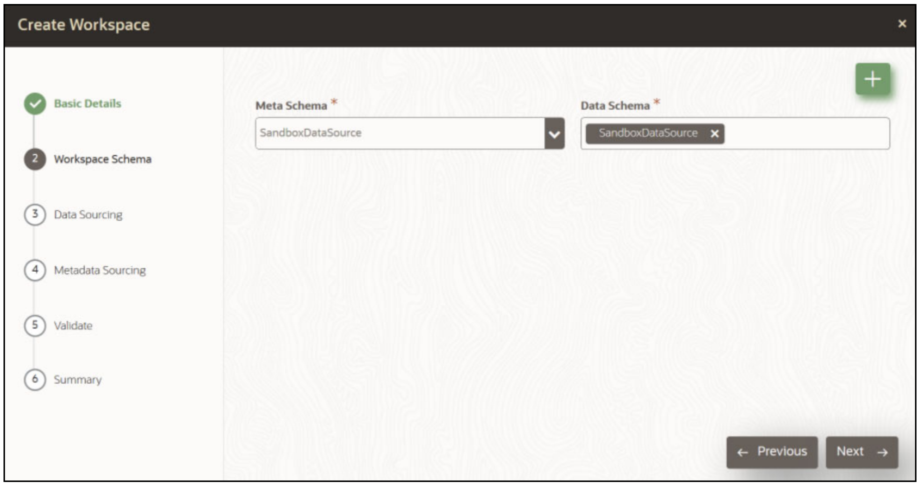
Attach Production Workspace (enabled)

Source Workspace * BD

11.2.2 Workspace Schema

- Choose a **newly created data source** (Refer to the previous section) as **Meta and Data Schema.**

Figure 59: Create Workspace



Meta Schema * SandboxDataSource

Data Schema * SandboxDataSource ×

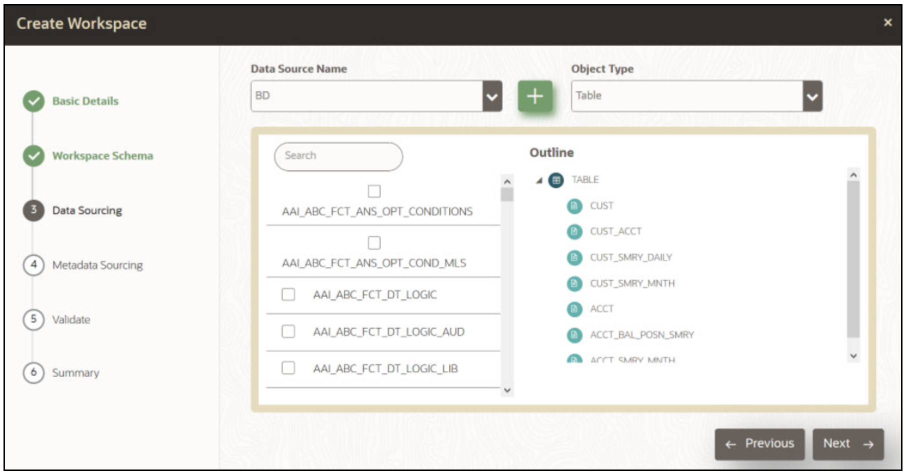
11.2.3 Data Sourcing

Select the following tables:

- CUST
- CUST_ACCT
- CUST_SMRY_DAILY
- CUST_SMRY_MNTH
- ACCT
- ACCT_BAL_POSN_SMRY
- ACCT_SMRY_MNTH
- ACCT_POSN
- CASH_TRXN
- WIRE_TRXN
- MI_TRXN
- BACK_OFFICE_TRXN
- TRADE
- TRADE_EXECUTION_EVENT
- SCRTY_MKT_DAILY
- SCRTY
- ORDR
- EXECUTION
- STDO_ERROR_DETAILS
- FCC_AM_EVENT_ENTITY_MAP
- FCC_AM_EVENTS

- FCC_AM_EVENT_BINDING
- FCC_AM_EVENT_DETAILS

Figure 60: Create Workspace



Data Source Name: BD

Object Type: Table

- □ AAI_ABC_FCT_ANS_OPT_CONDITIONS
- □ AAI_ABC_FCT_ANS_OPT_COND_MLS
- □ AAI_ABC_FCT_DT_LOGIC
- □ AAI_ABC_FCT_DT_LOGIC_AUD
- □ AAI_ABC_FCT_DT_LOGIC_LIB

TABLE
- CUST
- CUST_ACCT
- CUST_SMRY_DAILY
- CUST_SMRY_MNTH
- ACCT
- ACCT_BAL_POSN_SMRY
- ACCT_SMRY_MNTH

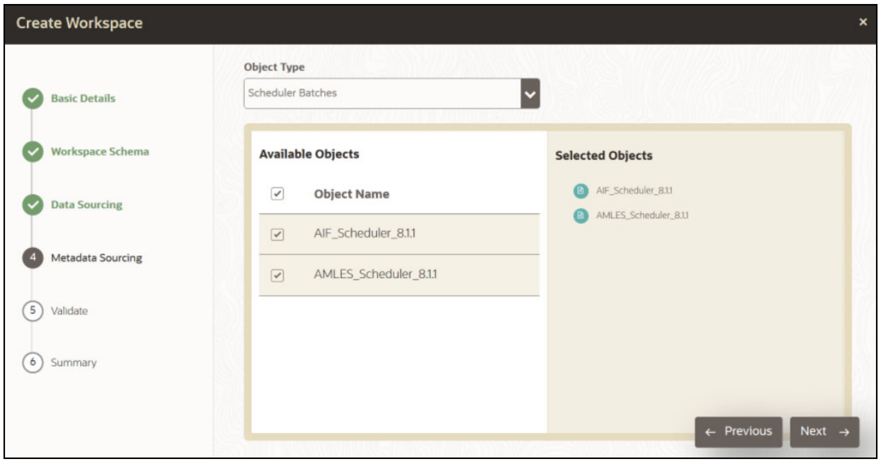
11.2.4 Metadata Sourcing

- Select **Scheduler Batches** from the Object Type drop-down menu.
- Choose schedulers:

  For example,

  - **AIF_Scheduler_8.1.1**
  - **AMLES_Scheduler_8.1.1**

Figure 61: Create Workspace



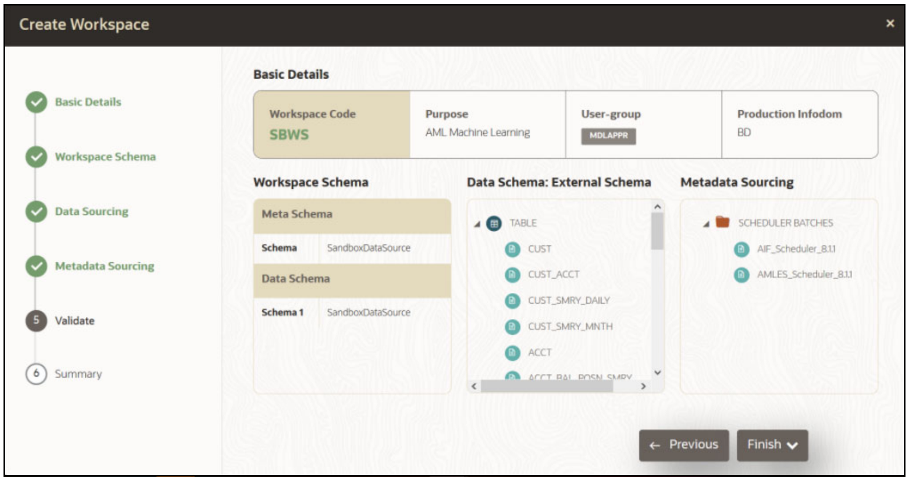
Object Type: Scheduler Batches

☑	Object Name
☑	AIF_Scheduler_8.1.1
☑	AMLES_Scheduler_8.1.1

Selected Objects:
- AIF_Scheduler_8.1.1
- AMLES_Scheduler_8.1.1

11.2.5 Validate Workspace

Figure 62: Validate Workspace



Create Workspace

Basic Details

Workspace Code	Purpose	User-group	Production Infodom
SBWS	AML Machine Learning	MDLAPPR	BD

Workspace Schema

Meta Schema

Schema	SandboxDataSource

Data Schema

Schema 1	SandboxDataSource

Data Schema: External Schema

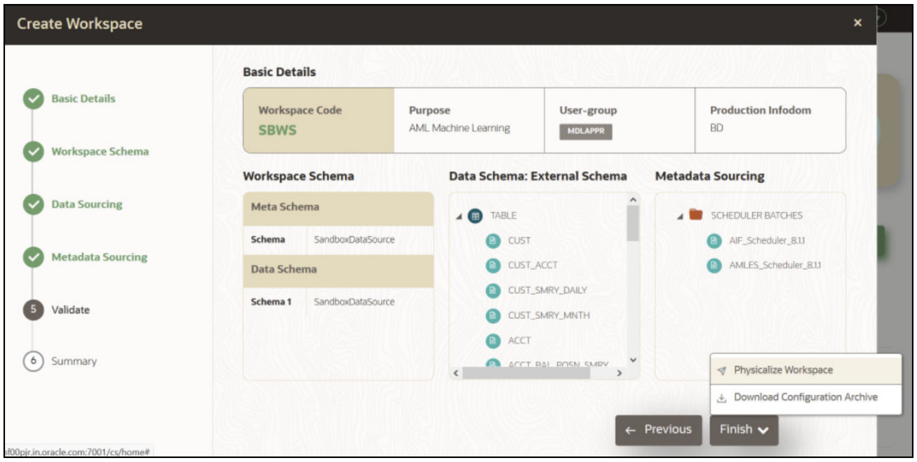
- TABLE
  - CUST
  - CUST_ACCT
  - CUST_SMRY_DAILY
  - CUST_SMRY_MNTH
  - ACCT
  - ACCT_BAL_POSN_SMRY

Metadata Sourcing

- SCHEDULER BATCHES
  - AIF_Scheduler_8.1.1
  - AMLES_Scheduler_8.1.1

- Click **Finish**.
- Choose **Physicalize Workspace**.

Figure 63: Create Workspace



Create Workspace

Basic Details

Workspace Code	Purpose	User-group	Production Infodom
SBWS	AML Machine Learning	MDLAPPR	BD

Workspace Schema

Meta Schema

Schema	SandboxDataSource

Data Schema

Schema 1	SandboxDataSource

Data Schema: External Schema

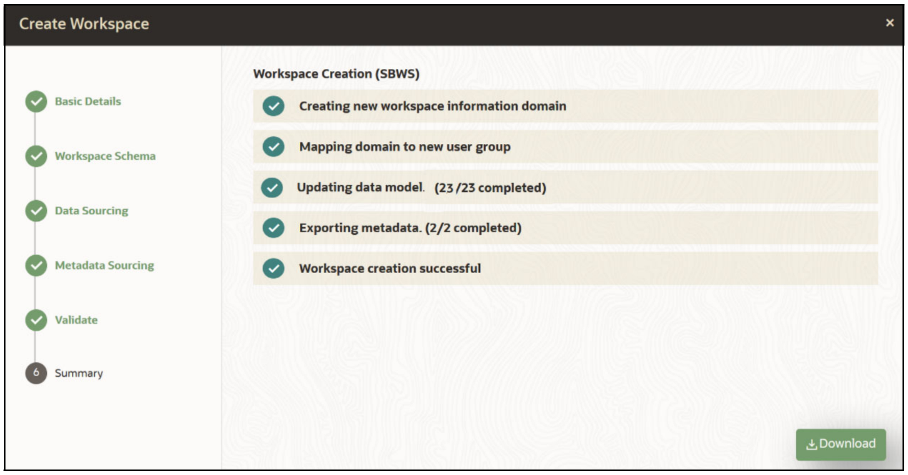
- TABLE
  - CUST
  - CUST_ACCT
  - CUST_SMRY_DAILY
  - CUST_SMRY_MNTH
  - ACCT
  - ACCT_BAL_POSN_SMRY

Metadata Sourcing

- SCHEDULER BATCHES
  - AIF_Scheduler_8.1.1
  - AMLES_Scheduler_8.1.1

11.2.6 Summary

Figure 64: Create Workspace



Create Workspace

- ✓ Basic Details
- ✓ Workspace Schema
- ✓ Data Sourcing
- ✓ Metadata Sourcing
- ✓ Validate
- 6 Summary

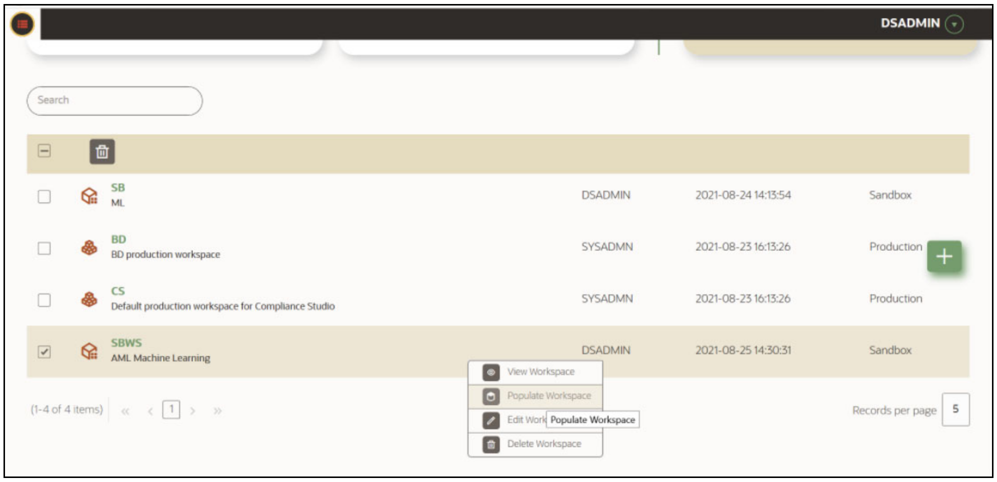
Workspace Creation (SBWS)

- ✓ Creating new workspace information domain
- ✓ Mapping domain to new user group
- ✓ Updating data model. (23 /23 completed)
- ✓ Exporting metadata. (2/2 completed)
- ✓ Workspace creation successful

11.3 Populating the Sandbox Workspace

- From the workspace summary screen, choose to **populate sandbox** for the newly created sandbox.

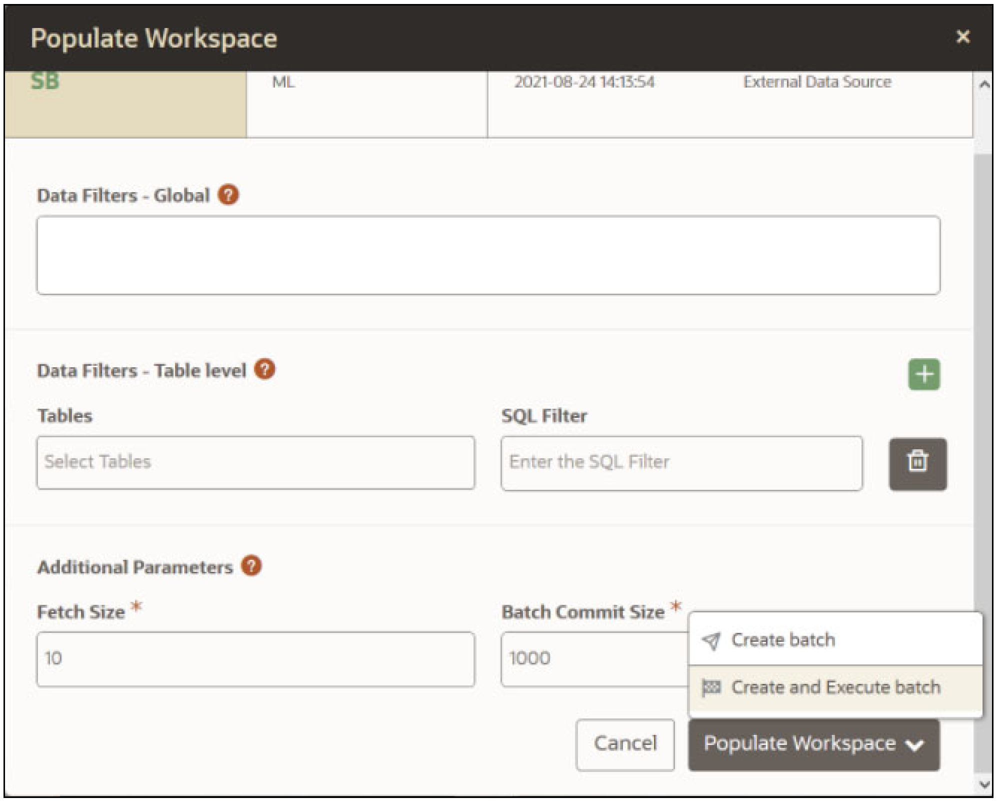
Figure 65: Sandbox Workspace



□	SB ML	DSADMIN	2021-08-24 14:13:54	Sandbox
□	BD BD production workspace	SYSADMN	2021-08-23 16:13:26	Production
□	CS Default production workspace for Compliance Studio	SYSADMN	2021-08-23 16:13:26	Production
☑	SBWS AML Machine Learning	DSADMIN	2021-08-25 14:30:31	Sandbox

- Choose **Create & Execute** batch option.

Figure 66: Populate Workspace



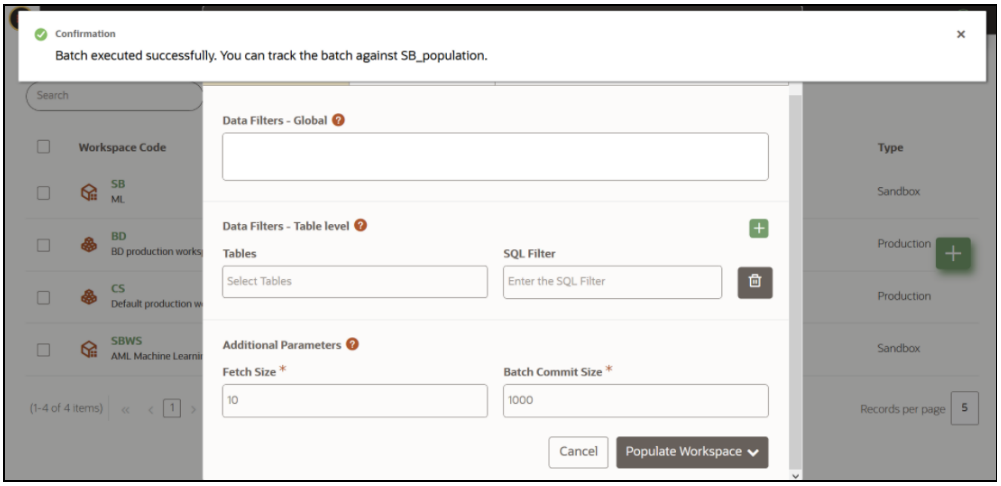
SB | ML | 2021-08-24 14:13:54 | External Data Source

Fetch Size * 10

Batch Commit Size * 1000

- Shows a Successful message on successfully triggering the **Workspace Data Population**.

Figure 67: Workspace Data Population


Confirmation

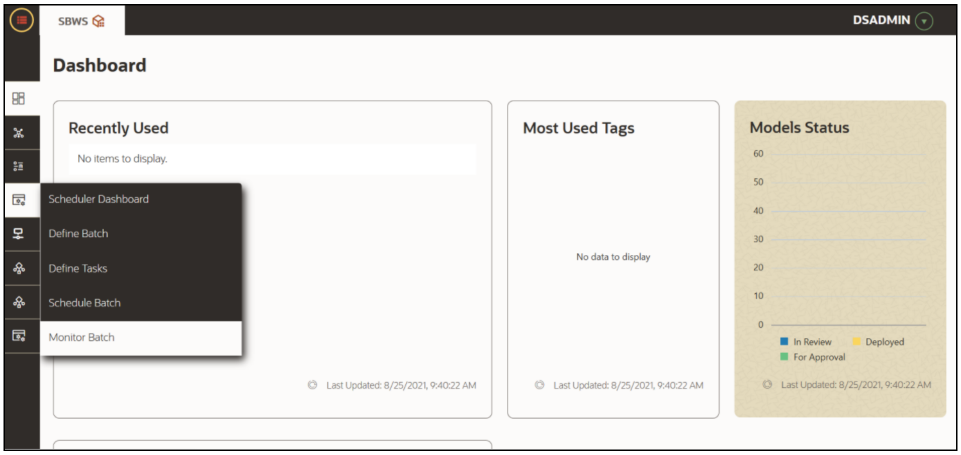
Batch executed successfully. You can track the batch against SB_population.

Fetch Size * 10

Batch Commit Size * 1000

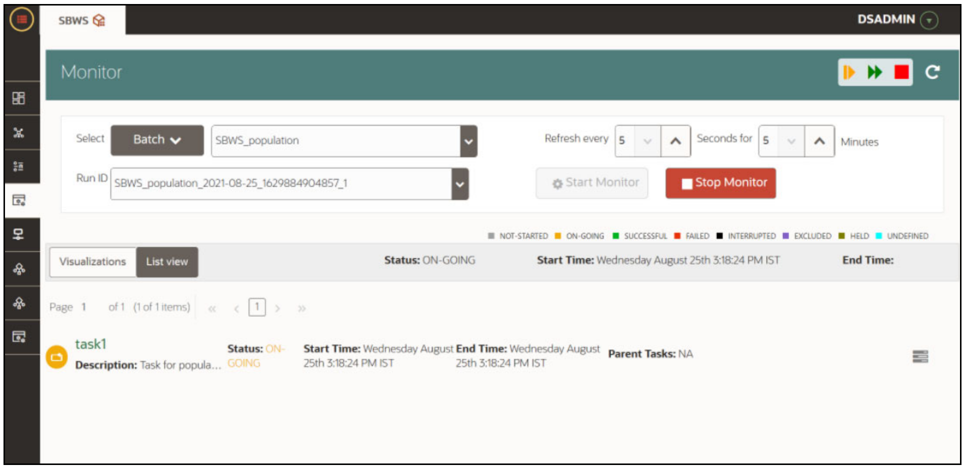
- Monitor the status **of Sandbox Workspace Population.**
  - Launch the sandbox workspace using the **launch** button.
  - Choose **Monitor Batch** Option under **Scheduler Dashboard** from the Menu.

Figure 68: Dashboard



- Select/Provide the Batch ID details using the drop-down to see the **status**.

Figure 69: Status



Select: Batch SBWS_population

Run ID: SBWS_population_2021-08-25_1629884904857_1

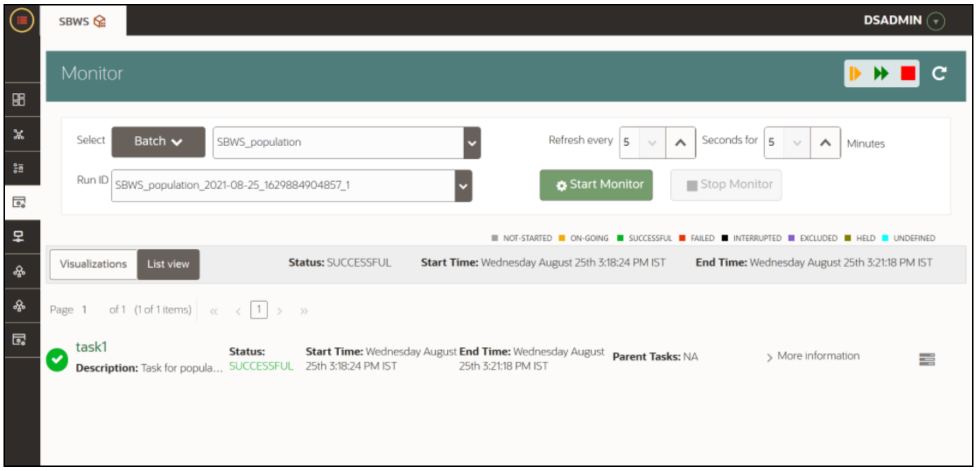
Refresh every 5 Seconds for 5 Minutes

Status: ON-GOING **Start Time:** Wednesday August 25th 3:18:24 PM IST **End Time:**

Page 1 of 1 (1 of 1 items)

task1 **Description:** Task for popula... **Status:** ON-GOING **Start Time:** Wednesday August 25th 3:18:24 PM IST **End Time:** Wednesday August 25th 3:18:24 PM IST **Parent Tasks:** NA

Figure 70: Status



Select: Batch SBWS_population

Run ID: SBWS_population_2021-08-25_1629884904857_1

Refresh every 5 Seconds for 5 Minutes

Status: SUCCESSFUL **Start Time:** Wednesday August 25th 3:18:24 PM IST **End Time:** Wednesday August 25th 3:21:18 PM IST

Page 1 of 1 (1 of 1 items)

task1 **Description:** Task for popula... **Status:** SUCCESSFUL **Start Time:** Wednesday August 25th 3:18:24 PM IST **End Time:** Wednesday August 25th 3:21:18 PM IST **Parent Tasks:** NA

11.4 Importing Workspace Metadata for ML4AML

- Login to Compliance Studio installed UNIX Machine.
- Navigate to <Compliance_Studio_HOME>/deployed/ml4aml/bin
- Execute following commands once against **Production workspace**
  - `./importNotebooksAIF.sh -w BD`
  - `./importNotebooksAMLES.sh -w BD`

> **NOTE** The above two commands do not contain any placeholders and can be executed without any modifications.

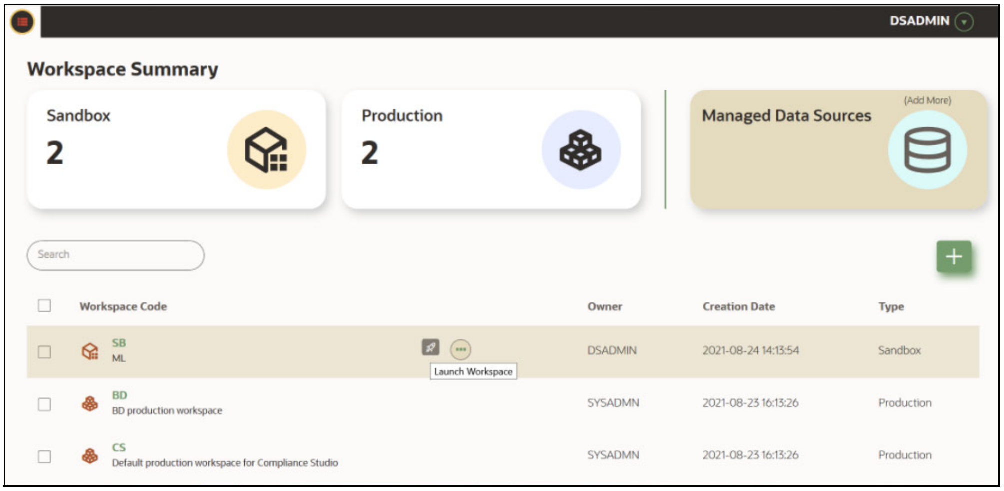
- Execute the following UNIX commands once against **Sandbox workspace**
  - `./sandbox.sh -w <sandbox_wallet_alias>`
  - `./importNotebooksAIF.sh -w <sandbox_workspace_code>`
  - `./importNotebooksAMLES.sh -w <sandbox_workspace_code>`

> **NOTE** **sandbox_wallet_alias** and **sandbox_workspace_code** are the place holders to be replaced with actual values used to create sandbox workspace.

11.5 Launch the Sandbox Workspace

- Click **Launch** icon from the workspace summary screen for launching the sandbox.

Figure 71: Sandbox Workspace

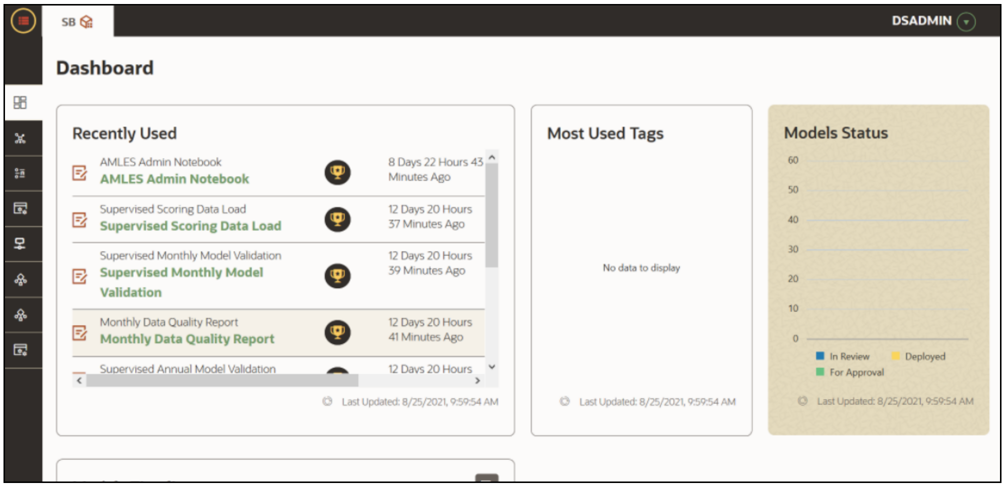


☐	Workspace Code	Owner	Creation Date	Type
☐	SB ML	DSADMIN	2021-08-24 14:13:54	Sandbox
☐	BD BD production workspace	SYSADMN	2021-08-23 16:13:26	Production
☐	CS Default production workspace for Compliance Studio	SYSADMN	2021-08-23 16:13:26	Production

- On launching, the workspace will land the user in a Dashboard with the following options:
  - Advanced Model Management
  - Model Actions
  - Scheduler Dashboard
  - Model Audit

  - Data Studio Options
  - Rule-set Details
  - Manual Decisioning
- Choose Advanced Model Management to start with ML

Figure 72: Dashboard



Dashboard

Recently Used

AMLES Admin Notebook
AMLES Admin Notebook — 8 Days 22 Hours 43 Minutes Ago

Supervised Scoring Data Load
Supervised Scoring Data Load — 12 Days 20 Hours 37 Minutes Ago

Supervised Monthly Model Validation
Supervised Monthly Model Validation — 12 Days 20 Hours 39 Minutes Ago

Monthly Data Quality Report
Monthly Data Quality Report — 12 Days 20 Hours 41 Minutes Ago

Supervised Annual Model Validation — 12 Days 20 Hours

Last Updated: 8/25/2021, 9:59:54 AM

Most Used Tags

No data to display

Last Updated: 8/25/2021, 9:59:54 AM

Models Status

Last Updated: 8/25/2021, 9:59:54 AM

11.6 Model Groups

OFS AIF4AML is an application that provides foundational building blocks to train, deploy and monitor models tailored to address specific use cases relevant to the AML domain. It has a pre-defined set of transformations and over 300 attributes to help expedite the model development process.

OFS AIF4AML uses the Model Management and Governance (MMG) application to manage the various stages of the modeling lifecycle, such as sandbox creation, deployment to production, and ongoing monitoring.

Topics:

- Obtain the SAR Information
- Configure Investigation Guidance
- Model Group at Account and Customer Levels
- Admin Activity

11.6.1 Obtain the SAR Information

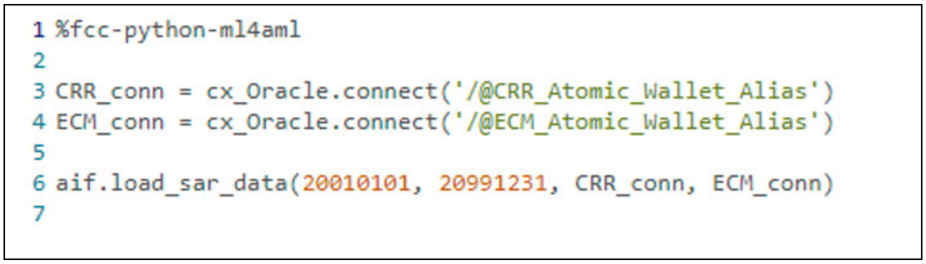
11.6.1.1 Populate Investigated Entity Details

11.6.1.1.1 Obtain the SAR from CRR/ECM

Use `aif.load_sar_data ()` API to load the Suspicious Activity Report (SAR) entities details from the Compliance Regulatory Reporting (CRR) application and Non-SAR entities from ECM into AIF.

The data will be loaded into the AIF `table aif_investigated_entity` table.

Figure 73: Aif Load SAR Data



```
%fcc-python-ml4aml

CRR_conn = cx_Oracle.connect('/@CRR_Atomic_Wallet_Alias')
ECM_conn = cx_Oracle.connect('/@ECM_Atomic_Wallet_Alias')

aif.load_sar_data(20010101, 20991231, CRR_conn, ECM_conn)

```

The following parameters are the input value for the paragraph:

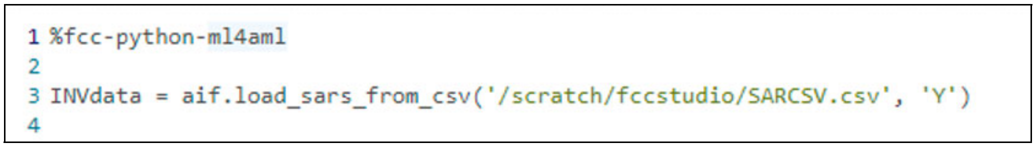
- **from_date**: From date range in **YYYYMMDD** format for SAR/Alert creation date.
- **to_date**: To date range in **YYYYMMDD** format for SAR/Alert creation date.
- **CRR_conn**: CRR Connection object.
- **ECM_conn**: ECM Connection object.

> **NOTE**
> - Register Oracle wallet entries/aliases for CRR & ECM Atomic schema to connect within Compliance Studio.
> - Use the aliases mentioned here to create/register entries. If aliases are being created with some other name, use them accordingly in the Admin Notebook.

11.6.1.1.2 Obtain the SAR from the CSV file

Use `aif.load_sars_from_csv()` API to load the SAR and Non-SAR entities into a CSV file.

Figure 74: Aif Load Sars from CSV



```
%fcc-python-ml4aml

INVdata = aif.load_sars_from_csv('/scratch/fccstudio/SARCSV.csv', 'Y')

```

The following parameters are the input value for the paragraph:

- **filename**: Complete path of the CSV file.
- **headerIncluded**: This parameter has two options: Y or N. If the file has data with the header, then Y or N.

> **NOTE**
> - The date should be in YYYYMMDD HH24:MI:SS format.
> - Records should be comma-separated (CSV).

Ensure that the following columns are available in the CSV files with the required values:

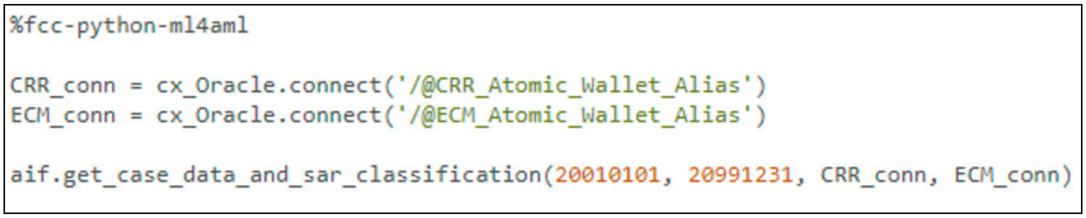
- **ENTITY_ID**: Customer Id or Account Id
- **SUSPICIOUS_FLAG**: This parameter has two options: Y or N. If E-file for Regulatory body has been sent for Customer or Account, then Y or N.

- **ALERT_DATE**: SAR/EVENT generated to date from Customers and Accounts
- **CREATED_ON**: CSV file creation date
- **CREATED_BY**: CSV file created by
- **UPDATED_ON**: CSV file updated date
- **UPDATED_BY**: CSV file updated by
- **LABELLED_SCENARIO**: This value has the following options:
  - **CUST**: For customer-level SAR
  - **ACCT**: For account level SAR
- **ENTITY_CD**: This value has the following options:
  - If entity type is customer
  - If entity type is the account

11.6.1.2 Obtain the SAR classification from the CRR database

The `aif.get_case_data_and_sar_classification()` API gets SAR classification from CRR schema, merge with entity ID (Customer ID) in ECM, and stores as metadata in AIF schema table, `aif_case_information`.

Figure 75: Aif Get Case Data



```
%fcc-python-ml4aml

CRR_conn = cx_Oracle.connect('/@CRR_Atomic_Wallet_Alias')
ECM_conn = cx_Oracle.connect('/@ECM_Atomic_Wallet_Alias')

aif.get_case_data_and_sar_classification(20010101, 20991231, CRR_conn, ECM_conn)
```

The `aif_case_information` table columns are as follows:

- ENTITY_ID
- CASE_ID
- SAR_CLASSIFICATION
- FILING_AM
- CONTINUING_SAR
- FILING_DATE

The following parameters are the input value for the paragraph:

- **from_date**: From date range in **YYYYMMDD** format.
- **to_date**: To date range in **YYYYMMDD** format.
- **CRR_conn**: CRR Connection object.
- **ECM_conn**: ECM Connection object.
- **AIF_conn:** AIF Connection object.

Format: `cx_Oracle.connect(<db_user/db_password@tns>)`

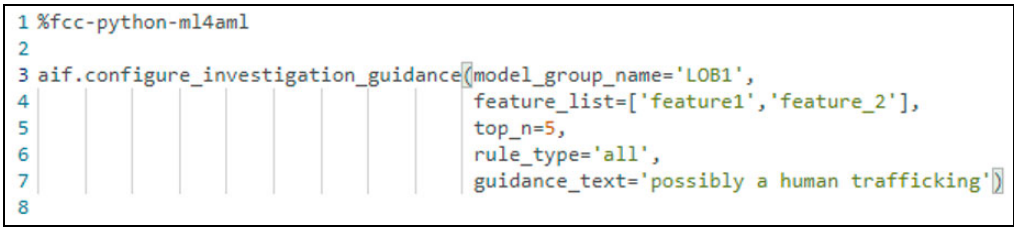
On successful execution of the paragraph, the details will be loaded in the `aif_case_information` table.

> **NOTE**
> - Register Oracle wallet entries/aliases for CRR & ECM Atomic schema to connect within Compliance Studio.
> - Use the aliases mentioned here to create/register entries. If aliases are being created with some other name, use them accordingly in the Admin Notebook.

11.6.2 Configure Investigation Guidance

Use `aif.configure_investigation_guidance()` API to load investigation guidance data in the `aif_investigation_guidance` table.

Figure 76: Configure Investigation Guidance



```
%fcc-python-ml4aml

aif.configure_investigation_guidance(model_group_name='LOB1',
                                     feature_list=['feature1','feature_2'],
                                     top_n=5,
                                     rule_type='all',
                                     guidance_text='possibly a human trafficking')

```

The `aif_investigation_guidance` table columns are as follows:

- V_MODEL_GROUP
- V_FEATURES
- TOP_N
- RULE_TYPE
- V_GUIDANCE_TEXT

The following parameters are the input value for the paragraph:

- **model_group_name**: Model group name for which we need to configure the investigation guidance.
- **feature_list**: The set of model features to be configured for investigation guidance.

  For example, ['feature1', 'feature2']
- **top_n**: The top `N` contributing features to be searched in the Model to consider for investigation guidance. The default value is **10**.
- **rule_type**: Consider feature(s) provided in the **feature_list** to be matched in model features.
  - **any**: Any one of the features in the **feature_list** will be matched with **top_n** contributing model features.
  - **all**: All of the features in the **feature_list** will be matched with **top_n** contributing model features.

- **guidance_text**: It provides the Investigation guidance for the following parameters:
  - Model group name
  - Feature list
  - Top N features

11.6.2.1 Output

The successful message is returned on successfully adding the top N features and Guided Text.

Returns error message if failed.

11.6.3 Model Group at Account and Customer Levels

The following metadata is used to create model groups:

- **Account Type1 Code**: Client-specified account type classification for the usage of this account.
- **Account Type2 Code**: Client-specified account type classification for the usage of this account.
- **Business Domain or Domains**: An account or customer (for example, institutional brokerage or retail brokerage).
- **Customer Type Code**: When a customer is involved in the execution, identify the type of customer.
- **Jurisdiction Code**: For an account or customer (for example, Americas, Europe, Middle East & Africa, India, and United States).
- **Account Status**: Account status (active, closed, and inactive).

Execute the following paragraph to view the metadata for the model groups:

```
%fcc-python-ml4aml

metadata_df = aif.show_metadata_for_model_group_creation()

z.show( metadata_df )
```

The output appears as shown in the Table 28.

Table 28: Output Data for Model Groups

ENTI- TY_NAME	ATTRIBUTE_NAME	ATTRIBUTE_VALUE
Customer/ Account	Business Domain (or Domains)	Asset Management, Corporate or Wholesale Banking, Employee Information, General, Institutional Broker-Dealer, Other values as specified by the client, Retail Banking, Retail Brokerage, or Private Client.
Customer	Customer Type	Financial Institution, Individual, Other Organization.
Customer/ Account	Jurisdiction Code	Americas, Europe, Middle East & Africa, India, United States.
Account	Account Type1 Code	Checking, Credit Card, Health Savings, Insurance Policy, Investment, Loan, Money Market, Other values as specified by the client, Others, Retirement, Savings, Stored Value Card, Term/Time/Certificate of Deposit.

Table 28: Output Data for Model Groups

ENTITY_NAME	ATTRIBUTE_NAME	ATTRIBUTE_VALUE
Account	Account Type2 Code	Checking, Credit Card, Health Savings, Insurance Policy, Investment, Loan, Money Market, Other values as specified by the client, Others, Retirement, Savings, Stored Value Card, Term/Time/Certificate of Deposit.

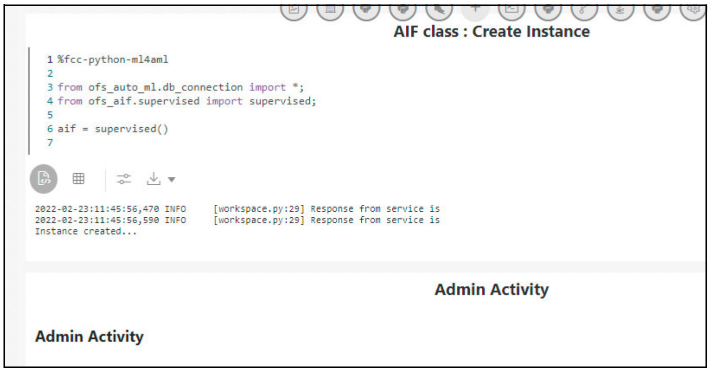
11.6.4 Admin Activity

11.6.4.1 Load the AIF Python Library

Execute the following instructions in the Notebook to load the AIF4AML library:

```
%fcc-python-ml4aml
import ofs_aif.supervised
```

Figure 77: AIF Admin



AIF class : Create Instance

```
%fcc-python-ml4aml

from ofs_auto_ml.db_connection import *;
from ofs_aif.supervised import supervised;

aif = supervised()

```

```
2022-02-23:11:45:56,470 INFO     [workspace.py:29] Response from service is
2022-02-23:11:45:56,590 INFO     [workspace.py:29] Response from service is
Instance created...
```

Admin Activity

Admin Activity

11.6.4.2 Metadata to Create Model Group(s)

A model group is used to define the Line Of Business (LOB) of a model group. Six variables are provided in the model group, and the LOB value can be found in these variables. The model group can be used at the account and customer levels.

The following metadata is used to create model groups:

- **Account Type1 Code**: Client-specified account type classification for the usage of this account.
- **Account Type2 Code**: Client-specified account type classification for the usage of this account.
- **Business Domain(s)**: An account or customer (for example, institutional brokerage or retail brokerage).
- **Customer Type Code**: When a customer is involved in the execution, identify the type of customer.
- **Jurisdiction Code**: For an account or customer (for example, Americas, Europe, Middle East & Africa, India, and United States).
- **Account Status**: Account status (active, closed, and inactive).

Use the `aif.show_metadata_for_model_group_creation` API to view the metadata, which you can use to create model groups.

Execute the following paragraph to view the metadata for the model groups:

```
%fcc-python-ml4aml
 metadata_df = aif.show_metadata_for_model_group_creation()
 z.show( metadata_df )
```

The output shows the default account and customer-level attributes enabled in the Table 29.

Table 29: Output Data for Model Groups

ENTITY_NAME	ATTRIBUTE_NAME	ATTRIBUTE_VALUE
Customer/Account	Business Domain(s)	Asset Management, Corporate or Wholesale Banking, Employee Information, General, Institutional Broker-Dealer, Other values as specified by the client, Retail Banking, Retail Brokerage, or Private Client.
Customer	Customer Type	Financial Institution, Individual, Other Organization.
Customer/Account	Jurisdiction Code	Americas, Europe, Middle East & Africa, India, United States.
Account	Account Type1 Code	Checking, Credit Card, Health Savings, Insurance Policy, Investment, Loan, Money Market, Other values as specified by the client, Others, Retirement, Savings, Stored Value Card, Term/Time/Certificate of Deposit.
Account	Account Type2 Code	Checking, Credit Card, Health Savings, Insurance Policy, Investment, Loan, Money Market, Other values as specified by the client, Others, Retirement, Savings, Stored Value Card, Term/Time/Certificate of Deposit.

11.6.4.3 Create the Input Dataframe for Model Groups

Create the Input Dataframe as shown in the following example:

```
%fcc-python-ml4aml

pdf = pd.DataFrame(
{'MODEL_GROUP_NAME'     : ["LOB13","LOB13"],
  'ENTITY_NAME'         : ["Account", "Account"],
   ATTRIBUTE_NAME'      : ["Business Domain(s)","Jurisdiction Code"],
  'ATTRIBUTE_VALUE'     : ["General","Europe Middle East & Africa"],
  'LABEL_FILTER'        : ["ACCT","ACCT"],
  'FEATURE_TYPE_FILTER' :
["CASH_TRXN,WIRE_TRXN,MI_TRXN","CASH_TRXN,WIRE_TRXN,MI_TRXN"]
                     })
```

```
z.show( pdf )
```

- **MODEL_GROUP_NAME**: The administrator-defined unique identifier for the model group. Only alphanumeric characters underscore, hyphens, and space are the special characters allowed.

- **ENTITY_NAME**: Logical Entity Name as displayed in the metadata section.
- **ATTRIBUTE_NAME**: Logical Attribute Name as displayed in the metadata section.
- **ATTRIBUTE_VALUE**: Logical Attribute Value as displayed in the metadata section.

11.6.4.3.1 Vertical and Horizontal Filters

The following filters are used as input data frames for model group creation:

- **LABEL_FILTER**: Use this filter to identify entities and labels from the table `AIF_INVESTIGATED_ENTITY`. It is a model group creation parameter that is mapped to the LABELLED_SCENARIO column in the `AIF_INVESTIGATED_ENTITY` table.
  - For **Unsupervised**, LABEL_FILTER to be passed as **UNSUPERVISED**
  - Foe **AMLES**, LABEL_FILTER to be passed as **AMLES**
- **FEATURE_TYPE_FILTER**: Use this filter to identify the features required for the model group. It is a model group creation parameter that is mapped to the ATTRIBUTE_NM column in the `aif_vertical_filter_lookup` table. Options include:
  - CASH_TRXN: Features specific to Cash Transactions
  - WIRE_TRXN: Features specific to Wire Transactions
  - MI_TRXN: Features specific to Monitory Instrument
  - TRADE: Features specific to Trading
  - BACK_OFFICE_TRXN: Features specific to Back-office Transactions

NOTE

- A vertical filter (FEATURE_TYPE_FILTER) is applicable only for supervised model groups.
- You can provide the list of features in the FEATURE_TYPE_FILTER that must be used while creating the supervised model group in the Admin Notebook.
- By default, it considers all features in the filter.
- In the case of Unsupervised, this is not applicable.

Any above combination such as comma (,) separated CASH_TRXN, MI_TRXN, or MI_TRXN, and CASH_TRXN, WIRE_TRXN is also allowed. The FEATURE_TYPE_FILTER helps to reduce the memory requirement at the model group level, so ensure that you optimize the storage by choosing only the required features.

- **Table AIF_VERTICAL_FILTER_LOOKUP**: Use this filter as a lookup table for feature list to feature type.

Execute the following paragraph to view data for the filters:

```
%fcc-python-ml4aml


pdf = pd.DataFrame(
{'MODEL_GROUP_NAME'     : ["LOB13","LOB13"],
  'ENTITY_NAME'         : ["Account", "Account"],
   ATTRIBUTE_NAME'      : ["Business Domain(s)","Jurisdiction Code"],
  'ATTRIBUTE_VALUE'     : ["General","Europe Middle East & Africa"],
  'LABEL_FILTER'        : ["ACCT","ACCT"],
  'FEATURE_TYPE_FILTER' :
["CASH_TRXN,WIRE_TRXN,MI_TRXN","CASH_TRXN,WIRE_TRXN,MI_TRXN"]
                     })
z.show( pdf )
```

The output appears as shown in the Table 30.

Table 30: Output Data for Filters

MOD-EL_GROUP _NAME	ENTITY _NAME	ATTRIBUTE _NAME	ATTRIBUTE _VALUE	LABEL_FIL-TER	FEATURE_-TYPE _FILTER
LOB13	Account	Business Domain(s)	General	ACCT	CASH_TRXN, WIRE_TRXN, MI_TRXN
LOB13	Account	Jurisdiction Code	Europe Middle East & Africa	ACCT	CASH_TRXN, WIRE_TRXN, MI_TRXN

11.6.4.4 Add Model Groups

Use the `aif.add_model_groups ()` API to view the list of available model groups.

The following is the input value for the paragraph:

meta_data_df: This is the input pandas data frame formed using the available metadata.

Execute the following paragraph to add Model Group(s):

```
%fcc-python-ml4aml
aif.add_model_groups(pdf)
```

The preceding code returns a confirmation message on successfully adding model groups or error messages for failures.

11.6.4.5 Import User Model Templates

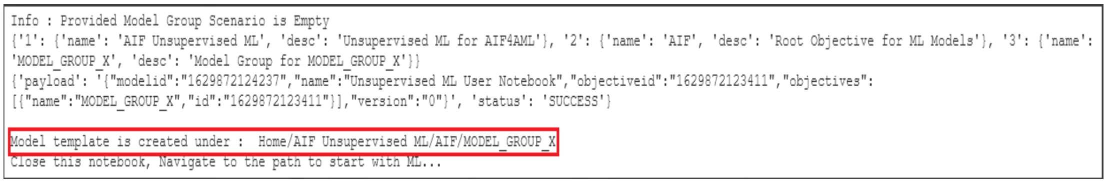
The steps for importing the user notebook into your workspace are:

1. Execute the following line of code which contains the **aif.import_model_template** API. Refer to this API's documentation here: (link to this API's documentation). Here **meta_data_df** refers to the same pandas dataframe created during creation of your model group.

   ```
   %fcc-python-ml4aml
   aif.import_model_template( meta_data_df = pdf,
   model_group_scenario = None )
   ```

   A message will be displayed saying that the model template has been created under "this" particular path.

Figure 78: Path

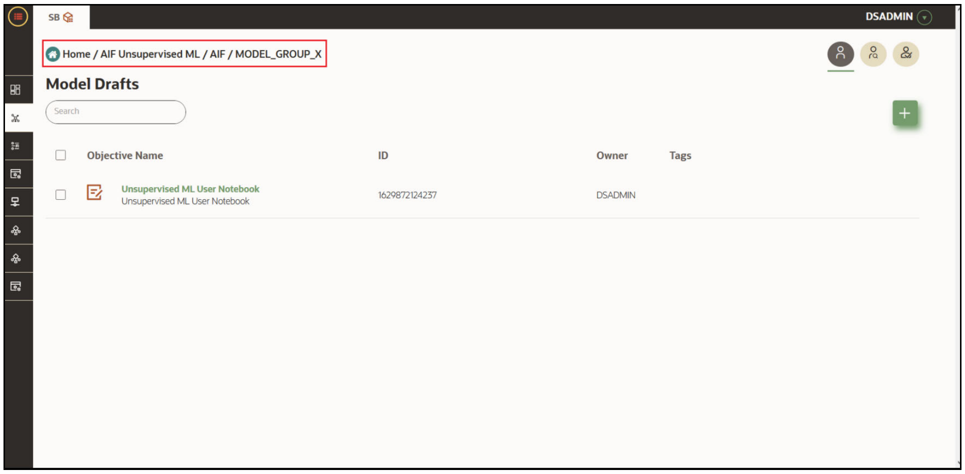


```
Info : Provided Model Group Scenario is Empty
{'1': {'name': 'AIF Unsupervised ML', 'desc': 'Unsupervised ML for AIF4AML'}, '2': {'name': 'AIF', 'desc': 'Root Objective for ML Models'}, '3': {'name':
'MODEL_GROUP_X', 'desc': 'Model Group for MODEL_GROUP_X'}}
{'payload': '{"modelid":"1629872124237","name":"Unsupervised ML User Notebook","objectiveid":"1629872123411","objectives":
[{"name":"MODEL_GROUP_X","id":"1629872123411"}],"version":"0"}', 'status': 'SUCCESS'}

Model template is created under :  Home/AIF Unsupervised ML/AIF/MODEL_GROUP_X
Close this notebook, Navigate to the path to start with ML...
```

2. Navigate to the directory mentioned in the output message to find the user notebook for your created model group.

Figure 79: Model Drafts



Home / AIF Unsupervised ML / AIF / MODEL_GROUP_X

Model Drafts

□	Objective Name	ID	Owner	Tags
□	Unsupervised ML User Notebook Unsupervised ML User Notebook	1629872124237	DSADMIN	

11.6.4.6 View the List of Available Model Groups

Use the `aif.show_model_groups` API to view the list of available model groups.

Execute the following paragraph to view a list of available model groups:

```
%fcc-python-ml4aml
z.show( aif.show_model_groups())
```

The output appears as shown in the Table 31.

Table 31: Output Data for Model Groups

MODEL_GROUP _ID	MODEL_GROUP _NAME	ENTITY_LOGICAL _NAME	ATTRIBUTE_LOGICAL _NAME	ATTRIBUTE _LOGICAL_VALUE
401	LOB1	Customer	Business Domain(s)	General
803	BUS_DMN_LIST_TX_E	Account	Business Domain(s)	General
1201	LOB13	Account	Business Domain(s)	General

Table 31: Output Data for Model Groups

MODEL_GROUP _ID	MODEL_GROUP _NAME	ENTITY_LOGICAL _NAME	ATTRIBUTE_LOGICAL _NAME	ATTRIBUTE _LOGICAL_VALUE
1201	LOB13	Account	Jurisdiction Code	Europe Middle East & Africa

11.6.4.7 Modify Model Groups

Use the `aif.modify_model_groups` API to modify an existing model group.

The following is the input value for the paragraph:

meta_data_df: This is the input pandas data frame that is formed using the available metadata.

To view a list of available model group(s), use the following paragraph:

```
%fcc-python-ml4aml


aif.modify_model_groups(pdf)
```

A successful message is displayed when you add model groups.

```
Successful: Model group modification
```

11.6.4.8 Input Data Frame for Model Group Modification

To modify a model group, a data frame should be specified as shown:

```
%fcc-python-ml4aml


pdf = pd.DataFrame(
{'MODEL_GROUP_NAME'      : ["LOB13"],
  'ENTITY_NAME'          : ["Account"],
  'ATTRIBUTE_NAME'       : ["Jurisdiction Code"],
  'ATTRIBUTE_VALUE'      : ["Americas"],
  'ACTION_TYPE'          : ["ADD"],
  'DISABLE_GROUP'        : ["N"]
                     })z.show(pdf)
```

The output appears as shown in the Table 32.

Table 32: Output Data for Model Group Modification

MODEL_GROUP _NAME	ENTITY _NAME	ATTRIBUTE _NAME	ATTRIBUTE _VALUE	ACTION _TYPE	DISABLE _GROUP
LOB13	Account	Jurisdiction Code	Americas	ADD	N

11.6.4.9 Show Unused Attributes for Model Group Creation

Use the `aif.show_unused_attributes_in_model_group_metadata` API to view the unused attributes after the model group is created. See the following sections to know how to enable the unused attributes.

Execute the following paragraph to view a list of unused attributes:

```
%fcc-python-ml4aml


z.show( aif.show_unused_attributes_in_model_group_metadata())
```

The output appears as shown in the Table 33.

Table 33: Output Data for Unused Attributes

Entity	Attributes
Account	Account Status
Customer	Employee Relationship Type Code
Customer	Employer Industry
Customer	Occupation
Customer	Resident Country
Customer	Registration Type
Customer	Source System

11.6.4.10 Enable or Disabling Unused Attributes for Model Group Creation

Use the `aif.show_unused_attributes_in_model_group_metadata()`

API to view the unused attributes after the model group is created.

The following is the input value for the paragraph:

- **entity_attribute_df**: This is the input data frame formed with respect to the **show_unused_attributes_in_model_group_metadata()**. The Data frame with the ENTITY & ATTRIBUTES column must be provided.
- **disable**: This value has two options, that is, TRUE or FALSE. The value is FALSE by default, which means that the attributes are enabled under metadata for model group creation. If you enter TRUE, then the attributes are disabled.

Execute the following paragraph to view a list of unused attributes:

```
%fcc-python-ml4aml


z.show( aif.show_unused_attributes_in_model_group_metadata())
```

The output appears as shown in the Table 34.

Table 34: Output Data for Unused Attributes

Entity	Attributes
Customer	Customer Status
Account	Account Status

11.6.4.10.1 Enable Unused Attributes

Execute the following paragraph to enable the unused attributes:

```
%fcc-python-ml4aml

aif.enable_attributes_as_model_group_metadata(pdf , disable = False )

z.show( aif.show_metadata_for_model_group_creation())
```

The output appears as shown in the Table 35.

Table 35: Output Data showing Enabled Attributes

ENTITY_NAME	ATTRIBUTE_NAME	ATTRIBUTE_VALUE
Customer	Business Domain(s)	Asset Management, Corporate or Wholesale Banking, Employee Information, General, Institutional Broker-Dealer, Other values as specified by the client, Retail Banking, Retail Brokerage, or Private Client.
Customer	Customer Status	Active, Inactive, Not a Customer, Pending.
Customer	Customer Type	Financial Institution, Individual, Other Organization.
Customer	Jurisdiction Code	Europe Middle East & Africa, India, United States.
Account	Account Status	Active, Closed, Dormant, Inactive, Purge.
Account	Account Type1 Code	Checking, Credit Card, Health Savings, Insurance Policy, Investment, Loan, Money Market, Other values as specified by the client, Others, Retirement, Savings, Stored Value Card, Term/Time/Certificate of Deposit.
Account	Account Type2 Code	Checking, Credit Card, Health Savings, Insurance Policy, Investment, Loan, Money Market, Other values as specified by the client, Others, Retirement, Savings, Stored Value Card, Term/Time/Certificate of Deposit.

11.6.4.10.2 Disable Unused Attributes

Execute the following paragraph to disable the unused attributes:

```
%fcc-python-ml4aml
```

```
aif.enable_attributes_as_model_group_metadata(pdf , disable = True )


z.show( aif.show_metadata_for_model_group_creation())
```

The output appears as shown in the Table 36.

Table 36: Output Data showing Disabled Attributes

ENTITY_NAME	ATTRIBUTE_NAME	ATTRIBUTE_VALUE
Customer	Business Domain(s)	Asset Management, Corporate or Wholesale Banking, Employee Information, General, Institutional Broker-Dealer, Other values as specified by the client, Retail Banking, Retail Brokerage, or Private Client.
Customer	Customer Type	Financial Institution, Individual, Other Organization.
Customer	Jurisdiction Code	Europe Middle East & Africa, India, United States.
Account	Account Type1 Code	Checking, Credit Card, Health Savings, Insurance Policy, Investment, Loan, Money Market, Other values as specified by the client, Others, Retirement, Savings, Stored Value Card, Term/Time/Certificate of Deposit.
Account	Account Type2 Code	Checking, Credit Card, Health Savings, Insurance Policy, Investment, Loan, Money Market, Other values as specified by the client, Others, Retirement, Savings, Stored Value Card, Term/Time/Certificate of Deposit.

11.6.4.11 Add or Remove Attributes to the Model Group Metadata

Use the `aif.add_new_attribute_values_for_model_group_metadata()` API to add or remove attributes after the model group is created.

The following are the input values for the paragraph:

- **entity_attribute_value_df**: The input data frame has the Data frame with the provided entities, Attributes, and Values columns.
- **remove**: This value has two options, that is, TRUE or FALSE. If you enter TRUE, then the attribute values are removed under metadata for model group creation.

Execute the following paragraph to view a list of unused attributes:

```
%fcc-python-ml4aml


pdf = pd.DataFrame({'ENTITY' : ["Customer"],
  'ATTRIBUTE_NAME' : ["Jurisdiction Code"],
 'ATTRIBUTE_VALUE' : ["Australia"],
  'ATTRIBUTE_CODE' : ["AU"]
                   })


z.show(pdf)
```

The output appears as shown in the Table 37.

Table 37: Output Data for Adding or Removing Attributes

ENTITY	ATTRIBUTE_NAME	ATTRIBUTE_VALUE	ATTRIBUTE_CODE
Customer	Jurisdiction Code	Australia	AU

11.6.4.11.1 Add Attributes

Execute the following paragraph to add the attributes:

```
%fcc-python-ml4aml


aif.add_new_attribute_values_for_model_group_metadata(pdf, remove =
False)


z.show( aif.show_metadata_for_model_group_creation())
```

The output appears as shown in the Table 38.

Table 38: Output Data showing Added Attributes

ENTITY_NAME	ATTRIBUTE_NAME	ATTRIBUTE_VALUE
Customer	Business Domain(s)	Asset Management, Corporate or Wholesale Banking, Employee Information, General, Institutional Broker-Dealer, Other values as specified by the client, Retail Banking, Retail Brokerage, or Private Client.
Customer	Customer Status	Active, Inactive, Not a Customer, Pending.
Customer	Customer Type	Financial Institution, Individual, Other Organization.
Customer	Jurisdiction Code	Australia, Europe, Middle East & Africa, India, United States.
Account	Account Status	Active, Closed, Dormant, Inactive, Purge.
Account	Account Type1 Code	Checking, Credit Card, Health Savings, Insurance Policy, Investment, Loan, Money Market, Other values as specified by the client, Others, Retirement, Savings, Stored Value Card, Term/Time/Certificate of Deposit.
Account	Account Type2 Code	Checking, Credit Card, Health Savings, Insurance Policy, Investment, Loan, Money Market, Other values as specified by the client, Others, Retirement, Savings, Stored Value Card, Term/Time/Certificate of Deposit.

11.6.4.11.2 Remove Attributes

Execute the following paragraph to remove the attributes:

```
%fcc-python-ml4aml

aif.add_new_attribute_values_for_model_group_metadata(pdf, remove =
True)

z.show( aif.show_metadata_for_model_group_creation())
```

The output appears as shown in the Table 39.

Table 39: Output Data showing Removed Attributes

ENTITY_NAME	ATTRIBUTE_NAME	ATTRIBUTE_VALUE
Customer	Business Domain(s)	Asset Management, Corporate or Wholesale Banking, Employee Information, General, Institutional Broker-Dealer, Other values as specified by the client, Retail Banking, Retail Brokerage, or Private Client.
Customer	Customer Status	Active, Inactive, Not a Customer, Pending.
Customer	Customer Type	Financial Institution, Individual, Other Organization.
Customer	Jurisdiction Code	Europe Middle East & Africa, India, United States.
Account	Account Status	Active, Closed, Dormant, Inactive, Purge.
Account	Account Type1 Code	Checking, Credit Card, Health Savings, Insurance Policy, Investment, Loan, Money Market, Other values as specified by the client, Others, Retirement, Savings, Stored Value Card, Term/Time/Certificate of Deposit.
Account	Account Type2 Code	Checking, Credit Card, Health Savings, Insurance Policy, Investment, Loan, Money Market, Other values as specified by the client, Others, Retirement, Savings, Stored Value Card, Term/Time/Certificate of Deposit.

11.7 Batch Framework

Batch Schedulers for ML4AML are available for the following three use cases:

1. Supervised ML Batch Framework
2. Unsupervised ML Batch Framework
3. AMLES Batch Framework

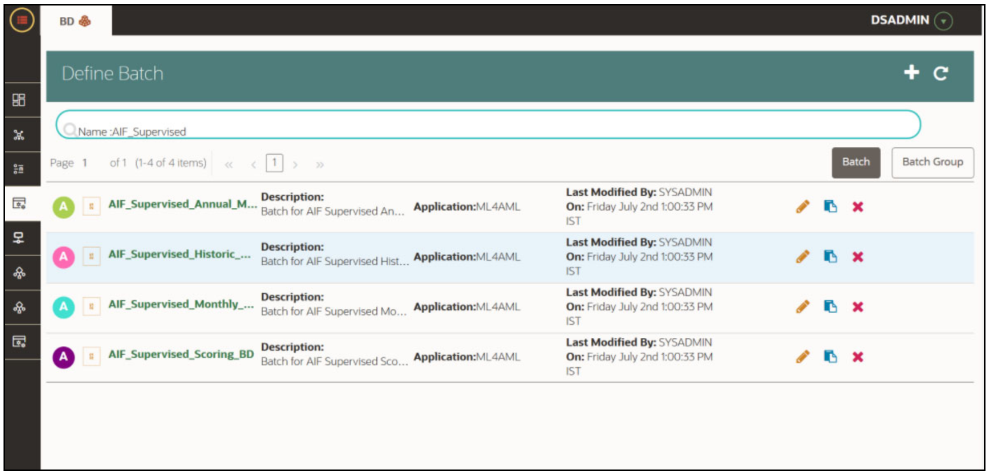
11.7.1 Supervised ML Batch Framework

Following Batches are available out of the box for the Supervised ML framework:

1. Supervised Historic Data Load

2. AIF Supervised Scoring
3. AIF Supervised Annual Model Validation
4. AIF Supervised Monthly Model Validation

Figure 80: Define Batch



BD DSADMIN

Define Batch

Name :AIF_Supervised

Page 1 of 1 (1-4 of 4 items)

Batch | Batch Group

AIF_Supervised_Annual_M... **Description:** Batch for AIF Supervised An... **Application:**ML4AML **Last Modified By:** SYSADMIN **On:** Friday July 2nd 1:00:33 PM IST

AIF_Supervised_Historic_... **Description:** Batch for AIF Supervised Hist... **Application:**ML4AML **Last Modified By:** SYSADMIN **On:** Friday July 2nd 1:00:33 PM IST

AIF_Supervised_Monthly_... **Description:** Batch for AIF Supervised Mo... **Application:**ML4AML **Last Modified By:** SYSADMIN **On:** Friday July 2nd 1:00:33 PM IST

AIF_Supervised_Scoring_BD **Description:** Batch for AIF Supervised Sco... **Application:**ML4AML **Last Modified By:** SYSADMIN **On:** Friday July 2nd 1:00:33 PM IST

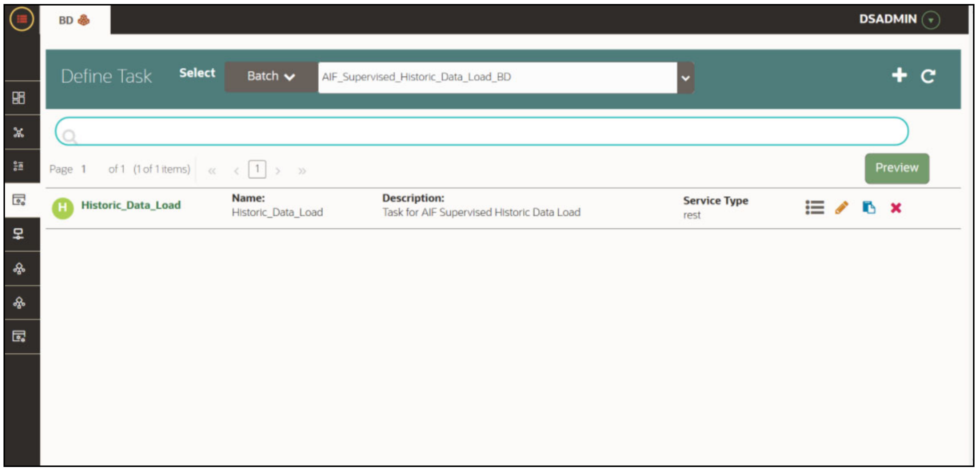
11.7.1.1 Supervised Historic Data Load

1. This is a pre-seeded batch and will be available in all workspaces (production & sandboxes)
2. This Batch is to be executed in the Sandbox workspace.
3. This Batch creates Historical Data Aggregates for ML Model training in the sandbox.

11.7.1.1.1 Batch and Task Parameters

The batch contains a single task named **Historic_Data_Load**

Figure 81: Define Task



BD DSADMIN

Define Task Select Batch AIF_Supervised_Historic_Data_Load_BD

Page 1 of 1 (1 of 1 items)

Preview

Historic_Data_Load **Name:** Historic_Data_Load **Description:** Task for AIF Supervised Historic Data Load **Service Type** rest

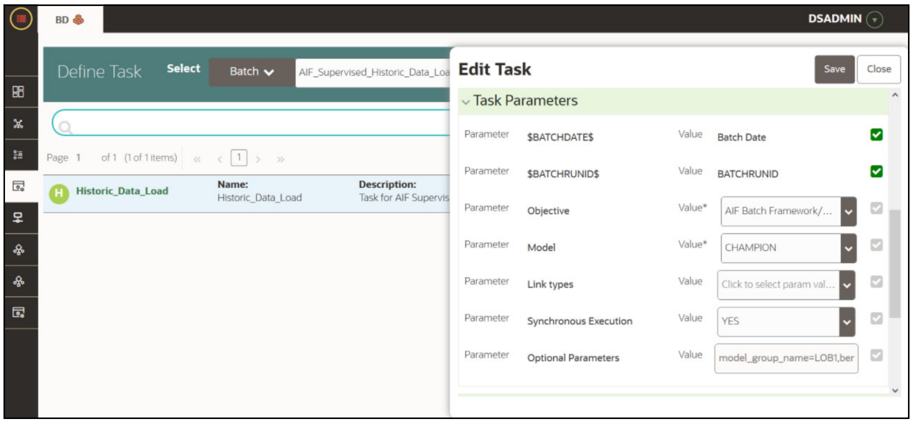
11.7.1.1.1.1 Task: Historic_Data_Load, Task Parameters

- Objective folder for this task:

```
Home / AIF Batch Framework / Supervised ML / Historical Data
```

- Do not change any parameter, except **Optional Parameters.**
- Optional Parameters:
  - **model_group_name**: Name of the Model Groups for which Data Aggregation is to be created. Example **LOB1**
  - **benford_flag**: Flag indicates whether **Benford Law** Computation is required or not. Options **Y** or **N**
  - **benford_digit**: Parameter to Benford law, **Benford Digit**. Options **1** or **2** or **3**
  - **from_date**: Start date for Historic Data lookup in **DD-Mon-YYYY format.**
  - **to_date**: End Date for Historic Data lookup in **DD-Mon-YYYY format.**
- **Example** : model_group_name=**LOB1**,benford_flag=**Y**,benford_digit=**1**,from_date=**01-Jul-2020**,to_date=**31-Jul-2021**
- **Edit** Task Parameters & **Save**.

Figure 82: Edit Task



Historic_Data_Load — Name: Historic_Data_Load — Description: Task for AIF Supervis

Edit Task

Task Parameters

Parameter	$BATCHDATE$	Value	Batch Date	☑
Parameter	$BATCHRUNID$	Value	BATCHRUNID	☑
Parameter	Objective	Value*	AIF Batch Framework/...	☑
Parameter	Model	Value*	CHAMPION	☑
Parameter	Link types	Value	Click to select param val...	☑
Parameter	Synchronous Execution	Value	YES	☑
Parameter	Optional Parameters	Value	model_group_name=LOB1,ber	☑

11.7.1.2 Supervised Scoring

1. This is a pre-seeded batch and will be available in all workspaces (production & sandboxes)
2. This Batch is to be executed in the Production workspace.

11.7.1.2.1 Batch and Task Parameters

The batch contains the following tasks:

- Task 1: Scoring_Data_Load
- Task 2: ML_Scoring
- Task 3: ECM_Event_Processing

11.7.1.2.1.1 Task 1: Scoring_Data_Load, Task Parameters

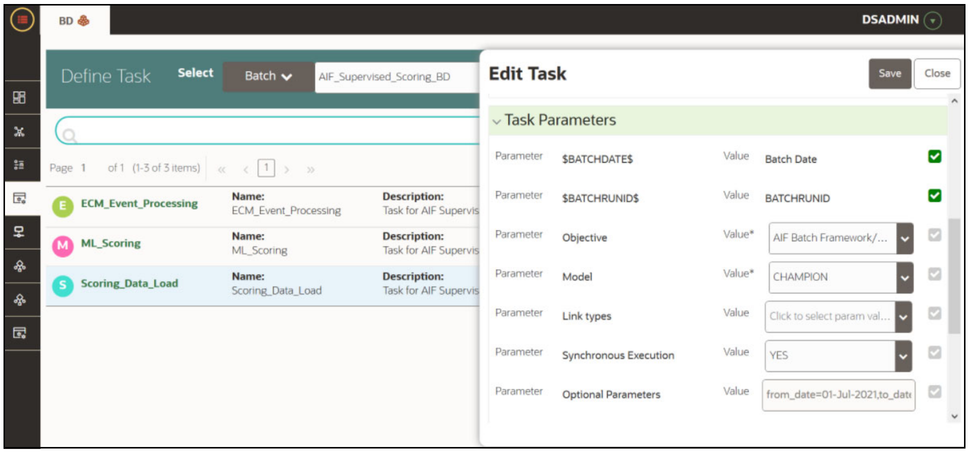
- Objective folder for this task :

```
Home / AIF Batch Framework / Supervised ML / Scoring Data
```

- **Optional Parameters:**

  - **from_date**: Start date for Scoring Data lookup in **DD-Mon-YYYY format**.
  - **to_date**: End Date for Scoring/New Data lookup in **DD-Mon-YYYY format**.
- **Example**: from_date=**01-Jul-2020**,to_date=**31-Jul-2021**
- Optional Parameters can be edited from the **Schedule Batch** option.
- Change any other batch /task parameters, except **Optional Parameters.**

Figure 83: Edit Task



AIF_Supervised_Scoring_BD

	Name:	Description:
ECM_Event_Processing	ECM_Event_Processing	Task for AIF Supervis
ML_Scoring	ML_Scoring	Task for AIF Supervis
Scoring_Data_Load	Scoring_Data_Load	Task for AIF Supervis

Edit Task

Task Parameters

Parameter	$BATCHDATE$	Value	Batch Date	☑
Parameter	$BATCHRUNID$	Value	BATCHRUNID	☑
Parameter	Objective	Value*	AIF Batch Framework/...	☑
Parameter	Model	Value*	CHAMPION	☑
Parameter	Link types	Value	Click to select param val...	☑
Parameter	Synchronous Execution	Value	YES	☑
Parameter	Optional Parameters	Value	from_date=01-Jul-2021,to_date	☑

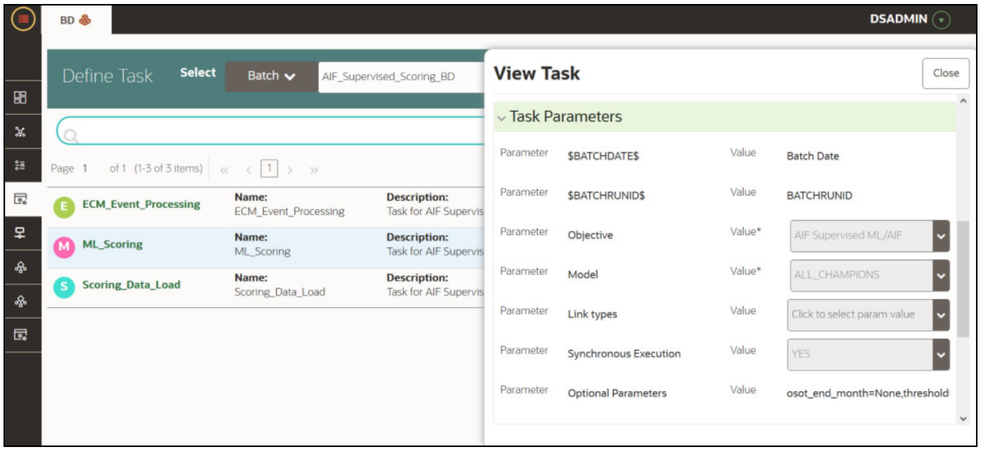
11.7.1.2.1.2 Task 2: ML Scoring, Task Parameters

- Objective folder for this task :

  `Home / AIF Supervised ML / AIF`

  - Navigate to respective model group/scenario folders for actual model templates.
  - Optional Parameters:
    - **osot_end_month:** Specify the scoring data month in **YYYYMM format**. If not specified by default latest month data available in the table will be picked up for scoring.
    - **threshold:** Input threshold or cutoff to create events. Events will be created if the score of an entity exceeds the threshold. **Example: 0.7**
    - **from_date**: Start date for Scoring Data lookup in **YYYYMM format.**
    - **to_date**: End Date for Scoring/New Data lookup in **YYYYMM format. Example** : from_date=**202007**,to_date=**202007**
- Optional Parameters can be edited from the **Schedule Batch** option.
- Do not change any batch/task parameters, except **Optional Parameters.**

Figure 84: View Task



ECM_Event_Processing	Name: ECM_Event_Processing	Description: Task for AIF Supervis
ML_Scoring	Name: ML_Scoring	Description: Task for AIF Supervis
Scoring_Data_Load	Name: Scoring_Data_Load	Description: Task for AIF Supervis

Parameter	$BATCHDATE$	Value	Batch Date
Parameter	$BATCHRUNID$	Value	BATCHRUNID
Parameter	Objective	Value*	AIF Supervised ML/AIF
Parameter	Model	Value*	ALL_CHAMPIONS
Parameter	Link types	Value	Click to select param value
Parameter	Synchronous Execution	Value	YES
Parameter	Optional Parameters	Value	osot_end_month=None,threshold

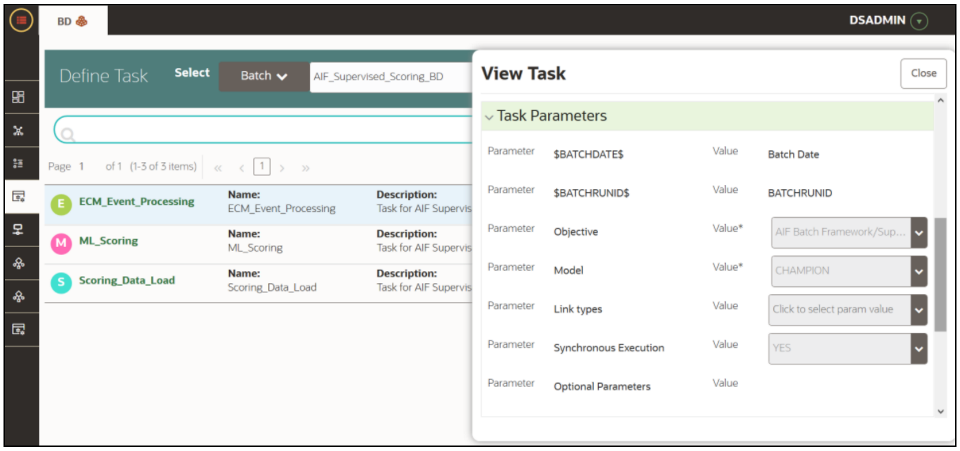
11.7.1.2.1.3 *Task 3: ECM_Event_Processing, Task Parameters*

- Objective folder for this task :

  `Home / AIF Batch Framework / Supervised ML / Event Processing`

- This task does not take any optional parameters.
- Do not change any other batch/task parameters.

Figure 85: View Task



ECM_Event_Processing	Name: ECM_Event_Processing	Description: Task for AIF Supervis
ML_Scoring	Name: ML_Scoring	Description: Task for AIF Supervis
Scoring_Data_Load	Name: Scoring_Data_Load	Description: Task for AIF Supervis

Parameter	$BATCHDATE$	Value	Batch Date
Parameter	$BATCHRUNID$	Value	BATCHRUNID
Parameter	Objective	Value*	AIF Batch Framework/Sup...
Parameter	Model	Value*	CHAMPION
Parameter	Link types	Value	Click to select param value
Parameter	Synchronous Execution	Value	YES
Parameter	Optional Parameters	Value	

11.7.1.2.2 Cleanup Steps in case of running the Scoring Process twice

In case the user wants to run the Scoring Process for the same FIC_MIS_DATE and same MODEL_GROUP_NAME twice, the following cleanup steps should be performed first:

1. Remove the existing events:

   ```
   delete from fcc_am_event_binding where v_event_cd in (select v_event_cd
   from fcc_am_events where prcsng_dt='DD-Mon-YYYY');
   ```

   ```
   delete from fcc_am_event_entity_map where v_event_cd in (select
   v_event_cd from fcc_am_events where prcsng_dt='DD-Mon-YYYY');
   ```

```
delete from fcc_am_event_details where n_event_cd in (select v_event_cd
from fcc_am_events where prcsng_dt='DD-Mon-YYYY');
```

```
delete from fcc_am_events where prcsng_dt='DD-Mon-YYYY');
```

2. Get the child tables which contain scoring results:

```
select D_FIC_MIS_DATE, V_MODEL_GROUP, V_OUTPUT_TABLE_NAME,
V_OUTPUT_TABLE_NAME_ALL_ENTITY

from aif_entity_score

where d_fic_mis_date ='DD-Mon-YYYY'

and model_group_name='<Model_Group_Name>';
```

3. Drop all child tables manually listed in V_OUTPUT_TABLE_NAME and V_OUTPUT_TABLE_NAME_ALL_ENTITY columns from the result of the above query :

```
drop <Child_Table_Name>;
```

4. Delete the parent entry from aif_entity_score:

```
delete from aif_entity_score where d_fic_mis_date='DD-Mon-YYYY'
```

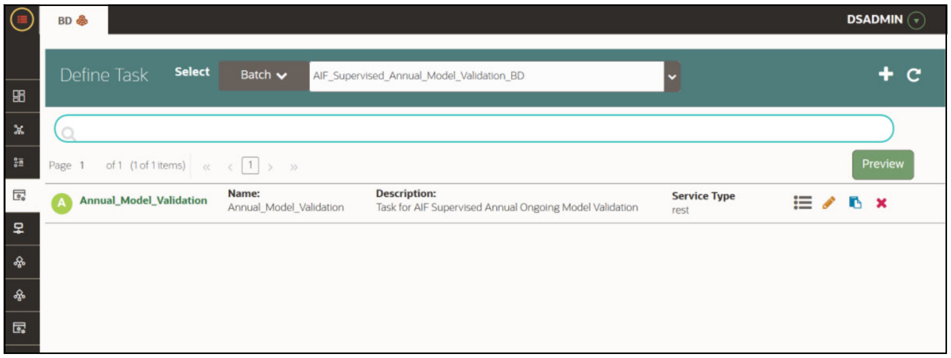
11.7.1.3 Annual Model Validation

1. This is a pre-seeded batch and will be available in all workspaces (production & sandboxes)
2. This Batch is to be executed in the **Production** workspace.
3. This Batch shows ongoing model performance annually.

11.7.1.3.1 Batch and Task Parameters

The batch contains a single task named Annual_Model_Validation

Figure 86: Define Task



AIF_Supervised_Annual_Model_Validation_BD

A Annual_Model_Validation

Name: Annual_Model_Validation

Description: Task for AIF Supervised Annual Ongoing Model Validation

Service Type rest

11.7.1.3.1.1 Task: Annual_Model_Validation, Task Parameters

- Objective folder for Data Quality :

  ```
  Home / AIF Batch Framework / Supervised ML / Ongoing Model Validation /
   Annual
  ```

- Do not change any parameter, except **Optional Parameters.**
- Optional Parameters:

- **model_group_name**: Name of the Model Groups for which Model has been trained. Example **LOB1**
- **model_group_scenario_name:** Name of the Model Groups Scenario for which Model has been trained. Example **Cash**
- **osot_end_month**: Specify the data month in **YYYYMM format**. If not specified by default latest month data available in the table will be picked up for monthly validations as scoring data / new data.
- **model_id**: Model for which ongoing performance to be observed.

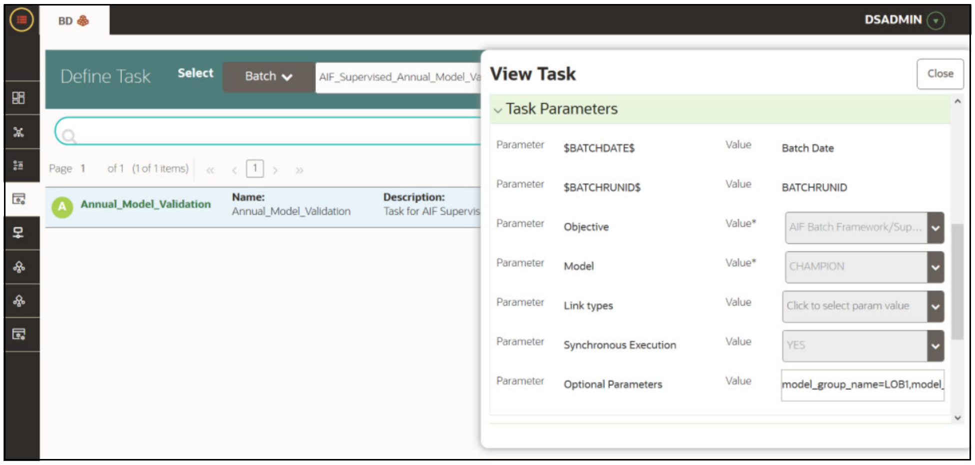
  Table 40 lists the options with their description:

Table 40: Options

Options	Description
Deployed	Shows ongoing performance for the deployed model for a given model group.
Best	Shows ongoing performance for the Best model for a given model group.
All	Shows ongoing performance for all the models for a given model group.
Individual Model ID	Specify individual Model ID's like **'XGB1'**. Refer to the first column in the AUC Summary for the exact Model ID's.
Performance_metrics_list	List of performance metrics on which the Model must be evaluated. Options are : • Kappa Curve • F1 Curve • PR Curve • ROC Curve • Prediction Density • Confusion Matrix: Kappa • AUC Change • PSI

- **Example** : model_group_name=**LOB1**,model_group_scenario_name=**None**,osot_end_month=**None**,model_id_list=**Deployed**,print_result=**False**,performance_metrics_list=**Kappa Curve~F1 Curve~PR Curve~ROC Curve~Prediction Density~Confusion Matrix:Kappa~AUC Change~PSI**
- Optional Parameters can be edited from the **Schedule Batch** option.
- Do not change any batch/task parameters, except **Optional Parameters**.

Figure 87: Define Task



AIF_Supervised_Annual_Model_Va

Page 1 of 1 (1 of 1 items)

Annual_Model_Validation — Name: Annual_Model_Validation — Description: Task for AIF Supervis

Parameter	$BATCHDATE$	Value	Batch Date
Parameter	$BATCHRUNID$	Value	BATCHRUNID
Parameter	Objective	Value*	AIF Batch Framework/Sup...
Parameter	Model	Value*	CHAMPION
Parameter	Link types	Value	Click to select param value
Parameter	Synchronous Execution	Value	YES
Parameter	Optional Parameters	Value	model_group_name=LOB1,model_

11.7.1.4 Monthly Model Validation

1. This pre-seeded batch will be available in all workspaces (production & sandboxes).
2. This Batch is to be executed in the **Production** workspace.
3. This Batch shows ongoing model drift and data quality with respect to new data every month (monthly).

11.7.1.4.1 Batch and Task Parameters

The batch contains a single task named Monthly_Model_Validation.

11.7.1.4.1.1 Task: Monthly_Model_Validation, Task Parameters

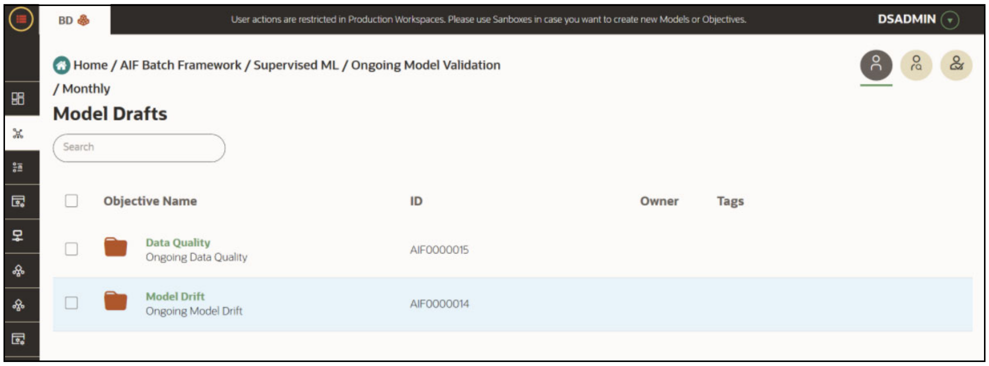
- Objective folder for Data Quality :

```
Home / AIF Batch Framework / Supervised ML / Ongoing Model Validation /
 Monthly / Data Quality
```

- Objective folder for Model Drift :

```
Home / AIF Batch Framework / Supervised ML / Ongoing Model Validation /
 Monthly / Model Drift
```

Figure 88: Model Drafts



User actions are restricted in Production Workspaces. Please use Sanboxes in case you want to create new Models or Objectives.

Home / AIF Batch Framework / Supervised ML / Ongoing Model Validation / Monthly

□	Objective Name	ID	Owner	Tags
□	Data Quality Ongoing Data Quality	AIF0000015		
□	Model Drift Ongoing Model Drift	AIF0000014		

- Do not change any parameter, except **Optional Parameters.**
- Optional Parameters:
  - **model_group_name**: Name of the Model Groups for which Model has been trained. Example **LOB1**
  - **model_group_scenario_name:** Name of the Model Groups Scenario for which Model has been trained. Example **Cash**
  - **osot_end_month**: Specify the data month in **YYYYMM format**. If not specified by default latest month data available in the table will be picked up for monthly validations as scoring data / new data.
  - **FEATURE_INCLUDE:** List of features to be included for **data quality**. Default **None** means everything.
  - **FEATURE_EXCLUDE:** List of features to be excluded for **data quality**. Default **None** means exclude nothing.
    - When both include & exclude is provided. Include takes precedence over exclude.
    - **Example 1** : feature_include="Feature1~Feature2"
    - **Example 2** : feature_exclude="Feature3~Feature4~Feature5"
  - **model_id**: Model for which ongoing performance to be observed.

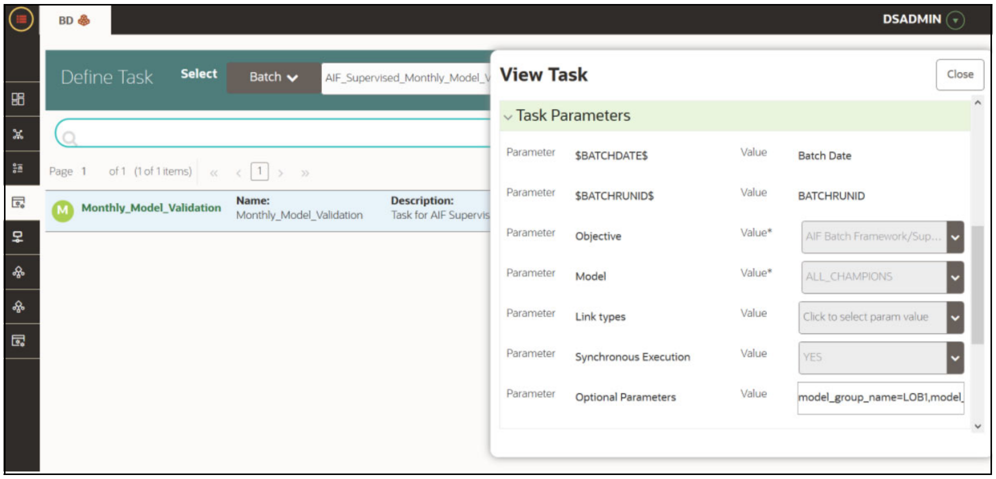
    Table 41 lists the options with their description:

Table 41: Options

Options	Description
Deployed	Shows ongoing performance for the deployed model for a given model group.
Individual Model ID	Specify individual Model ID's like **'XGB1'**. Refer to the first column in AUC Summary for exact Model IDs.

  - **Number_Of_Bins:** Number of bins to be used in discretizing (scalar). **Default is 9.**
  - **Boot_Strap_Samples:** Number of bootstrap samples on which to estimate thresholds. **Default is 5.**
  - **Standard_Deviation_Band_Sigma:** Number of standard deviation band (sigma band). Threshold setting to be used. **Default is 2 sigma.**
- **Example**: **model_group_name**=LOB1,**model_group_scenario_name**=None,**osot_end_month**=None,**model_id**=Deployed,**Number_Of_Bins**=9,**Boot_Strap_Samples**=5,**Standard_Deviation_Band_Sigma**=2,**FEATURE_INCLUDE**=None,**FEATURE_EXCLUDE**=None
- Optional Parameters can be edited from the **Schedule Batch** option.
- Do not change any batch/task parameters, except **Optional Parameters.**

Figure 89: Define Task



AIF_Supervised_Monthly_Model_V

Page 1 of 1 (1 of 1 items)

Monthly_Model_Validation — Name: Monthly_Model_Validation — Description: Task for AIF Supervis

View Task

Task Parameters

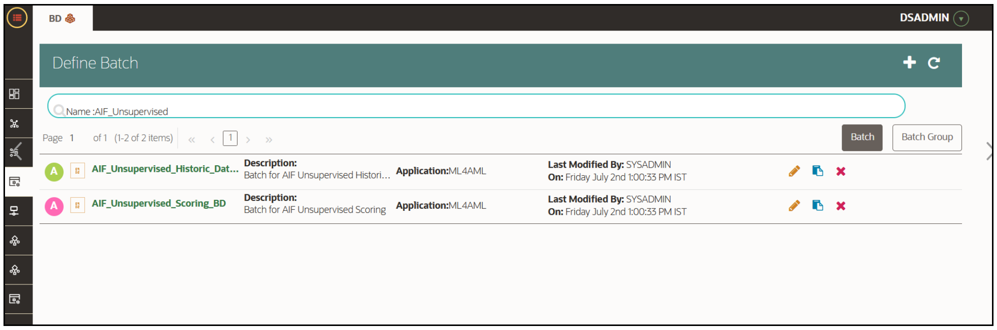
Parameter	$BATCHDATE$	Value	Batch Date
Parameter	$BATCHRUNID$	Value	BATCHRUNID
Parameter	Objective	Value*	AIF Batch Framework/Sup...
Parameter	Model	Value*	ALL_CHAMPIONS
Parameter	Link types	Value	Click to select param value
Parameter	Synchronous Execution	Value	YES
Parameter	Optional Parameters	Value	model_group_name=LOB1,model_

11.7.2 Unsupervised ML Batch Framework

The following batches are available out of the box:

1. Unsupervised Historic Data Load
2. Unsupervised Scoring

Figure 90: Define Batch



Name :AIF_Unsupervised

Page 1 of 1 (1-2 of 2 items)

AIF_Unsupervised_Historic_Dat...	Description: Batch for AIF Unsupervised Histori...	Application:ML4AML	Last Modified By: SYSADMIN On: Friday July 2nd 1:00:33 PM IST
AIF_Unsupervised_Scoring_BD	Description: Batch for AIF Unsupervised Scoring	Application:ML4AML	Last Modified By: SYSADMIN On: Friday July 2nd 1:00:33 PM IST

11.7.2.1 Unsupervised Historic Data Load

1. This is a pre-seeded batch and will be available in all workspaces (production & sandboxes)
2. This Batch is to be executed in the **Sandbox** workspace.

The historic data batch fetches 12 months or more of transactional data for all entities and will be aggregated for each entity. These aggregated features are used to populate the tables in the following with just one row for each entity.

The following tables that this batch will populate.

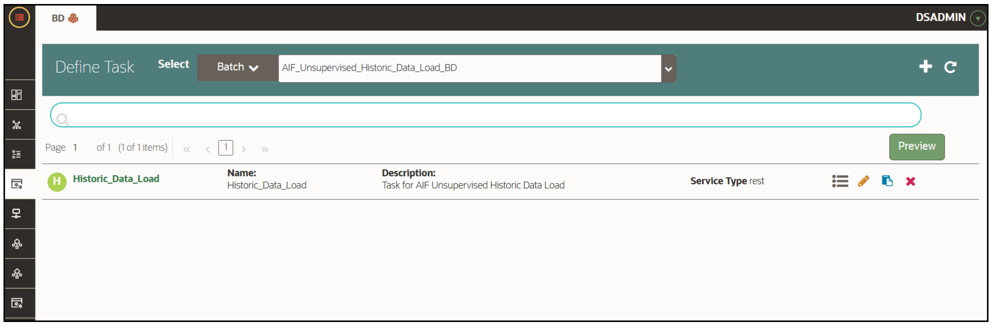
- AIF_BEHAVIORAL_DATA_UNSUP
- AIF_NON_BEHAVIORAL_DATA

These tables will be used for customer segmentation.

This batch has only one task defined under it:

- Historic_Data_Load

Figure 91: Define Task



AIF_Unsupervised_Historic_Data_Load_BD

Page 1 of 1 (1 of 1 items)

Historic_Data_Load

Name: Historic_Data_Load

Description: Task for AIF Unsupervised Historic Data Load

Service Type rest

11.7.2.1.1 Historic_Data_Load

- The objective folder for this task is **Home/AIF Batch Framework/Unsupervised ML/Historical Data**.
- Do not change the parameters **Objective**, **Model**, **Link types**, and **Synchronous Execution**.
- The values in "Optional Parameters" can be edited:
  - **model_group_name**: Name of the model group the batch has to be run for as created in the admin notebook.
  - **model_group_scenario_name**: Name of the model group scenario under this model group for which the batch has to be run.
  - **from_date**: From date in DD-MON-YYYY format. Example: 01-Jul-2021
  - **to_date**: To date in DD-MON-YYYY format. Example: 31-Jul-2021
- **Example**: model_group_name=MODEL_GROUP_X,model_group_scenario_name=None,from_date =01-Jan-2020,to_date=31-Jan-2021

11.7.2.2 Unsupervised Scoring

1. This is a pre-seeded batch and will be available in all workspaces (production & sandboxes)
2. This Batch is to be executed in the **Production** workspace.

The scoring data batch is used to fetch one month or more of transactional data for previously segmented customers, and this data will be used for anomaly scoring.

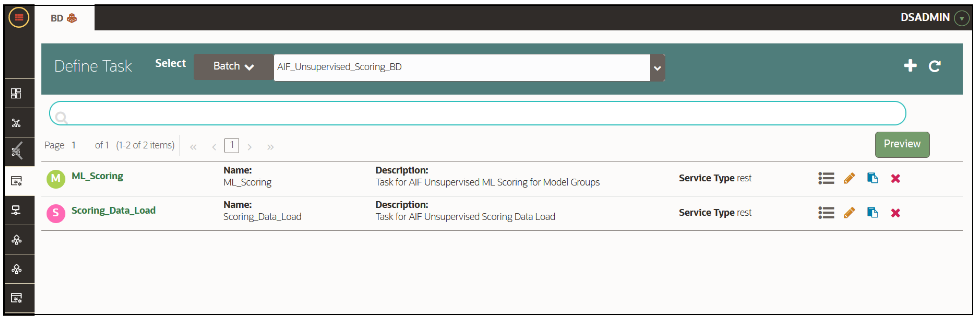
The following tables that this batch will populate.

- **AIF_BEHAVIORAL_DATA_UNSUP_PROD**

- **AIF_NON_BEHAVIORAL_DATA_PROD**

> **NOTE**
>
> 1. This batch has 2 tasks defined under it:
>   - **Scoring_Data_Load**
>   - **ML_Scoring**
> 2. In Sandbox, Cluster Information will be stored in the AIF_ENTITY_CLUSTERS_UNSUP table.
> 3. The data load will bring more entities in AIF_BEHAVIORAL_DATA_UNSUP_PROD and AIF_NON_BEHAVIORAL_DATA_PROD.
> 4. Only the entities in AIF_ENTITY_CLUSTERS_UNSUP should be present in AIF_BEHAVIORAL_DATA_UNSUP_PROD and AIF_NON_BEHAVIORAL_DATA_PROD, so some entities need to be deleted manually.

Figure 92: Define Task



AIF_Unsupervised_Scoring_BD

M ML_Scoring	**Name:** ML_Scoring	**Description:** Task for AIF Unsupervised ML Scoring for Model Groups	**Service Type** rest
S Scoring_Data_Load	**Name:** Scoring_Data_Load	**Description:** Task for AIF Unsupervised Scoring Data Load	**Service Type** rest

Steps to validate Unsupervised Production tables:

1. First, exclude the second task, and execute the batch.
2. After this run the following delete queries:

```
delete from AIF_BEHAVIORAL_DATA_UNSUP_PROD where ENTITY_ID not in
(select ENTITY_ID from AIF_ENTITY_CLUSTERS_UNSUP) and
model_group_name='<Model Group Name>'
```

```
delete from AIF_NON_BEHAVIORAL_DATA_PROD where ENTITY_ID not in (select
ENTITY_ID from AIF_ENTITY_CLUSTERS_UNSUP) and model_group_name='<Model
Group Name>'
```

3. Next, exclude the first task, and execute the batch again.

11.7.2.2.1 Scoring_Data_Load

- The objective folder for this task is:

  `Home/AIF Batch Framework/Unsupervised ML/Scoring Data.`

- Do not change the parameters **Objective**, **Model**, **Link types**, and **Synchronous Execution**.
- The values in "Optional Parameters" can be edited:
  - **from_date**: From date in DD-MON-YYYY format. Example: 01-Jul-2021
  - **to_date**: To date in DD-MON-YYYY format. Example: 31-Jul-2021
- **Example**: from_date=01-Jan-2021,to_date=31-Jan-2021

11.7.2.2.2 ML_Scoring

- The objective folder for this task is **Home/AIF Unsupervised ML/AIF**.
- Do not change the parameters **Objective**, **Model**, **Link types**, and **Synchronous Execution**.

The values in "Optional Parameters" can be edited:

- **osot_end_month**: Specify the scoring data month in **YYYYMM format**. The latest month data available in the table will be picked up for scoring by default if it is not specified.
- **debug_flag**: Assign **True** if debug mode is to be switched on. Default is **False**.
- **data_start_date**: Start date for Scoring Data lookup in **YYYYMM format.**
- **data_end_date**: End Date for Scoring/New Data lookup in **YYYYMM format.**
- **method**: String indicating which anomaly scoring method to use. Currently, **"LDCOF"** is supported, and the default is the same.
- **cutoff_pctl**: Cutoff percentile for anomaly flags. Ranges from 0 to 100. Defaults to **None**.
- **return_flag**: Boolean flag indicates if the user wants a data frame returned. Defaults to **False** and which is the real production use case.

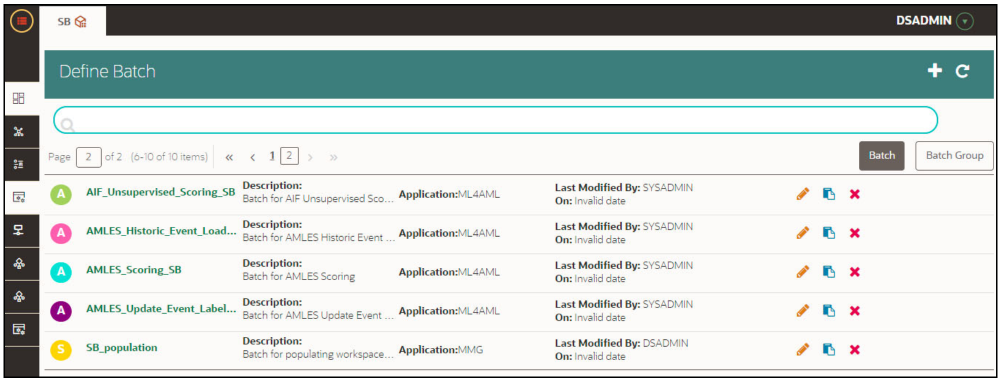
- Example: osot_end_month=None,debug_flag=False,data_start_date=202101,data_end_date=202101,method="LDCOF",cutoff_pctl=None,return_flag=False

11.7.3 AMLES Batch Framework

Following Batches are available out of the box for the Supervised ML framework

1. AMLES Historic Event Load
2. AMLES Scoring
3. AMLES Update Event Labels

Figure 93: Define Batch



Page 2 of 2 (6-10 of 10 items) « ‹ 1 2 › »

A	AIF_Unsupervised_Scoring_SB	**Description:** Batch for AIF Unsupervised Sco...	**Application:**ML4AML	**Last Modified By:** SYSADMIN **On:** Invalid date
A	AMLES_Historic_Event_Load...	**Description:** Batch for AMLES Historic Event ...	**Application:**ML4AML	**Last Modified By:** SYSADMIN **On:** Invalid date
A	AMLES_Scoring_SB	**Description:** Batch for AMLES Scoring	**Application:**ML4AML	**Last Modified By:** SYSADMIN **On:** Invalid date
A	AMLES_Update_Event_Label...	**Description:** Batch for AMLES Update Event ...	**Application:**ML4AML	**Last Modified By:** SYSADMIN **On:** Invalid date
S	SB_population	**Description:** Batch for populating workspace...	**Application:**MMG	**Last Modified By:** DSADMIN **On:** Invalid date

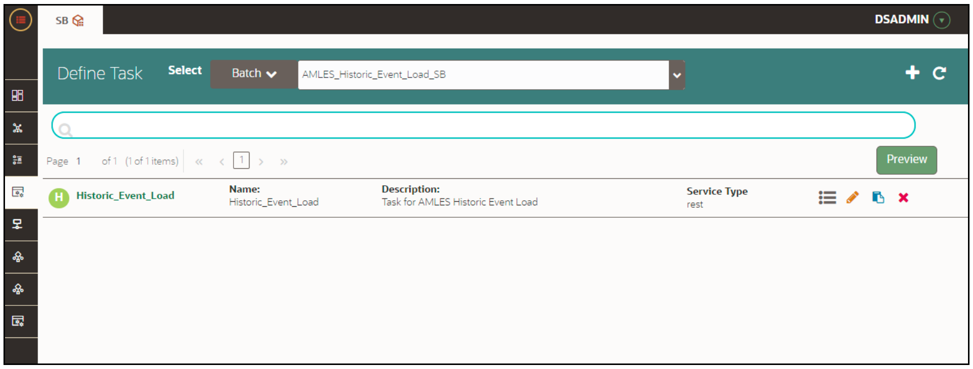
11.7.3.1 AMLES Historic Event Load

1. This is a pre-seeded batch and will be available in all workspaces (production & sandboxes)
2. This Batch is to be executed in the **Sandbox** workspace.
3. This Batch pulls data from the ECM system used for ML Model training in the sandbox.

11.7.3.1.1 Batch and Task Parameters

The batch contains a single task named **Historic_Event_Load.**

Figure 94: Define Task



H	Historic_Event_Load	Name: Historic_Event_Load	Description: Task for AMLES Historic Event Load	Service Type rest

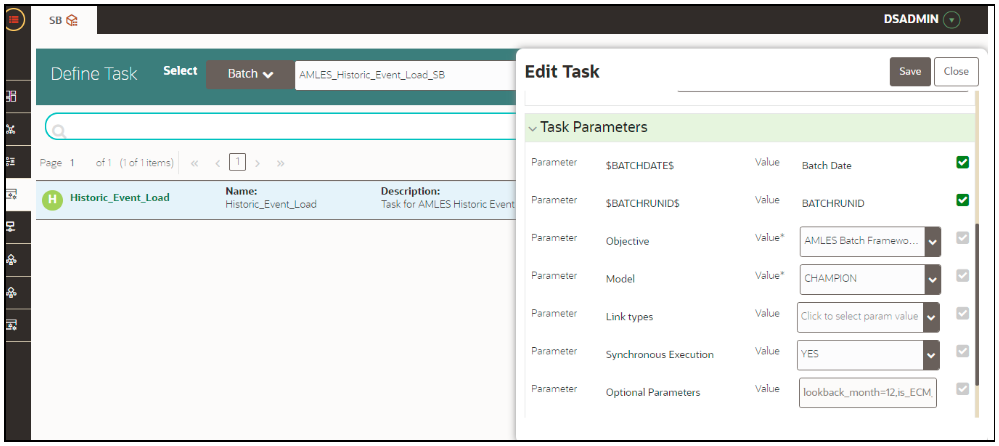
11.7.3.1.1.1 Historic_Event_Load, Task Parameters

- Objective folder for this task :

  `Home / AMLES Batch Framework / Load Events / AMLES Data Load`

- Do not change any parameter, except **Optional Parameters.**
- Optional Parameters:
  - lookback_month: Number of months to look back for data. Example **12**
  - is_ECM_on_remote_schema: Flag indicates **ECM Schema** is on different schema or not. Options **True** or **False**
  - enable_debug_mode: enable debug mode or not. Options **True** or **False**
- **Example** : lookback_month=**12**,is_ECM_on_remote_schema=**True**,enable_debug_mode=**False**
- **Edit** Task Parameters & **Save**.

Figure 95: Define Task



SB
DSADMIN
Define Task
Select
Batch
AMLES_Historic_Event_Load_SB
Page 1 of 1 (1 of 1 items)
Historic_Event_Load
Name: Historic_Event_Load
Description: Task for AMLES Historic Event
Edit Task
Save
Close
Task Parameters

Parameter	$BATCHDATE$	Value	Batch Date	☑
Parameter	$BATCHRUNID$	Value	BATCHRUNID	☑
Parameter	Objective	Value*	AMLES Batch Framewo...	☑
Parameter	Model	Value*	CHAMPION	☑
Parameter	Link types	Value	Click to select param value	☑
Parameter	Synchronous Execution	Value	YES	☑
Parameter	Optional Parameters	Value	lookback_month=12,is_ECM_	☑

11.7.3.2 AMLES Scoring

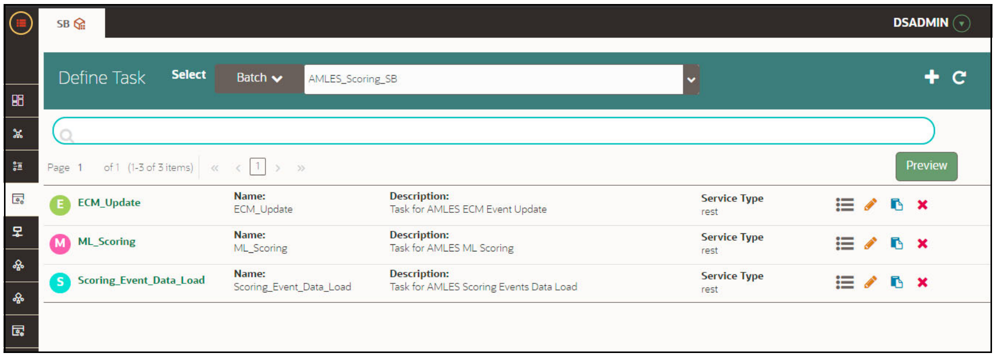
1. This is a pre-seeded batch and will be available in all workspaces (production & sandboxes)
2. This Batch is to be executed in the **Production** workspace.

11.7.3.2.1 Batch and Task Parameters

The batch contains the following tasks:

- **Scoring_Event_Data_Load**
- **ML_Scoring**
- **ECM_Update**

Figure 96: Define Task



E ECM_Update	Name: ECM_Update	Description: Task for AMLES ECM Event Update	Service Type rest
M ML_Scoring	Name: ML_Scoring	Description: Task for AMLES ML Scoring	Service Type rest
S Scoring_Event_Data_Load	Name: Scoring_Event_Data_Load	Description: Task for AMLES Scoring Events Data Load	Service Type rest

11.7.3.2.1.1 Scoring_Event_Data_Load, Task Parameters

- Objective folder for this task :

  `Home / AMLES Batch Framework / Load Events / AMLES Data Load`
- Do not change any parameter, except **Optional Parameters.**
- **Optional Parameters:**
  - lookback_month: Number of months to look back for data. Example **12**
  - is_ECM_on_remote_schema: Flag indicates **ECM Schema** is on different schema or not. Options **True** or **False**
  - enable_debug_mode: enable debug mode or not. Options **True** or **False**
- **Example** : lookback_month=**12**,is_ECM_on_remote_schema=**True**,enable_debug_mode=**False**
- Optional Parameters can be edited from **Schedule Batch** option.
- Do not change any other batch /task parameters, except **Optional Parameters.**

11.7.3.2.1.2 ML_Scoring, Task Parameters

- Objective folder for this task :

  `Home / AMLES`
- Navigate to respective model group/scenario folders for actual model templates.
- **Optional Parameters:**

  - **osot_end_month:** Specify the scoring data month in **YYYYMM format**. If not specified by default latest month data available in the table will be picked up for scoring.
  - **threshold:** Input threshold or cutoff to create events. Events will be created if the score of an entity exceeds the threshold. **Example: 0.7**
  - **data_start_date**: Start date for Scoring Data lookup in **YYYYMM format. Example** : data_start_date=**202007**
  - **data_end_date**: End Date for Scoring/New Data lookup in **YYYYMM format. Example** : data_end_date=**202007**
  - **debug_flag**: flag to set for debugging purpose**.** Few records will be selected. **Options**: **True** or **False**
  - Optional Parameters can be edited from the **Schedule Batch** option.
  - Do not change any batch/task parameters, except **Optional Parameters.**

11.7.3.2.1.3 ECM_Update, Task Parameters

- Objective folder for this task :

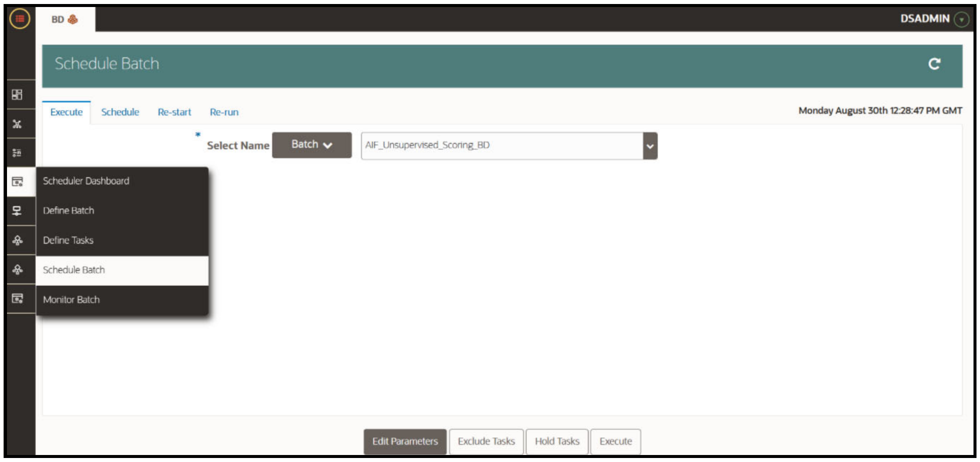
  ```
  Home / AMLES Batch Framework / ECM Update
  ```

- This task does not take any optional parameters.
- Do not change any other batch/task parameters.
- After successfully completing this task, the event score is updated in the KDD_REVIEW table.
  - Table Name: **KDD_REVIEW**
  - Column Name: **EVENT_SCORE**

11.7.4 Execute Batch

- Under **Scheduler Dashboard**, select **Schedule Batch**
- Select the Batch from the drop-down.
- Click **Edit Parameters** to select **MIS Date** and other parameters for the various tasks. **Save** changes.
- Click **Execute** to Execute/Trigger the Batch.

Figure 97: Schedule Batch



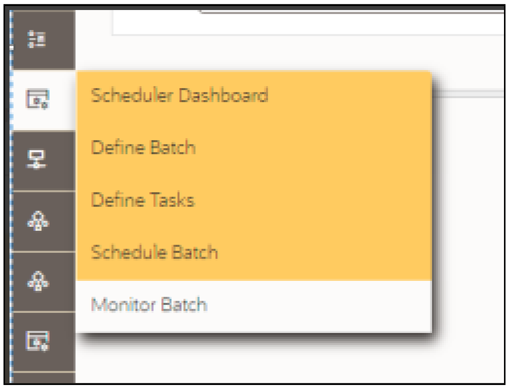
Monday August 30th 12:28:47 PM GMT

* Select Name: Batch — AIF_Unsupervised_Scoring_BD

11.7.5 Monitor Batch

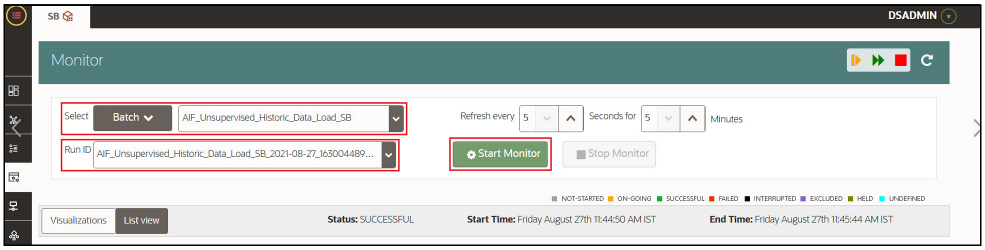
1. Click **Monitor Batch** from **Scheduler Services** on the left pane.

Figure 98: Monitor Batch



2. Select the desired batch name from the drop-down list.
3. Choose the batch ID that has to be monitored.
4. Click **Start Monitor** to start monitoring the batch.

Figure 99: Monitor



Select: Batch — AIF_Unsupervised_Historic_Data_Load_SB

Run ID: AIF_Unsupervised_Historic_Data_Load_SB_2021-08-27_163004489...

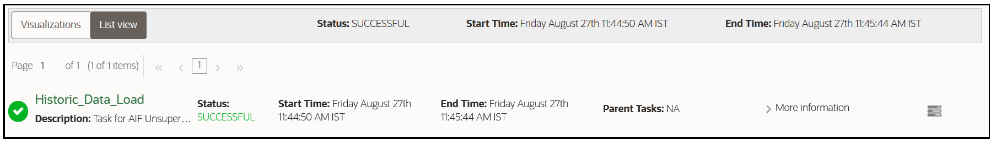
Refresh every 5 Seconds for 5 Minutes

Status: SUCCESSFUL **Start Time:** Friday August 27th 11:44:50 AM IST **End Time:** Friday August 27th 11:45:44 AM IST

5. Click **List View** to view the status of the batch.

6. After the batch has been successfully executed, the status for the batch will be "successful".

Figure 100: List View



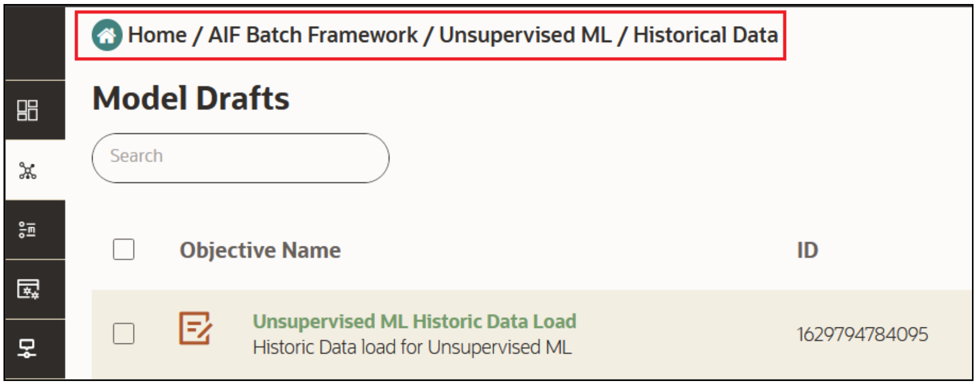
Status: SUCCESSFUL **Start Time:** Friday August 27th 11:44:50 AM IST **End Time:** Friday August 27th 11:45:44 AM IST

Page 1 of 1 (1 of 1 items)

Historic_Data_Load
Description: Task for AIF Unsuper… **Status:** SUCCESSFUL **Start Time:** Friday August 27th 11:44:50 AM IST **End Time:** Friday August 27th 11:45:44 AM IST **Parent Tasks:** NA

7. For further verification of the successful batch execution, navigate to "Home > AIF Batch Framework/Unsupervised ML/Historical Data," where the draft is located.

Figure 101: Model Drafts



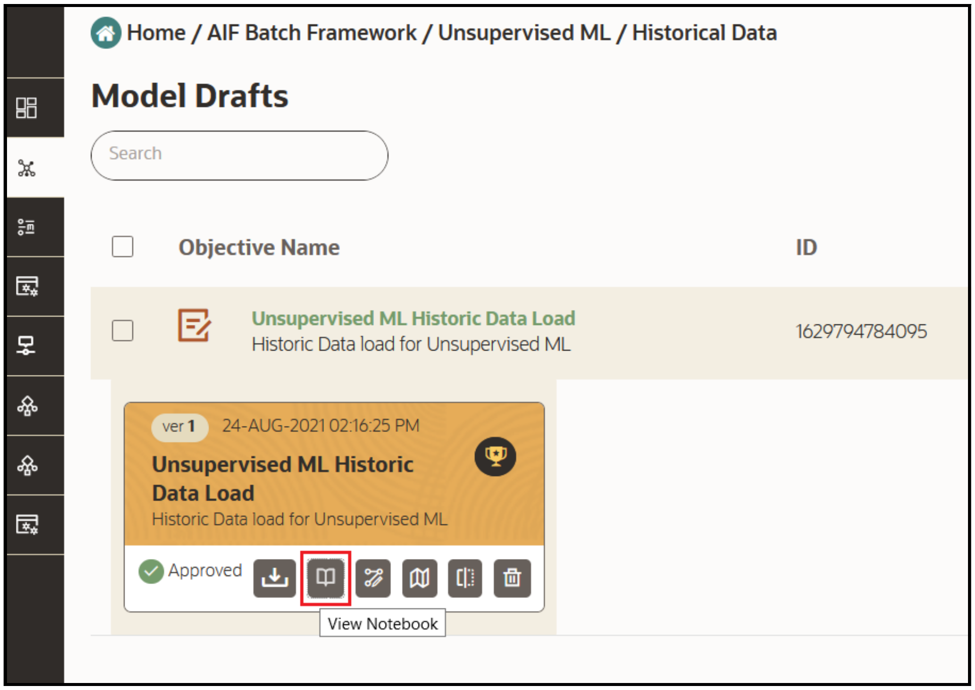
Home / AIF Batch Framework / Unsupervised ML / Historical Data

Model Drafts

Objective Name	ID
Unsupervised ML Historic Data Load Historic Data load for Unsupervised ML	1629794784095

8. Click on the draft and then click **View Notebook**.

Figure 102: Model Drafts



Home / AIF Batch Framework / Unsupervised ML / Historical Data

Model Drafts

Objective Name	ID
Unsupervised ML Historic Data Load Historic Data load for Unsupervised ML	1629794784095

ver 1 24-AUG-2021 02:16:25 PM

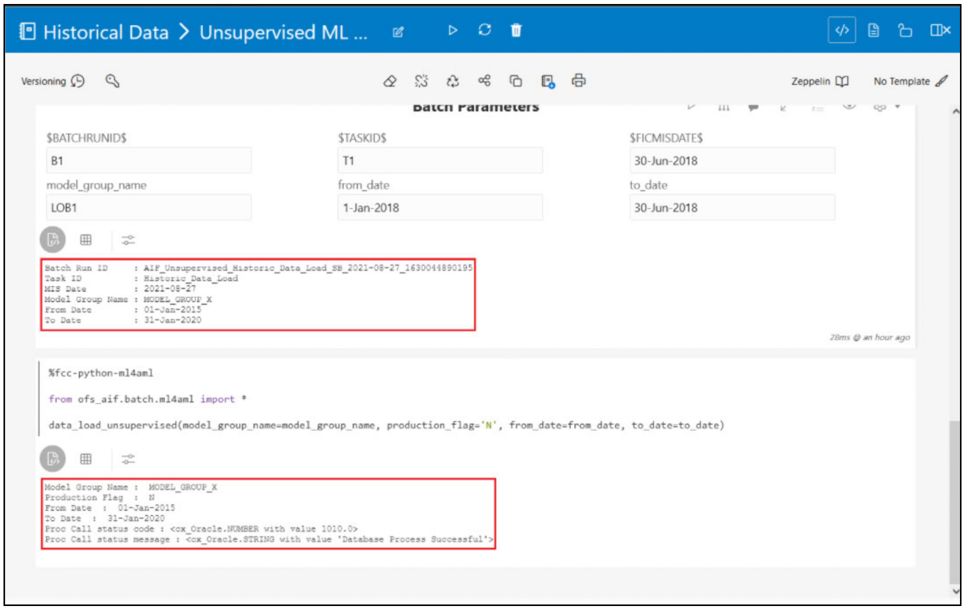
Unsupervised ML Historic Data Load

Historic Data load for Unsupervised ML

Approved

9. Verify if all the draft paragraphs have been executed successfully and displayed no failure messages.

Figure 103: Unsupervised ML



$BATCHRUNID$	$TASKID$	$FICMISDATE$
B1	T1	30-Jun-2018
model_group_name	from_date	to_date
LOB1	1-Jan-2018	30-Jun-2018

```
Batch Run ID      : AIF_Unsupervised_Historic_Data_Load_SB_2021-08-27_1630044890195
Task ID           : Historic_Data_Load
MIS Date          : 2021-08-27
Model Group Name : MODEL_GROUP_X
From Date         : 01-Jan-2015
To Date           : 31-Jan-2020
```

28ms @ an hour ago

```
%fcc-python-ml4aml

from ofs_aif.batch.ml4aml import *

data_load_unsupervised(model_group_name=model_group_name, production_flag='N', from_date=from_date, to_date=to_date)
```

```
Model Group Name :  MODEL_GROUP_X
Production Flag  :  N
From Date  :  01-Jan-2015
To Date  :  31-Jan-2020
Proc Call status code : <cx_Oracle.NUMBER with value 1010.0>
Proc Call status message : <cx_Oracle.STRING with value 'Database Process Successful'>
```

11.8 Data Movement

11.8.1 Supervised

NOTE

- You must drop the partition before re-deployment for the particular model group.
- To drop a partition, run the following SQL commands:

  ```
  ALTER TABLE AIF_NON_BEHAVIORAL_DATA_PROD DROP PARTITION <MODEL_GROUP_NAME>;
  ```

  ```
  ALTER TABLE AIF_BEHAVIORAL_DATA_PROD DROP PARTITION <MODEL_GROUP_NAME>;
  ```

- Import/Export utility is available under the folder `$<Compliance_Studio_HOME>/deployed/ml4aml/datamovement`

11.8.1.1 Export from Sandbox

NOTE This section is intended for DBA/UNIX Admin.

1. Provide read/write/execute permissions to Export_Sandbox_Data.sh
2. Execute following Unix command

   ```
   dos2unix Export_Sandbox_Data.sh
   ```

3. Following grants are needed on Sandbox_Schema / Export_Schema (using sysdba)

```
grant read, write on directory DATA_PUMP_DIR to export_schema_name;
grant export full database to export_schema_name;
```

4. Execute the export utility using the following command

   ```
   ./Export_Sandbox_Data.sh
   ```

   a. Provide Oracle schema details when prompted
   b. Model Group Name will also be captured as part of inputs.

11.8.1.1.1 Outputs

`AIF_DATA.dmp` will be created as part of successful execution.

11.8.1.1.2 Execution Logs

`EXP_AIF_DATA.log` will be created as part of the execution in case of any issues.

> **NOTE** Oracle Drive Compatibility:
>
> 1. This utility can be executed from the same BD folder if the oracle drivers for the BD client and sandbox database server are compatible.
> 2. If not compatible, this utility can be copied to the database UNIX server of the sandbox schema under the folder DATA_PUMP_DIR.
> 3. DATA_PUMP_DIR for any oracle database server can be found out using the following query (using sysdba)
>
>    ```
>    select * from dba_directories where directo-
>    ry_name = 'DATA_PUMP_DIR'
>    ```

11.8.1.2 Import into Production

> **NOTE** This section is intended for DBA/UNIX Admin.

1. Copy `AIF_DATA.dmp` (output of export) and `Import_Sandbox_Data.sh` to DATA_PUMP_DIR of BD Production Database server.
2. Provide read/write/execute permissions to `AIF_DATA.dmp` and `Import_Sandbox_Data.sh`
3. Execute following Unix command

   ```
   dos2unix Import_Sandbox_Data.sh
   ```

4. Following grants are needed on BD Production Schema / Import Schema (using sysdba)

   ```
   GRANT read, write on directory DATA_PUMP_DIR to import_schema_name;
   GRANT import full database to import_schema_name;
   ```

5. Execute the import utility using the following command

   ```
   ./Import_Sandbox_Data.sh
   ```

   a. Provide Oracle schema details of the importing schema when prompted
   b. The Export schema user name / ID will also be captured as part of inputs.

11.8.1.2.1 Outputs

On successful execution, AIF_BEHAVIORAL_DATA & AIF_NON_BEHAVIORAL_DATA will be populated for the model group.

11.8.1.2.2 Execution Logs

`IMP_AIF_DATA.log` will be created as part of the execution in case of any issues.

> **NOTE** DATA_PUMP_DIR for any oracle database server can be found out using the following query (using sysdba)
>
> ```
> select * from dba_directories where directory_name =
> 'DATA_PUMP_DIR'
> ```

11.8.2 Unsupervised

> **NOTE**
>
> - You must drop the partition before re-deployment for the particular model group.
> - To drop a partition, run the following SQL commands:
>
>   ```
>   ALTER TABLE AIF_NON_BEHAVIORAL_DATA_PROD DROP PARTITION <MODEL_GROUP_NAME>;
>   ```
>
>   ```
>   ALTER TABLE AIF_BEHAVIORAL_DATA_UNSUP_PROD DROP PARTITION <MODEL_GROUP_NAME>;
>   ```
> - Import/Export utility is available under the folder `$<Compliance_Studio_HOME>//deployed/ml4aml/datamovement`

11.8.2.1 Export from Sandbox

> **NOTE** This section is intended for DBA/UNIX Admin.

1. Provide read/write/execute permissions to `Export_Sandbox_Data.sh`
2. Execute following Unix command

   ```
   dos2unix Export_Sandbox_Data.sh
   ```
3. Following grants are needed on Sandbox_Schema / Export_Schema (using sysdba)

   ```
   grant read, write on directory DATA_PUMP_DIR to export_schema_name;
   ```

   ```
   grant export full database to export_schema_name;
   ```
4. Execute the export utility using the following command

   ```
   ./Export_Sandbox_Data.sh
   ```

   a. Provide Oracle schema details when prompted

   b. Model Group Name will also be captured as part of inputs.

11.8.2.1.1 Outputs

`AIF_DATA_UNSUP.dmp` will be created as part of successful execution.

11.8.2.1.2 Execution Logs

`EXP_AIF_DATA_UNSUP.log` will be created as part of the execution in case of any issues.

> **NOTE** Oracle Drive Compatibility:
>
> 1. This utility can be executed from the same BD folder if the oracle drivers for the BD client and sandbox database server are compatible.
> 2. If not compatible, this utility can be copied to the database UNIX server of the sandbox schema under the folder DATA_PUMP_DIR.
> 3. DATA_PUMP_DIR for any oracle database server can be found out using the following query (using sysdba)
>
>    ```
>    select * from dba_directories where directory_name = 
>    'DATA_PUMP_DIR'
>    ```

11.8.2.2 Import into Production

> **NOTE** This section is intended for DBA/UNIX Admin.

1. Copy `AIF_DATA.dmp` (output of export) and `Import_Sandbox_Data.sh` to `DATA_PUMP_DIR` of BD Production Database server.
2. Provide read/write/execute permissions to `AIF_DATA.dmp` and `Import_Sandbox_Data.sh`
3. Execute following Unix command

   ```
   dos2unix Import_Sandbox_Data.sh
   ```
4. Following grants are needed on BD Production Schema / Import Schema (using sysdba)

   ```
   GRANT read, write on directory DATA_PUMP_DIR to import_schema_name;
   GRANT import full database to import_schema_name;
   ```
5. Execute the import utility using the following command

   ```
   ./Import_Sandbox_Data.sh
   ```
   a. Provide Oracle schema details of the importing schema when prompted
   b. The Export schema user name / ID will also be captured as part of inputs.

11.8.2.2.1 Outputs

On successful execution, AIF_BEHAVIORAL_DATA_UNSUP will be populated for the model group.

11.8.2.2.2 Execution Logs

`IMP_AIF_DATA _UNSUP.log` will be created as part of the execution in case of any issues.

> **NOTE** DATA_PUMP_DIR for any oracle database server can be found out using the following query (using sysdba)
>
> ```
> select * from dba_directories where directory_name =
> 'DATA_PUMP_DIR'
> ```

11.9 ECM Connector Batch

11.9.1 Supervised ML-ECM Connector Batch

Post Supervised ML Scoring Batch, execute ML-ECM connector batch from ECM UI (AIF-ECM connector batch)

- **RRF Run Name**: Oracle AIF Event Processing in ECM
- **RRF Run code**: Oracle_AIF_Event_Processing
- **RRF Run Parameters**: FIC MIS Date (should match the FIC MIS date of ML scoring batch)

For more information on how to navigate to RRF/Batch framework for the execution in the **Performing Batch Run** section in the ECM Administration Guide.

12 Monitoring Scheduled Batches and Tasks

Tasks are created when the Notebook users execute notebooks or paragraphs. It is important to know the execution status of whether the tasks are created, rejected, canceled, etc. The Tasks page allows you to view the status of the task and associated notebooks, paragraphs, interpreters, etc. By default, all the tasks are listed on the Task page. You can view the specific task using filters such as the task's status, date of creation, and notebook name.

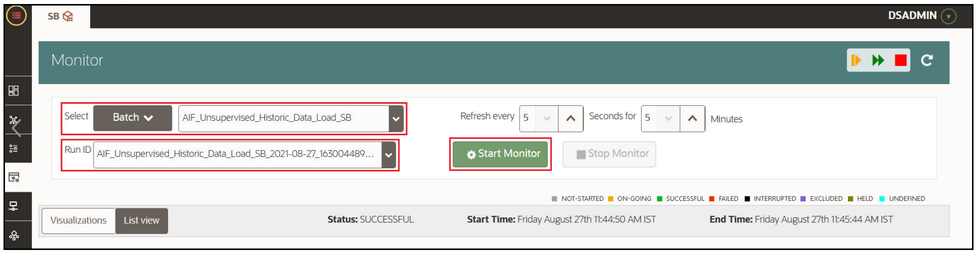
Topics:

- Monitoring Scheduled Batches
- Monitoring Tasks on Notebook Server

12.1 Monitoring Scheduled Batches

1. Login to the Compliance Studio application.
2. On the **Workspace Summary** page, select Launch workspace to display the **CS Production Workspace** window with application configuration and model creation menu.
3. Hover the mouse over the Scheduler Service widget and click **Monitor Batch**. The Monitor page is displayed.
4. Select the desired batch name from the drop-down list.
5. Choose the batch ID that has to be monitored.
6. Click **Start Monitor** to start monitoring the batch.

Figure 104: Monitor



Select Batch AIF_Unsupervised_Historic_Data_Load_SB

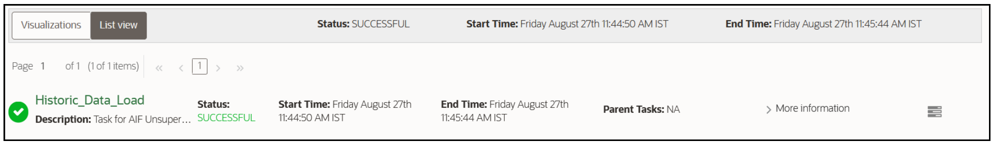
Run ID AIF_Unsupervised_Historic_Data_Load_SB_2021-08-27_163004489...

Refresh every 5 Seconds for 5 Minutes

Status: SUCCESSFUL **Start Time:** Friday August 27th 11:44:50 AM IST **End Time:** Friday August 27th 11:45:44 AM IST

7. Click **List View** to view the status of the batch.
8. After the batch has been successfully executed, the status for the batch will be "successful".

Figure 105: List View



Status: SUCCESSFUL **Start Time:** Friday August 27th 11:44:50 AM IST **End Time:** Friday August 27th 11:45:44 AM IST

Page 1 of 1 (1 of 1 items)

Historic_Data_Load **Description:** Task for AIF Unsuper... **Status:** SUCCESSFUL **Start Time:** Friday August 27th 11:44:50 AM IST **End Time:** Friday August 27th 11:45:44 AM IST **Parent Tasks:** NA

For more information, see the OFS Scheduler Service User Guide.

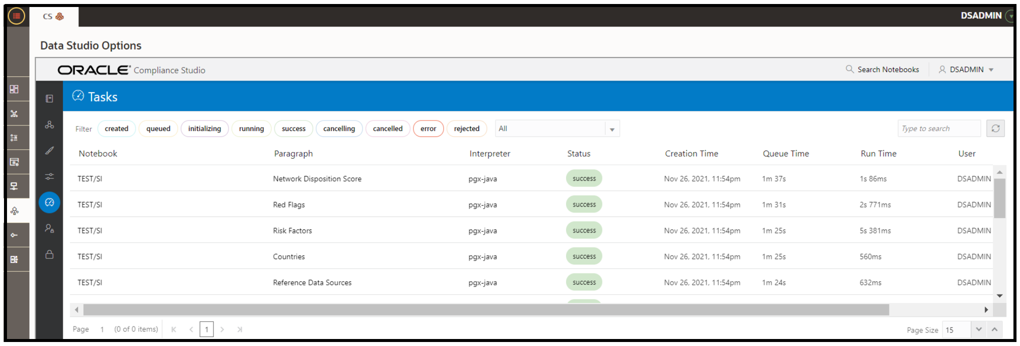
12.2 Monitoring Tasks on Notebook Server

1. Login to the Compliance Studio application.

2. On the **Workspace Summary** page, select Launch workspace 🚀 to display the **CS Production Workspace** window with application configuration and model creation menu.

3. Hover the mouse over the Data Studio Options widget ⚙ and click **Tasks**. The Task page is displayed.

Figure 106: Tasks



Notebook	Paragraph	Interpreter	Status	Creation Time	Queue Time	Run Time	User
TEST/SI	Network Disposition Score	pgx-java	success	Nov 26, 2021, 11:54pm	1m 37s	1s 86ms	DSADMIN
TEST/SI	Red Flags	pgx-java	success	Nov 26, 2021, 11:54pm	1m 31s	2s 771ms	DSADMIN
TEST/SI	Risk Factors	pgx-java	success	Nov 26, 2021, 11:54pm	1m 25s	5s 381ms	DSADMIN
TEST/SI	Countries	pgx-java	success	Nov 26, 2021, 11:54pm	1m 25s	560ms	DSADMIN
TEST/SI	Reference Data Sources	pgx-java	success	Nov 26, 2021, 11:54pm	1m 24s	632ms	DSADMIN

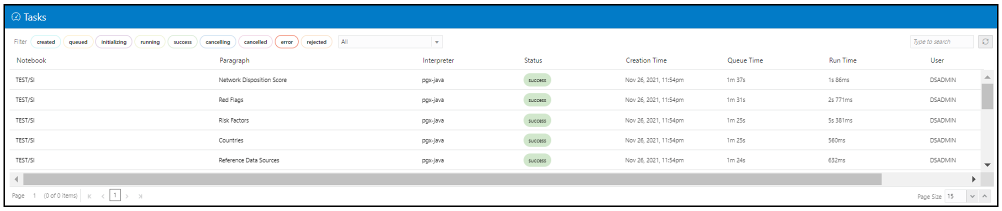
12.2.1 View Tasks Using Status

Use this section to filter the tasks by their status, such as created, rejected, queued, canceled, etc.

To view the task using the status filter, follow these steps:

1. In the Compliance Studio menu list, click Tasks. By default, the Tasks page lists all the tasks of Compliance Studio and displays the notebook, paragraph, interpreter, and user associated with each task. The status filters are displayed on the top of the page. For example, Created, Rejected, etc.

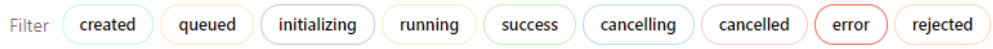
Figure 107: Tasks



Notebook	Paragraph	Interpreter	Status	Creation Time	Queue Time	Run Time	User
TEST/SI	Network Disposition Score	pgx-java	success	Nov 26, 2021, 11:54pm	1m 37s	1s 86ms	DSADMIN
TEST/SI	Red Flags	pgx-java	success	Nov 26, 2021, 11:54pm	1m 31s	2s 771ms	DSADMIN
TEST/SI	Risk Factors	pgx-java	success	Nov 26, 2021, 11:54pm	1m 25s	5s 381ms	DSADMIN
TEST/SI	Countries	pgx-java	success	Nov 26, 2021, 11:54pm	1m 25s	560ms	DSADMIN
TEST/SI	Reference Data Sources	pgx-java	success	Nov 26, 2021, 11:54pm	1m 24s	632ms	DSADMIN

2. To filter the tasks based on the status, click the required status or statuses (created, error, rejected, etc.).

Figure 108: Filter



The tasks in those statuses are displayed with the associated Notebooks, Paragraphs, and Interpreters.

Table 42 describes the list of statuses:

Table 42: List of statuses

Field	Description
created	The task is just created.
queued	The task is in a queue, waiting to be run. This can happen when the same user runs multiple paragraphs of the same interpreter in the same notebook - the interpreter will first finish executing the first paragraph (that is., task) and then move to the second task, which will have status queued until then.
initializing	The tasks are initialized.
running	The tasks are being executed.
rejected	The task is rejected.
success	The task is completed.
Cancelling	The execution of the task is cancelling.
cancelled	The execution of the task is canceled (for example, by clicking the 'Cancel Execution' ("Stop") button on a paragraph).
error	An error occurred during the execution of a task. The error can be one of the following: The concerned interpreter is unsupported The interpreterClient is disconnected The task is not found The status of the task cannot be changed to running or success

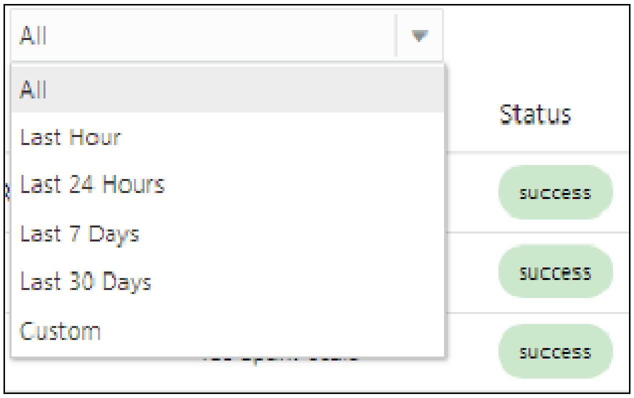
12.2.2 View Tasks Using Time of Creation

Use this section to filter the tasks by their creation time, for example, last hour, last 24 hours, last 7 days, etc.

To filter tasks using date and time, follow these steps:

1. Select the required task creation time or days, for example, last hour, last 24 hours, last 30 days, etc.

   Figure 109: Drop-down list



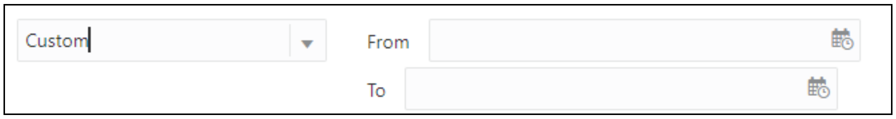
The following is the list of available options

- Last Hour

- Last 24 Hours
- Last 7 days
- Last 30 days
- Custom: Allows the user to enter a specific date range (From and to date-time). If it is empty, it assumes an infinite past or future.

2. To specify the date range, click **Custom**. The date range is displayed.

Figure 110: Custom



Custom

From

To

3. Select date range and time using the calendar. The respective tasks are displayed.

12.2.3 View Tasks Using Names of Notebook

Use this section to filter the tasks by the search field. This filter allows you to filter the tasks based on the notebook name, paragraph title, paragraph code, interpreter, and user.

To filter Tasks, follow these steps:

1. Enter the name of paragraph code, notebook, interpreter, or user in the Search field.

Figure 111: Search

Type to search

The respective tasks are displayed in the Task list.

Table 43 describes the notebook's name, paragraph, and interpreter.

Table 43: Task list

Field	Description
Notebook	The name of the notebook for which the task was created.
Paragraph	The title of the paragraph is associated with the task, or if there is no title, the first line of code of the paragraph.
Interpreter	The name of the interpreter that the task was created to run against.
Status	The status of the task.
Creation Time	The time (in device-local time) when the task was created.
Queue Time	The total time spent by the task in the queue that is, the time between the creation of the task and the beginning of the task execution
Run Time	The time taken for the task to run is the time between the beginning and the end of the task execution.
User	The username of the user who created the task.

2. Click **Refresh** to get the latest list.

13 Restart Services

Use this section to understand how to stop or start the Compliance Studio service if you have an issue with the services.

Topics:

- Stop and Start the Compliance Studio Services
- Stop and Start the PGX Service

13.1 Stop and Start the Compliance Studio Services

To stop the Compliance Studio installer, follow these steps:

1. Navigate to the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/bin` directory.
2. Run the following command:

   ```
   ./stop-script.sh.
   ```

To start the Compliance Studio services, follow these steps:

1. Navigate to the `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/bin` directory.
2. Execute the following command in the console:

   ```
   ./compliance-studio.sh
   ```

13.2 Stop and Start the PGX Service

To stop the PGX service, follow these steps:

1. Navigate to the `<PGX_Installation_Path>/pgx/server/bin` directory.
2. Run the following command:

   ```
   Run ./stop-script.sh
   ```

To start the PGX service, follow these steps:

1. Copy the `<Keystore file name>.jks` file from <`Compliance Studio Installation Path>/batchservice/conf` to the `<PGX Server path>/server/conf` directory.
2. Navigate to the `<PGX_Installation_Path>/pgx/server/bin` directory.
3. Run the following command:

   ```
   nohup ./start-pgx.sh &
   ```

4. After the PGX service runs successfully, run the

   `./FCCM_Studio_ETL_BulkSimilarityEdgeGeneration.sh job` and `<FCCM_Studio path>/FCCM_Studio_ApplyGraphRedaction.sh` file.

   > **NOTE** Ensure that the Global graph is loaded in the PGX Server.

14 Appendix

Topics:

14.1 Create and Execute a Run Executable for Scenario Notebooks

> **NOTE** This is deprecated in the current release and will be removed in the future release.

To create and execute the run executable, follow these steps:

> **NOTE** Ensure that the username and password are set before executing a notebook batch. For more information, see Prepare for Batches.

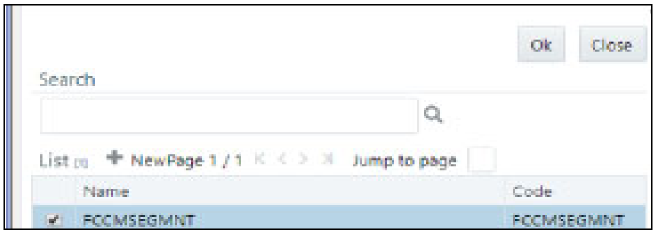
1. Log in to the OFSAA application.
2. Select Financial Services Anti Money Laundering from the tiles menu.

   The Financial Services Anti Money Laundering Application Home Page is displayed with the Navigation list to the left.
3. Navigate to Common Tasks, select Rule Run Framework, and click Run. The Run Definition page is displayed.
4. Click **New** on the List toolbar. The Rule Run Framework window is displayed.

5. Under the Linked To toolbar, click the button next to Directory.

The Folder Selector dialog box is displayed.

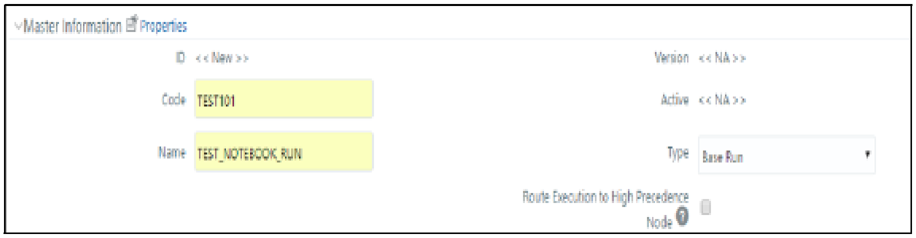
Figure 112: Folder Selector



	Name	Code
☑	FCCMSEGMNT	FCCMSEGMNT

6. Select the directory to link run executable.
7. Click **OK**. The Master Information toolbar is displayed.

Figure 113: Master Information Toolbar



ID << New >>

Code TEST101

Name TEST_NOTEBOOK_RUN

Version << NA >>

Active << NA >>

Type Base Run

Route Execution to High Precedence Node □

8. Enter the following details in the Master Information toolbar as tabulated in the Table 44.

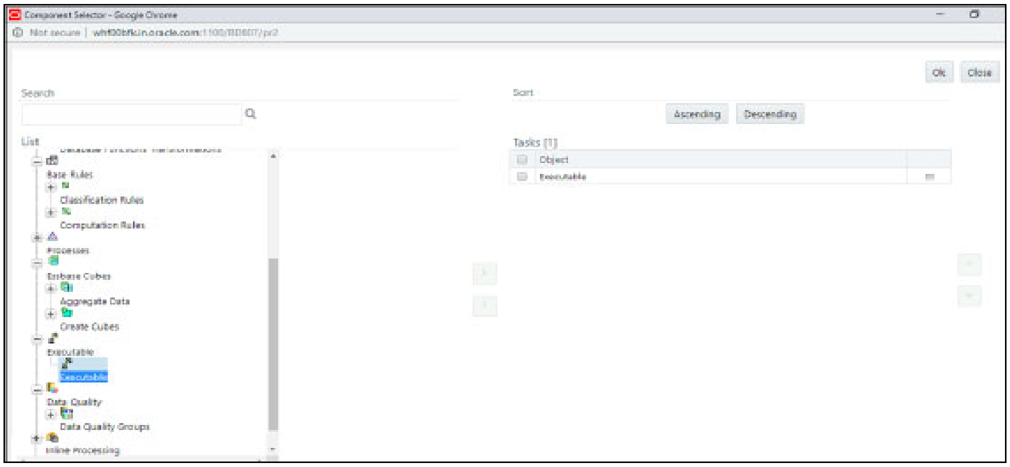
Table 44: Master Information toolbar

Field	Description
Code	Enter the code of the process.
Name	Enter the name of the process.
Type	Select the type for the process.

9. Click **OK**.
10. Click Selector in the List tree. From the options displayed, select Job. The Jobs page is displayed.
11. Click **Executable** on the List tree. From the options displayed, select **Executable**.

    The Executable is displayed on the right.

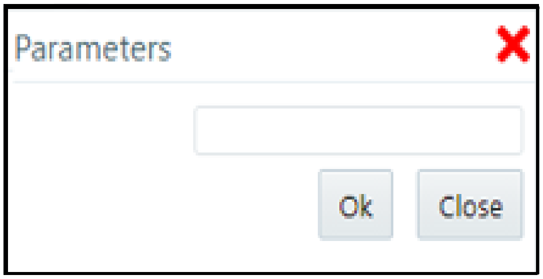
Figure 114: List Tree Screen



12. Select **Executable** from the Tasks list; click the button next to the Executable option.

    The Parameters dialog box is displayed.

    Figure 115: Parameters pop-up

13. Enter the parameters in the following format to create the run executable using the following commands:

    Command

    ```
    "FCCM_Studio_ETL_SqoopJob.sh”,”
    <FROM_FIC_MIS_DATE>”,”<TO_FIC_MIS_DATE>”,”SNAPSHOT_DT=<SNAPSHOT_DATE>:
    ```
    Use this command for **SQOOP** job.

    File Name

    `"FCCM_Studio_ETL_SqoopJob.sh”` - Used for the **SQOOP** job.

    Table 45 lists the parameters with their description:

    Table 45: Parameters

Parameters	Description
FROM_FIC_MIS_DATE	Indicates the date from when the data movement must be performed.
SNAPSHOT_DT	Indicates the date of the data movement code and snapshot date.

Table 45: Parameters

Parameters	Description
TO_FIC_MIS_DATE	Indicates the date until when the data movement must be performed.

Command

`"FCCM_Studio_ETL_Connector.sh”,”<Source>”,”SNAPSHOT_DT=<SNAPSHOT_DATE>"`– Use this command for the **Connector** job.

File Name

`“FCCM_Studio_ETL_Connector.sh”` – Used for the **Connector** job.

Table 46 lists the parameters with their description:

Table 46: Parameters

Parameters	Description
Source	Indicates the data source. Ready-to-use sources are **FCDM** and **ICIJ**.
SNAPSHOT_DATE	Indicates the date of the data movement code and snapshot date.

Command

`"FCCM_Studio_ETL_Graph.sh"`– Used for the **Graph Loading** job.

Command

`"FCCM_Studio_ETL_BulkSimilarityEdgeGeneration.sh"`– Used for the **Bulk Similarity Edge Generation** job.

Command

`"FCCM_Studio_NotebookExecution.sh","notebookID","outputParagraphID","scenarioID","thresholdsetID","extraparams"`– Used for batch execution of published notebook.

File Name

`“FCCM_Studio_NotebookExecution.sh”`– Used for the **Batch Execution of Published Notebook.**

Table 47 lists the parameters with their description:

Table 47: Parameters

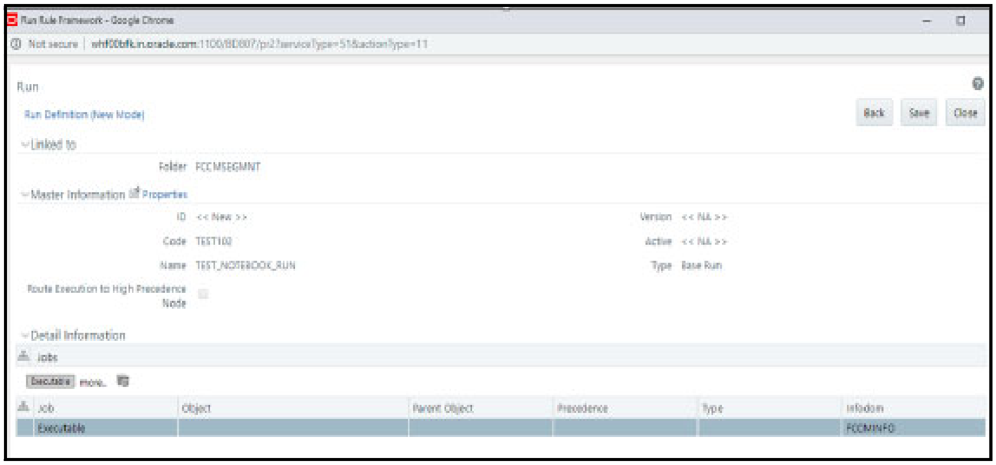
Parameters	Description
notebookId	Indicates the ID of the required notebook.

Table 47: Parameters

Parameters	Description
outputParagraphId	Indicates that the value is always "null"
scenarioId	Indicates the ID of Scenario
thresholdsetId	Indicates the ID of the threshold set with which the notebook will run
sessionId	Indicates the ID of the session in which the notebook will run
extraparams	For the scenario notebook, it will be “null”, but for notebook execution, it depends on the paramkeys used in the notebook

14. Click **OK**. The run executable is displayed in the **Detail Information** pane on the *Run Definition* page.

Figure 116: Run Rule Framework



Run Rule Framework - Google Chrome

Not secure | whf00bfk.in.oracle.com:1100/BD807/pr2?serviceType=51&actionType=11

Run

Run Definition (New Mode)

Back Save Close

Linked to

Folder FCCMSEGMNT

Master Information Properties

ID << New >>

Version << NA >>

Code TEST102

Active << NA >>

Name TEST_NOTEBOOK_RUN

Type Base Run

Route Execution to High Precedence Node □

Detail Information

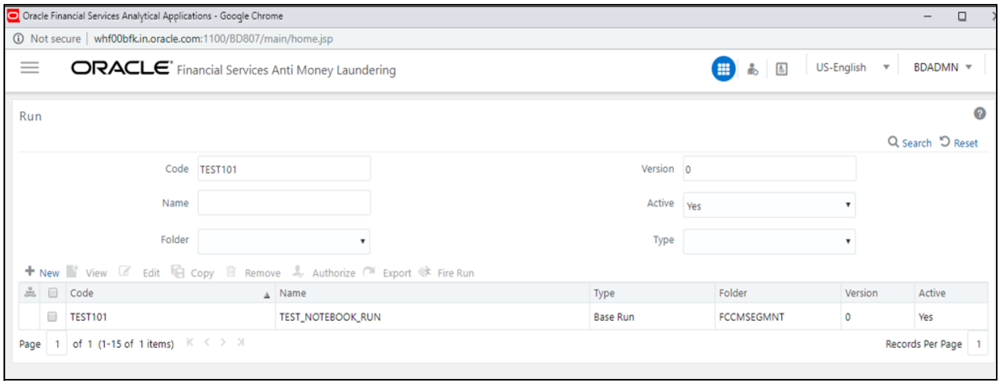
Jobs

Executable more..

Job	Object	Parent Object	Precedence	Type	Infodom
Executable					FCCMINFO

15. Click **Save**. A confirmation message is displayed. The Run executable is created.

Figure 117: Run Screen



Oracle Financial Services Analytical Applications - Google Chrome

Not secure | whf00bfk.in.oracle.com:1100/BD807/main/home.jsp

ORACLE Financial Services Anti Money Laundering

US-English BDADMN

Run

Search Reset

Code TEST101

Version 0

Name

Active Yes

Folder

Type

New View Edit Copy Remove Authorize Export Fire Run

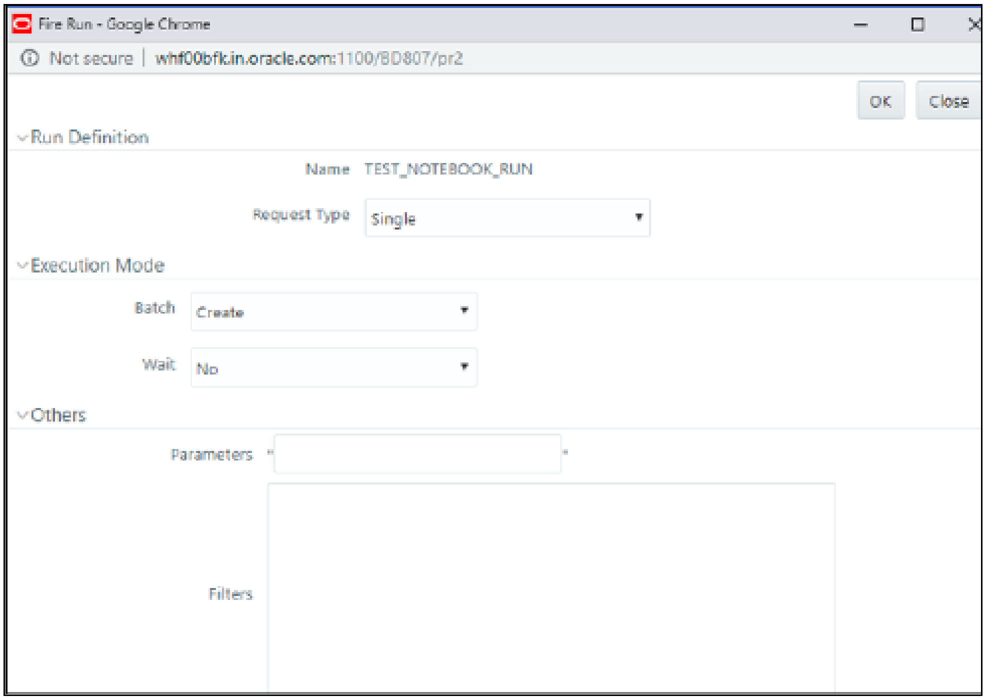
□	Code	Name	Type	Folder	Version	Active
□	TEST101	TEST_NOTEBOOK_RUN	Base Run	FCCMSEGMNT	0	Yes

Page 1 of 1 (1-15 of 1 items)

Records Per Page 1

16. Select the newly created run executable from the Run Definition page that is to be created and click **Fire Run**. The Fire Run Rule Framework dialog box is displayed.

Figure 118: Fire Run Screen



Name TEST_NOTEBOOK_RUN

Request Type Single

Batch Create

Wait No

Parameters

Filters

17. Enter the following Fire Run details as tabulated in the Table 48:

Table 48: Fire Run details

Field	Description
Request Type	Select Request Type based on the following options: ● Single: If the batch must be executed once. ● Multiple: If the batch must be executed multiple times at different intervals.
Batch	Select Batch. It has the following options: ● Create ● Create and Execute From these options, select Create and Execute.
Wait	Select Wait. It has the following options: ● Yes: This will execute the batch after a certain duration. Enter the duration as required. ● No: This will execute the batch immediately.
Filters	Enter the filter details.

18. Click **OK** to run the Run Executable.

    The Run executable starts executing.

19. From the Navigation List, navigate to Common Tasks, click **Operations**, and then select Batch Monitor.

    The Batch Monitor window is displayed.

20. Select the batch that is run in **step 18.** Select the Information Date and Batch Run ID from the drop-down list.
21. Click **Start Monitoring** in Batch Run Details.

    The Batch Run ID and Batch Status details are displayed in the Batch Status details pane.

14.2 Example of ETL

The transformed entity (node/edge) are compared, and the changes are saved in the form of insert and delete, and then the full table is also updated so that graph can load from them in case of PGX server failure or restart.

For Example:

For the account Node: fcdm_account, the Batch 1 Transformed source data is displayed as provided in the Table 49:

Table: Node: **fcdm_account**

Table 49: Account Node

node_id	label	original_id	name	risk	source	date
1000000191101	Account	AMLBMAC664	BENOY	1	FCDM	15-11-15
1000000191102	Account	AMLBMAC420	PERRY	9	FCDM	28-05-15
1000000191103	Account	AMLBMAC504	RAMESH	5	FCDM	18-12-14
1000000191104	Account	AMLBMAC654	DURKA	4	FCDM	14-12-14

On Batch 1 (first batch of ETL), since earlier data are absent, all the entries are also listed as **'insert'** and saved into hive tables with name like '<`entity_provider_name>_insert`', example: '`fcdm_acct_insert`' and delete tables with name like '`<entity_provider_name>_delete`' are not created.

Table 50 lists the fcdm account insert:

Table 50: fcdm_account_insert

node_id	label	original_id	name	risk	source	date
1000000191101	Account	AMLBMAC664	BENOY	1	FCDM	15-11-15
1000000191102	Account	AMLBMAC420	PERRY	9	FCDM	28-05-15
1000000191103	Account	AMLBMAC504	RAMESH	5	FCDM	18-12-14
1000000191104	Account	AMLBMAC654	DURKA	4	FCDM	14-12-14

Table 51 lists the fcdm account:

Table 51: fcdm_account

node_id	label	original_id	name	risk	source	date
1000000191101	Account	AMLBMAC664	BENOY	1	FCDM	15-11-15
1000000191102	Account	AMLBMAC420	PERRY	9	FCDM	28-05-15
1000000191103	Account	AMLBMAC504	RAMESH	5	FCDM	18-12-14
1000000191104	Account	AMLBMAC654	DURKA	4	FCDM	14-12-14

The comparison is made on the subsequent batch for the graph between the previous batch of full data and the current batch of full data to identify insert and delete for all the nodes or edges. The update is considered as deletion of old and addition of new.

For the account Node: fcdm_account, the **Batch 2** Transformed source data is displayed as provided in the Table 52:

Table 52: Node: fcdm_account

node_id	label	original_id	name	risk	source	date
1000000191101	Account	AMLBMAC664	BENOY	1	FCDM	15-11-15
1000000191102	Account	AMLBMAC420	PERRY	9	FCDM	28-05-15
1000000191104	Account	AMLBMAC654	DURGA	4	FCDM	14-12-14
1000000191105	Account	XXXACFRKITINGAC-009	THOMAS	7	FCDM	05-02-15

Here node id ending with:

- 103 has been removed
- 105 has been added
- 104 has been updated

Table 53 lists the fcdm account insert:

Table 53: fcdm_account_insert

node_id	label	original_id	name	risk	source	date
1000000191104	Account	AMLBMAC654	DURGA	4	FCDM	14-12-14
1000000191105	Account	XXXACFRKITINGAC-009	THOMAS	7	FCDM	05-02-15

Table 54 lists the fcdm account delete:

Table 54: fcdm_account_delete

node_id
1000000191103
1000000191104

Table 55 lists the fcdm account of node:

Table 55: Node: fcdm_account

node_id	label	original_id	name	risk	source	date
1000000191101	Account	AMLBMAC664	BENOY	1	FCDM	15-11-15
1000000191102	Account	AMLBMAC420	PERRY	9	FCDM	28-05-15
1000000191104	Account	AMLBMAC654	DURGA	4	FCDM	14-12-14
1000000191105	Account	XXXACFRKITINGAC-009	THOMAS	7	FCDM	05-02-15

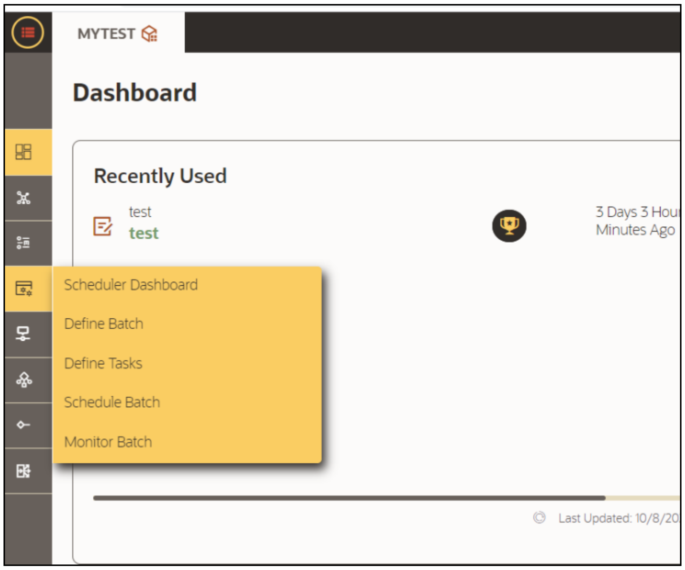
14.3 Example of Creating a batch in Scheduler for Notebook Execution

This topic provides the steps to create a batch in Scheduler for Notebook Execution.

To schedule Notebook execution, perform the following:

1. Login to the Compliance Studio application.
2. On the **Workspace Summary** page, select Launch workspace to display the **Workspace** window with application configuration and model creation menu.
3. Hover the mouse over the Scheduler Service widget the following options are available.

Figure 119: Dashboard



4. Click **Define Batch** to display the Define Batch window.
5. Click on the **+** sign to define a new batch.

Figure 120: Define Batch

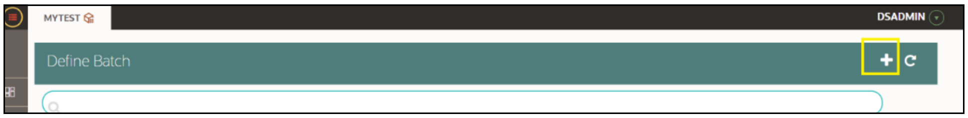
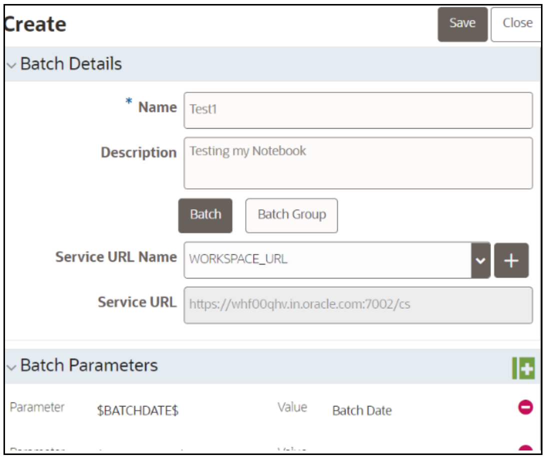


Figure 121: Batch Details



Create

Save Close

Batch Details

* Name: Test1

Description: Testing my Notebook

Batch | Batch Group

Service URL Name: WORKSPACE_URL

Service URL: https://whf00qhv.in.oracle.com:7002/cs

Batch Parameters

Parameter $BATCHDATE$ Value Batch Date

Give the following input:

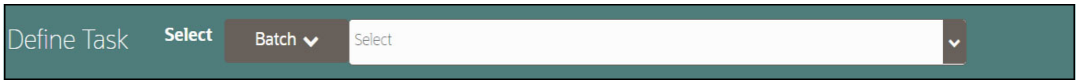
- **Name**: Name of Batch Run (e.g., test1).
- **Description**: Description of defined Batch.
- **Service URL Name**: “workspace_URL”.

- **Service URL**: It will auto-populate after selecting the **Service UL Name** from the drop-down.

6. Click **Save**.

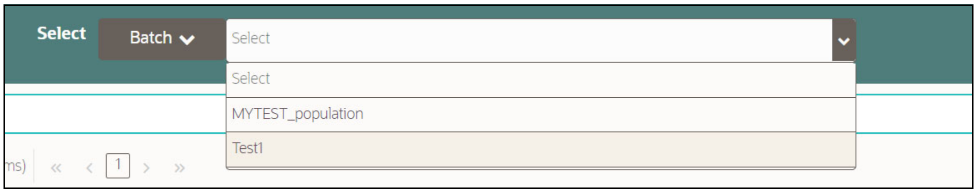
7. Hover the mouse over the Scheduler Service widget and click **Define Task** to display the Define Task window.

Figure 122: Define Task Window



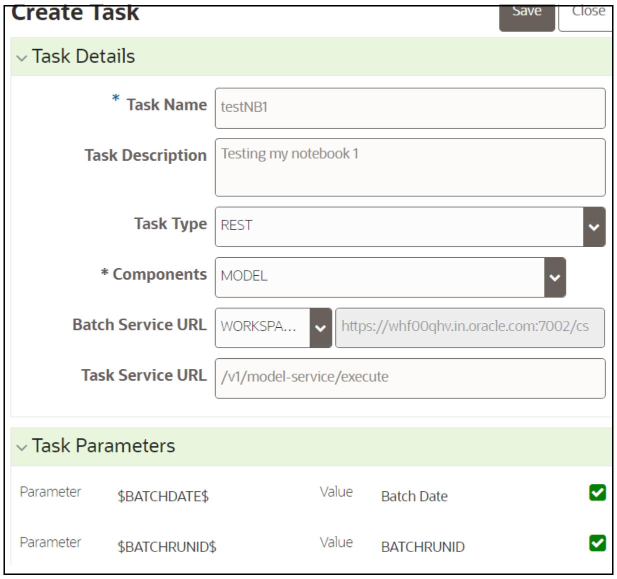
8. Select the Batch from the drop-down and define a task to execute the Notebook through Batch Scheduler.

Figure 123: Batch Scheduler



9. Click on the **+** sign to define a new task.

Figure 124: Create Task



Create Task

Save Close

Task Details

* Task Name: testNB1

Task Description: Testing my notebook 1

Task Type: REST

* Components: MODEL

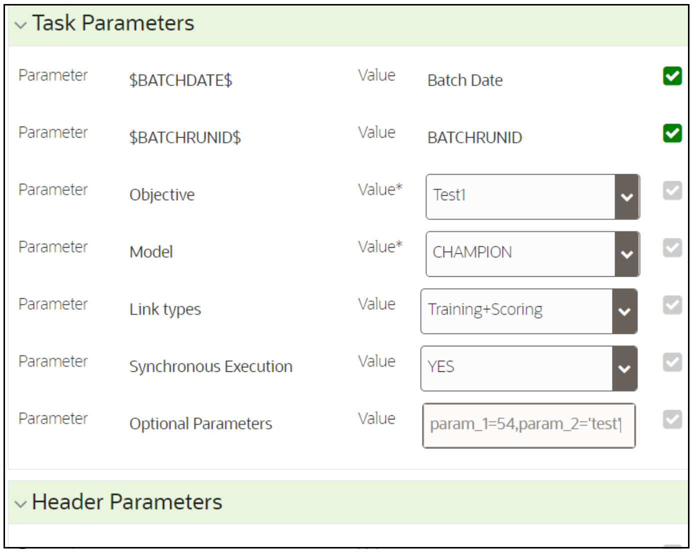
Batch Service URL: WORKSPA... https://whf00qhv.in.oracle.com:7002/cs

Task Service URL: /v1/model-service/execute

Task Parameters

Parameter	$BATCHDATE$	Value	Batch Date	✓
Parameter	$BATCHRUNID$	Value	BATCHRUNID	✓

Figure 125: Task Parameters



Task Parameters

Parameter	$BATCHDATE$	Value	Batch Date	☑
Parameter	$BATCHRUNID$	Value	BATCHRUNID	☑
Parameter	Objective	Value*	Test1	☑
Parameter	Model	Value*	CHAMPION	☑
Parameter	Link types	Value	Training+Scoring	☑
Parameter	Synchronous Execution	Value	YES	☑
Parameter	Optional Parameters	Value	param_1=54,param_2='test1	☑

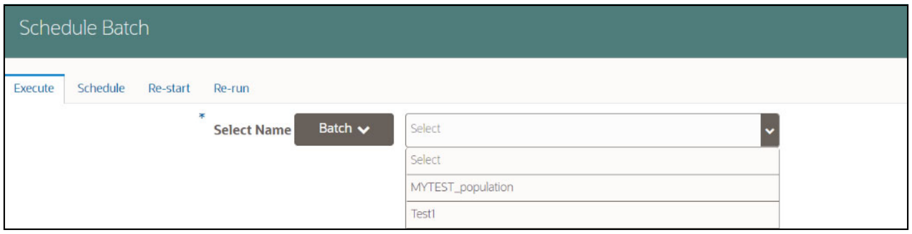
Header Parameters

10. Click **Save**.

11. Hover the mouse over the Scheduler Service widget and click Schedule Batch to display the Schedule Batch window.

12. Select the defined Batch from the drop-down.

Figure 126: Schedule Batch



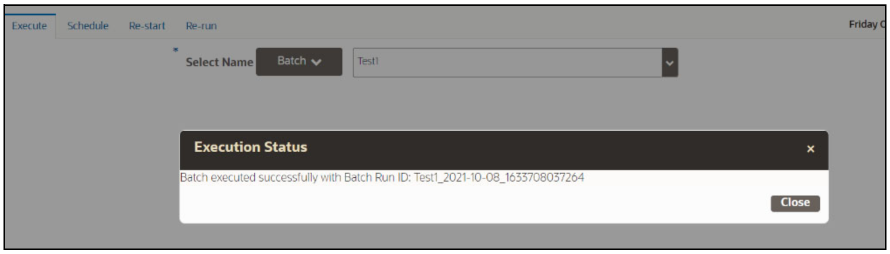
* Select Name: Batch — Select

Select
MYTEST_population
Test1

13. Click **Execute** to execute the Notebook.

The following message is displayed.

Figure 127: Execution Status pop-up



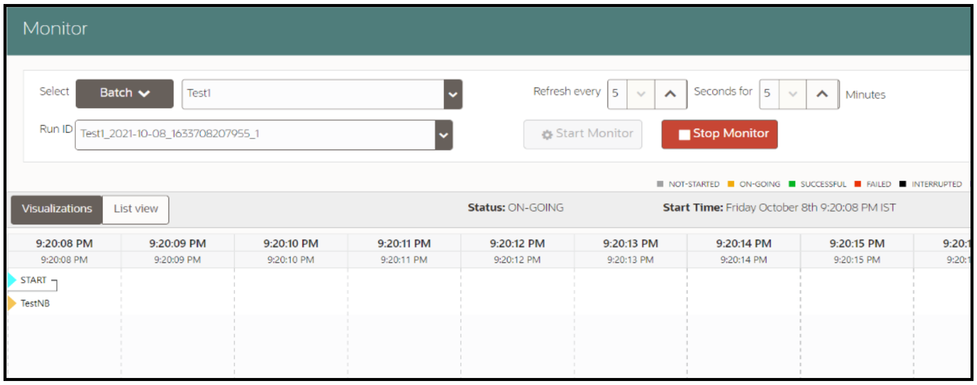
* Select Name: Batch — Test1

Execution Status

Batch executed successfully with Batch Run ID: Test1_2021-10-08_1633708037264

14. Hover the mouse over the Scheduler Service widget and click Scheduler Service to monitor the Notebook execution through the Scheduler batch.

Figure 128: Monitor



Select: Batch — Test1

Run ID: Test1_2021-10-08_1633708207955_1

Refresh every 5 Seconds for 5 Minutes

Status: ON-GOING

Start Time: Friday October 8th 9:20:08 PM IST

a. Select the defined batch name from the Batch drop-down list.

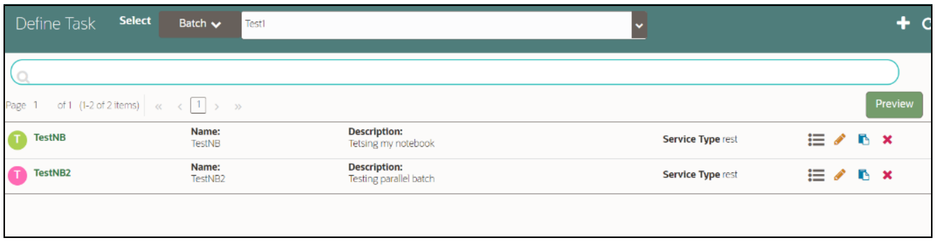
b. Select the Run Id from the drop-down list.

c. Click **Start Monitor**.

14.4 Run Batch in Parallel Mode

1. Create multiple tasks in the same Batch pointing to the same Notebook.

Figure 129: Define Task



Select: Batch — Test1

	Name:	Description:	Service Type
TestNB	TestNB	Tetsing my notebook	rest
TestNB2	TestNB2	Testing parallel batch	rest

2. Execute the Batch from Schedule Batch.

The same Notebook will execute multiple tasks in parallel.

14.5 Create Metadata Indexes using Logstash

To create metadata indexes using Logstash, perform the following:

1. Navigate to `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/load-to-elastic-search/conf` directory.

2. Set the following parameter value as true in the `application.properties` file.

```
index.logstash-conf.apply=true
```

3. Restart Compliance Studio services.

4. Create Indexes. Perform the steps specified in Create Index and Load the Data section.

14.6 Add Self-Signed Certificate

Add the Self-Signed Certification if you use AAI http to access the Compliance Studio.

To add AAI Certificate, perform the following:

1. Download the certificate from AAI and save it.
2. Navigate to `<JAVA INSTALLATION PATH>` where Compliance Studio is installed.
3. Run the following command:

   ```
   keytool -import -file "<Certificate file path>/test.cer" -alias gek_als
   -keystore "<JAVA INSTALLATION PATH>/<JDK Version>/jre/lib/security/
   cacerts" -storepass "changeit"
   ```

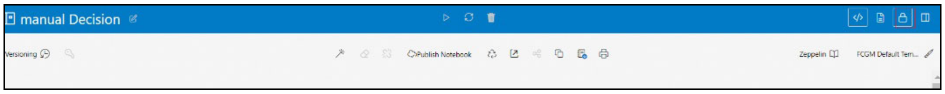
   For example,

   ```
   keytool -import -file "testserver/test.cer" -alias gek_als -keystore
   "testserver/jdk1.8.0_261/jre/lib/security/cacerts" -storepass "changeit"
   ```

14.7 Unlock the Notebook

1. Navigate to the Compliance Studio server with the same URL by changing the port to 7008. `(http://hostname:7008 from http://hostname:7001/cs/home )`
2. Open the notebook. Unlock the notebook, and replace it with the new interpreter name in each paragraph.

   Figure 130: Manual Decision notebook

3. Click **Write** 🔒 Paragraphs icon at the top-right corner to unlock the notebook.

14.8 PGX Advanced Configurations

14.8.1 Data Memory Limit

Data memory limits allow PGX server administrators to control the maximum memory usage for users (individual user or per role) and sessions and shared data. For example, Published Graphs, etc.

The following figure illustrates the data limits where the session limit is bounded to the user limit, which is bounded to the private memory limit, and shared and private memory are bounded to the total PGX memory limit, which is bounded to available machine memory.

Figure 131: Data Memory Limit

<table>
<tr><td colspan="4">Available Machine Memory (Fixed)</td></tr>
<tr><td colspan="4">Total PGX Memory</td></tr>
<tr><td colspan="3">Private Memory</td><td rowspan="3">Shared Memory</td></tr>
<tr><td colspan="2">User 1</td><td>User 2</td></tr>
<tr><td>Session 1</td><td>Session 2</td><td>Session 3</td></tr>
</table>

Table 56 describes the default configurations:

Table 56: Default Configurations

Field	Type	Description	Default
max_per_session_data_memory_ratio	number	The ratio of memory limit for any one session of the PGX engine in relation to the user data memory limit.	1.0
max_per_session_data_memory_size	string	Absolute memory limit for any one session of the PGX engine.	null
max_per_user_data_memory_ratio	number	The ratio of memory limit for any one user of the PGX engine in relation to the private data memory limit.	1.0
max_per_user_data_memory_size	string	Absolute memory limit for any one user of the PGX engine.	null
max_total_data_memory_ratio	number	The ratio of memory limit across the entire PGX engine in relation to available system memory.	0.9
max_total_data_memory_size	string	Absolute memory limit across entire PGX engine.	null
max_total_private_data_memory_ratio	number	The ratio of private memory limit (includes non-published Graphs and PGQL results) in relation to the total PGX engine memory limit.	1.0
max_total_private_data_memory_size	string	Absolute memory limit of private memory (includes non-published graphs and PGQL results).	null
max_total_shared_data_memory_ratio	number	The ratio of memory limit of shared data (includes published graphs and pinned non-referenced graphs) in relation to the total PGX engine memory limit.	1.0
max_total_shared_data_memory_size	string	Absolute memory limit of shared data (includes published graphs and pinned non-referenced graphs).	null

In Compliance Studio, the data memory limits are configured per memory size instead of the ratio. In the case of custom configurations, see the Setting Overall Data Limit section.

14.8.1.1 Checking Data Memory Limit

To check the Memory Limit for PGX Roles, perform the following steps:

1. Login to the OFS Compliance Studio application.
2. Launch **CS Production** workspace.
3. Navigate to **Advanced Model Management**.
4. Create **Objective**.
5. Create **Draft** with Model Studio.
6. Copy the following commands to the Paragraph to verify the memory limit for the role:

```
%pgx-java
instance.getPgxConfigObject()
```

7. Run the Paragraph. Successful execution will display the details in a Paragraph with the memory limit.

14.8.1.2 Setting Overall Data Limit

1. Navigate to `<PGX_Home>/conf` folder and edit the `pgx.conf` file.
2. Update the `data_memory_limits` object with a new key as `max_total_data_memory_size` and its value.

   Example:

   ```
     "data_memory_limits": {
       "max_total_data_memory_size": "100G",
       "max_total_shared_data_memory_size": "80G",
       "max_total_private_data_memory_size": "20G",
       "max_per_session_data_memory_size": "512M"
     },
   ```

3. Save the file.

14.8.1.3 Setting Limits for Each Role

1. Navigate to `<PGX_Home>/conf` folder and edit the `pgx.conf` file.
2. Update the `authorization` section with the required `pgx_role` and set the value for `max_data_memory_size` corresponding to that role.

   Example:

   ```
   {
     "authorization": [
       {
         "pgx_role": "ANALYST",
         "pgx_permissions": [
           {
             "grant": "pgx_session_create"
           },
           {
             "grant": "pgx_server_manage"
           },
           {
             "grant": "pgx_session_new_graph"
           },
           {
             "grant": "pgx_session_add_published_graph"
   ```

```
                },
                {
                  "grant": "pgx_session_compile_algorithm"
                },
                {
                  "grant": "pgx_session_get_published_graph"
                },
                {
                  "grant": "pgx_server_manage"
                },
                {
                  "file_location": "hdfs_storage",
                  "grant": "write"
                },
                {
                  "preloaded_graph": "GlobalGraphIH",
                  "grant": "manage"
                }
              ],
              "max_data_memory_size": "3G"
            }
          }
```

3. Save the file.

14.8.2 PGX Permissions

In PGX, permissions can be granted to roles and individual users.

PGX has the following permissions types:

- resource permissions

  Resource permission allows the user to perform a set of PGX operations on a specific resource (graphs or file-locations).

  For example, the user must have READ permission to run a PGQL query on a graph.

- static permissions

  Static permissions are not related to a specific resource (e.g., graph) but allow to perform a set of PGX operations.

  For example, `PGX_SESSION_CREATE` lets the user create a new PGX session in the notebook. PGX will throw an exception if a user tries to use it without proper permissions.

Some operations require a combination of different static and resource permissions.

For example,

User for Investigation Hub creates a subgraph from Global Graph for Investigation. These users need to have the static `PGX_NEW_GRAPH` permission and the `READ` permission on that graph resource. Similarly, to export a graph, a user would need the `EXPORT` permission on the graph and the `WRITE` permission on the file location to which the graph should be written.

14.8.2.1 Understanding Permissions

14.8.2.1.1 Static Permissions

PGX has the following static permissions.

Table 57 describes the static permissions:

Table 57: PGX Static Permissions

Permission	Related Actions
PGX_SESSION_CREATE	create a new PGX session
PGX_SESSION_NEW_GRAPH	create a new graph, e.g., via loading, in a notebook, or by creating a sub-graph from another graph
PGX_SESSION_GET_PUBLISHED_GRAPH	get a global graph from the public namespace
PGX_SESSION_ADD_PUBLISHED_GRAPH	publish a graph to the public namespace
PGX_SESSION_COMPILE_ALGORITHM	compile a custom algorithm using the PGX Algorithm API
PGX_SESSION_READ_MODEL	load and use an ML model
PGX_SESSION_MODIFY_MODEL	create, train, and store an ML model
PGX_SERVER_GET_INFO	get status information for the PGX instance
PGX_SERVER_MANAGE	manage the PGX server instance

> **NOTE** `PGX_SESSION_ADD_PUBLISHED_GRAPH` also includes `PGX_SESSION_GET_PUBLISHED_GRAPH`; `PGX_SERVER_MANAGE` includes `PGX_SERVER_GET_INFO`.

14.8.2.1.2 Resource Permissions

PGX has two types of resources: file locations and graphs. Each has a different set of permissions defined for them.

14.8.2.1.2.1 File Location Permissions

File locations represent a specific directory (including sub-directories) on a file system or HDFS. PGX allows to define of two different permissions on file locations in the Table 58:

Table 58: File Location Permissions

Permission	Related Actions
READ	Read a file at the file-location

Table 58: File Location Permissions

Permission	Related Actions
WRITE	Write a file to the file location

NOTE `WRITE` also includes `READ`.

14.8.2.1.2.2 Graph Permissions

There are three different permissions defined for graphs in the Table 59:

Table 59: Graph Permissions

Permission	Related Actions
READ	● read or query on the graph data ● run Analyst or custom algorithms on a graph ● create a subgraph or clone the given graph
EXPORT	export the graph
MANAGE	● publish the graph or snapshot ● grant or revoke permissions on the graph ● see the graph

NOTE `EXPORT` also includes `READ`; `MANAGE` includes `EXPORT` and `READ`.

14.8.2.1.2.3 Obtaining a Graph

In PGX, there are different ways a user can obtain a (new) graph. Depending on how users obtain a graph, they will receive different permissions on it. The Table 60 shows the different methods and what permissions a user will be granted on the graph:

Table 60: Obtaining a Graph

Method for Obtaining a Graph	Permissions the user is being granted on the new Graph
Load a graph	`MANAGE`
Create a new graph in the notebook	`MANAGE`
Get a published graph	permissions the user has been granted on the graph
Create a clone/sub-graph from an existing graph	same permissions as the user have on the source graph

14.8.2.1.2.4 Graph Operations

Table 61 describes some essential operations on graphs and what permissions are required for them. Note that some operations require a combination of different permissions:

Table 61: Graph Operations

Operation	Required Permissions
Load a graph	`PGX_SESSION_NEW_GRAPH` & `READ` on all file locations (if any)
Create a new graph in the notebook	`PGX_SESSION_NEW_GRAPH`
Create a clone/sub-graph from an existing graph	`PGX_SESSION_NEW_GRAPH` & `READ` on the source graph
Publish a graph	`PGX_SESSION_ADD_PUBLISHED_GRAPH` & `MANAGE` on the graph
Get a published graph	`PGX_SESSION_GET_PUBLISHED_GRAPH` & `READ` on the graph

14.8.2.1.2.5 ML Model Operations

Table 62 table describes some essential operations on ML models and what permissions are required for them:

Table 62: ML Model Operations

Operation	Required Permissions
Build an ML model	`PGX_SESSION_MODIFY_MODEL`
Fit an ML model	`PGX_SESSION_MODIFY_MODEL`
Store an ML model	`PGX_SESSION_MODIFY_MODEL` & `WRITE` on the location
Load an ML model	`PGX_SESSION_READ_MODEL` & `READ` on the location
Infer using a pre-trained ML model	`PGX_SESSION_MODIFY_MODEL` or `PGX_SESSION_READ_MODEL`

14.8.2.2 Managing Permissions

The permissions are managed in the `pgx.conf` file as `authorization` config. In this, the users and roles are mapped to permissions that are being granted. Such a mapping consists of a list of defined users and roles, each containing a list of permissions granted to them.

For example:

```
{
  "authorization": [
    {
      "pgx_role": "DSADMIN",
      "pgx_permissions": [
        {
          "grant": "pgx_session_create"
        },
        {
```

```
      "grant": "pgx_server_manage"
    },
    {
      "grant": "pgx_session_new_graph"
    },
    {
      "grant": "pgx_session_add_published_graph"
    },
    {
      "grant": "pgx_session_compile_algorithm"
    },
    {
      "grant": "pgx_session_get_published_graph"
    },
    {
      "grant": "pgx_server_manage"
    },
    {
      "grant": "pgx_session_read_model"
    },
    {
      "grant": "pgx_session_modify_model"
    },
    {
      "file_location": "hdfs_storage",
      "grant": "write"
    },
    {
      "preloaded_graph": "GlobalGraphIH",
      "grant": "manage"
    }
  ],
  "max_data_memory_size": "5G"
},
{
  "pgx_user": "James",
      "pgx_permissions": [
        {
          "grant": "pgx_session_create"
        },
        {
          "grant": "pgx_server_manage"
        },
        {
          "grant": "pgx_session_new_graph"
        },
        {
          "file_location": "hdfs_storage",
          "grant": "write"
        },
        {
          "preloaded_graph": "GlobalGraphIH",
          "grant": "manage"
        }
      ],
      "max_data_memory_size": "2G"
    }
  ]
}
```

In the above example, the role, `DSADMIN`, has all the grants, and the user, `James`, has grants to create the session, manage the server, and create a new graph.

To add/update the ready-to-use permission, perform the following steps:

1. Navigate to `<PGX_Home>/conf` folder and edit the `pgx.conf` file
2. Update the `authorization` section by adding mapping for a user or role

Example:

a. For a role, `ANALYST` with grants to query on the published graph, for example: query on Global Graph.

```
{
  "authorization": [
    {
      "pgx_role": "ANALYST",
      "pgx_permissions": [
        {
```

```
        "grant": "pgx_session_create"
      },
      {
        "grant": "pgx_session_get_published_graph"
      },
      {
        "file_location": "hdfs_storage",
        "grant": "write"
      },
      {
        "preloaded_graph": "GlobalGraphIH",
        "grant": "manage"
      }
    ],
    "max_data_memory_size": "2G"
  }
 ]
}
```

b. For a user, `MARIA` with grants to create and publish new graphs used by Analysts.

```
{
  "authorization": [
    {
      "pgx_user": "MARIA",
      "pgx_permissions": [
        {
          "grant": "pgx_session_create"
        },
        {
          "grant": "pgx_session_add_published_graph"
        },
        {
          "grant": "pgx_server_manage"
        },
        {
          "file_location": "hdfs_storage",
          "grant": "write"
```

```
              },
              {
                "preloaded_graph": "GlobalGraphIH",
                "grant": "manage"
              }
            ],
            "max_data_memory_size": "2G"
          }
        ]
      }
```

3. Save the file.

14.8.2.3 Managing File Locations and Preloaded Graphs

As described above, PGX allows permission on file locations. The following sections describe how to add multiple file locations and preloaded graphs, then authorize the grant.

14.8.2.3.1 Manage Multiple File Locations

In a scenario where multiple graphs are loaded from multiple file systems like the local file system or HDFS, these file locations have to be mapped first, and then the grants are provided.

> **NOTE** Each file locations represent a specific directory (including sub-directories).

To add/update file locations:

1. Navigate to `<PGX_Home>/conf` folder and edit the `pgx.conf` file
2. Update the `file_locations` section by providing a name and location.

   Example:

```
{
  "file_locations": [
    {
      "name": "hdfs_storage",
      "location": "hdfs:///user/fccstudio"
    },
    {
      "name": "local_storage",
      "location": "/home/fccstudio/graphs"
    }
  ]
}
```

In the above example, two file locations are present, where the first one represents a file location from HDFS and the second one from the local file system.

3. To grant access, update the authorization section of a user or role.

   Example:

```
{
  "authorization": [
    {
      "pgx_user": "MARIA",
      "pgx_permissions": [
        {
          "grant": "pgx_session_create"
        },
        {
          "grant": "pgx_session_add_published_graph"
        },
        {
          "grant": "pgx_server_manage"
        },
        {
          "file_location": "hdfs_storage",
          "grant": "write"
        },
        {
          "file_location": "local_storage",
          "grant": "read"
        },
        {
          "preloaded_graph": "GlobalGraphIH",
          "grant": "manage"
        }
      ],
      "max_data_memory_size": "2G"
    }
  ]
}
```

   In the above example, user Maria has read access on files present in local_storage and writes access on files present in `hdfs_storage`.

4. Save the file.

14.8.2.3.2 Manage Multiple Preloaded Graphs

In a scenario where multiple preloaded graphs need to be loaded, these graphs have to be mapped first, and then the grants should be provided in the authorization section.

A preload_graph mapping consists of:

- Path: Path of the graph JSON file
- Name: Name of the Graph

Optional properties:

- Publish: Boolean value to publish the graph or not. If not set, the default value of this flag is true.
- Publish with snapshot: Boolean value to publish the graph and all future graph snapshots. The default value is false.

> **NOTE**
>
> - Only one of these two flags can be set to true at the time. However, publishing the graph with snapshots also publishes the first version of the graph.
> - Ensure that the corresponding file_location is also set.

To add/update the preloaded graph:

1. Navigate to `<PGX_Home>/conf` folder and edit the `pgx.conf` file
2. Update the `file_locations` section by providing a name and location.

   Example:

```
{
  "preload_graphs": [
    {
      "name": "GlobalGraphIH",
      "path": "hdfs:///user/fccstudio/graph.json",
      "publish": false,
      "publish_with_snapshots": true
    },
    {
      "name": "OtherGraph",
      "path": "/home/fccstudio/graphs/OtherGraph/graph.json",
      "publish": true,
      "publish_with_snapshots": false
    },
    {
      "name": "AnotherGraph",
      "path": "/home/fccstudio/graphs/AnotherGraph/graph.json"
    }
  ]
}
```

In the above example, three graphs are present, where the `GlobalGraphIH` graph and its future snapshots will be published, and graphs, `OtherGraph` and `AnotherGraph`, will be published.

3. To grant access, update the authorization section of a user or role.

   Example:

```
{
  "authorization": [
    {
      "pgx_user": "Harry",
      "pgx_permissions": [
        {
          "grant": "pgx_session_create"
        },
        {
          "grant": "pgx_server_manage"
        },
        {
          "grant": "pgx_session_new_graph"
        },
        {
          "grant": "pgx_session_add_published_graph"
        },
        {
          "grant": "pgx_session_compile_algorithm"
        },
        {
          "grant": "pgx_session_get_published_graph"
        },
        {
          "grant": "pgx_server_manage"
        },
        {
          "grant": "pgx_session_read_model"
        },
        {
          "grant": "pgx_session_modify_model"
        },
        {
          "file_location": "hdfs_storage",
          "grant": "write"
        },
        {
          "file_location": "local_storage",
          "grant": "read"
        },
        {
          "preloaded_graph": "GlobalGraphIH",
          "grant": "manage"
        },
        {
          "preloaded_graph": "OtherGraph",
          "grant": "read"
        },
        {
          "preloaded_graph": "AnotherGraph",
          "grant": "read"
        }
      ],
      "max_data_memory_size": "5G"
    },
    {
      "pgx_user": "PETER",
      "pgx_permissions": [
        {
          "grant": "pgx_session_create"
        },
        {
          "grant": "pgx_session_add_published_graph"
        },
        {
          "grant": "pgx_server_manage"
        },
        {
          "file_location": "local_storage",
          "grant": "read"
        },
        {
          "preloaded_graph": "AnotherGraph",
          "grant": "read"
        }
      ],
      "max_data_memory_size": "2G"
    }
  ]
}
```

In the above example, user Harry has access to `GlobalGraphIH` and `read` access to `OtherGraph` and `AnotherGraph`. The user, `Peter`, has read access on just `AnotherGraph`.

4. Save the file.

14.8.3 Setup PGX Server without kerberos

You might want to use the PGX server without Kerberos in specific scenarios. In that case, perform the following steps:

1. Navigate to `<PGX_HOME>/bin`
2. Edit the `start-pgx.sh` file and comment out following lines:

   ```
   #export KERBEROS_KEYTAB_BASE_DIRECTORY=$APP_HOME/conf/kerberos
   #export KRB5_CONFIG=$APP_HOME/conf/kerberos/$KRB5_CONFIG_FILENAME
   #bash $APP_HOME/bin/init-kerberos.sh &
   #export HADOOP_CONF_DIR="$APP_HOME/conf/hadoop_cluster"
   ```

3. Ignore these configurations in `config.sh` file:

   ```
   KERBEROS_TICKET_RENEWAL_PERIOD
   KERBEROS_PRINCIPAL
   KERBEROS_KEYTAB_FILENAME
   KRB5_CONFIG_FILENAME
   ```

4. Ignore copying Kerberos-related files and hdfs-libs in respective folders inside `<PGX_HOME>/conf`
5. Follow the remaining steps from setting up the PGX server.

14.8.4 Updating Data Limits and Permissions without Restarting PGX Server

To update the data limits and permissions without restarting the PGX server, perform the following steps:

1. Navigate to Compliance Studio schema.
2. To update the overall data limits, like private, shared, or session limit size in table `fcc_studio_graph_data_limit`

> **NOTE** If you use ratio instead of size, update the column **data_limit_name** to have a ratio-based configuration instead of size.

3. To update the grants and data_limits for user/roles, update the table `fcc_studio_graph_permission_mapping`
   a. Table structure:
      - Id: Sequence number
      - Type: values are either `role` or `user`
      - Entity: Role or User name. Example: DSADMIN or PETER (in case of username)
      - Permission: permissions / resource permission
      - resource_type: Values are `file_location, preloaded_graph, data_limit,` and `null`

        For `null` : static permissions like `pgx_session_new_graph,pgx_session_get_published_graph,`etc.
      - resource name: resource name or value of file location, preloade graph or `data_limit`

   b. Update the Table 63 and set the static and resource permissions as required.

      Example:

Table 63: Static and Resource Permissions

ID	TYPE	ENTITY	PERMISSION	RESOURCE_TYPE	RESOURCE_NAME
1	role	ANALYST	pgx_session_create	-	-
2	role	ANALYST	pgx_session_get_published_graph	-	-
3	role	ANALYST	read	preloaded_graph	GlobalGraphIH
4	role	ANALYST	max_data_memory_size	data_limit	2G
5	user	PETER	pgx_session_create	-	-
6	user	PETER	pgx_session_add_published_graph	-	-
7	user	PETER	pgx_server_manage	-	-

Table 63: Static and Resource Permissions

ID	TYPE	ENTITY	PERMISSION	RESOURCE_-TYPE	RESOURCE_NAME
8	user	PETER	read	file_location	local_storage
9	user	PETER	read	preloaded_graph	AnotherGraph
10	user	PETER	max_data_memory_size	data_limit	2G

4. Navigate to `<PGX_HOME>/conf` and update the `pgx.conf` file with the same details. Refer above sections for additional details.
5. Navigate to `<COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/ficdb/bin.`
6. Run the following command:

   ```
   Export FIC_DB_HOME = <COMPLIANCE_STUDIO_INSTALLATION_PATH>/deployed/
   ficdb
   ```

7. Run the following command to update these without restarting PGX servers:

   ```
   ./FCCM_Studio_ApplyGraphRedaction.sh
   ```

   NOTE

   - `pgx_server_manage` cannot be added at runtime.
   - You must execute the Redaction job twice for the memory limits to reflect without restarting PGX.

8. See the steps detailed in the Checking Data Memory Limit section to verify the data limits.

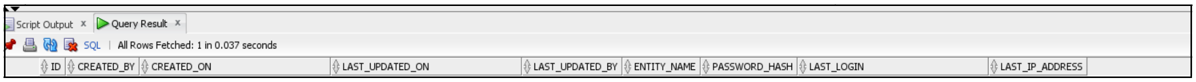
14.9 Checking IP Address for User’s Last Login

Navigate to the Compliance Studio schema in the database and run the following query:

```
select * from ds_user;
```

The output table will look like this:

Figure 132: Output Table



All Rows Fetched: 1 in 0.037 seconds

ID	CREATED_BY	CREATED_ON	LAST_UPDATED_ON	LAST_UPDATED_BY	ENTITY_NAME	PASSWORD_HASH	LAST_LOGIN	LAST_IP_ADDRESS

You can check the LAST_IP_ADDRESS column, which will contain the IP address from where the user has last logged in.

14.10 Roles, Functions and Permissions

14.10.1 Roles

A Role consists of one or more actions (functions/permissions). A Group can have single or multiple roles. For example, Admin, user, and guest. The Table 64 describes the Preconfigured Roles.

Table 64: Roles

Role Code	Role Name	Description
WKSPACC	Workspace Access	Workspace Access Role
WKSPAUTH	Workspace Authorize	Workspace Authorize Role
WKSPREAD	Workspace Read	Workspace Read Role
WKSPWRITE	Workspace Write	Workspace Write Role
FLDRACC	Folder Access	Folder Access Role
FLDRAUTH	Folder Authorize	Folder Authorize Role
FLDRREAD	Folder Read	Folder Read Role
FLDRWRITE	Folder Write	Folder Write Role
IDMGMTACC	Identity MGMT access	System admin access
IDMGMTADVN	Identity MGMT advanced	Identity management advanced
IDMGMTAUTH	Identity MGMT authorize	Identity management authorize
IDMGMTREAD	Identity MGMT read	Identity management read
IDMGMTWRIT	Identity MGMT write	Identity management write
FUNC_READ	Function Read Role	-
FUNC_WRITE	Function Write Role	-
FUNC_ADV	Function Advanced Role	-
ROLE_READ	Role Read Role	-
ROLE_WRITE	Role Write Role	-
ROLE_ADV	Role Advanced Role	-
ROLE_AUTH	Role Authorize Role	-
GRP_READ	Group Read Role	-
GRP_WRITE	Group Write Role	-
GRP_ADV	Group Advanced Role	-
GRP_AUTH	Group Authorize Role	-
USR_READ	User Read Role	-
USR_WRITE	User Write Role	-

Table 64: Roles

Role Code	Role Name	Description
USR_ADV	User Advanced Role	-
USR_AUTH	User Authorize Role	-
SRVC_READ	Service Read Role	-
APP_READ	Application Read Role	-
WRKSP_READ	Workspace Read Role	-
WRKSP_WRITE	Workspace Write Role	-
WRKSP_ADV	Workspace Advanced Role	-
FLDR_READ	Folder Read Role	-
FLDR_WRITE	Folder Write Role	-
FLDR_ADV	Folder Advanced Role	-
DTSRC_READ	Datasource Read Role	-
ADMIN_LINK	Admin Link Role	-
BATCH_READ	Batch Read Role	Batch read role in scheduler service
BATCH_WRITE	Batch Write Role	Batch write role in scheduler service
BATCH_ADV	Batch Advance Role	Batch advance role in scheduler service
BATCH_AUTH	Batch Authorization Role	Batch authorize role in scheduler service
BATCH_OPER	Bath Operation Role	Batch operation role in scheduler service
BATCH_MAINT	Batch Maintenance Role	Batch maintenance role in scheduler service
MDLACCESS	Model Access	User Group mapped will have access to Model Link and Summary
MDLREAD	Model Read	Model Read
MDLWRITE	Model Write	Model Write
MDLPHANTOM	Model Phantom	Model Phantom
MDLAUTH	Model Authorize	Model Authorize
MDLADV	Model Advanced	Model Advanced
MDLREVIEW	Model Review	Model Review
MDLDEPLOY	Model Deployment	Model Deployment
MDLADMIN	Model Admin	Model Admin
SBADMIN	Sandbox Admin	Sandbox Admin

Table 64: Roles

Role Code	Role Name	Description
DSREAD	Datasource Read	Datasource Read
DSWRITE	Datasource Write	Datasource Write
DSACCESS	Datasource Access	Datasource Access
DSADMIN	DSADMIN	Compliance Studio Admin Role
DSBATCH	DSBATCH	Batch Role
DSINTER	DSINTER	Compliance Studio Interpreter Configuration Role
DSUSER	DSUSER	Compliance Studio User Role
DSAPPROVER	DSAPPROVER	Manual Edges Approver role
DSREDACT	DSREDACT	Redaction role for Graph
MDLEXE	Model Execute	Model Execute
MDAPPROVER	MDAPPROVER	Approver
MDREQUESTER	MDREQUESTER	Requester

14.10.1.1 Default Roles Seeded in Notebook Server through permissions-int.yml file

Table 65 describes the Default Roles.

Table 65: Default Roles

Name	Description
DSADMIN	Admin Role (all permissions)
DSBATCH	Batch Role for running ETL and executing notebook using shell script
DSUSER	General Role (does not have access to modify Interpreter configurations or run batches)
DSINTER	Interpreter configurator Role
DSAPPROVER	A role for Approving Manual Edge
DSREDACT	Roles for applying redaction in Graph

14.10.2 Functions in Compliance Studio

Set of actions in the Compliance Studio. For example, limited_read, read, and write. A Role can have single or multiple functions. The Table 66 describes the Preconfigured Functions.

Table 66: Compliance Studio Functions

Role Code	Role Name	Description
WKSP_SUMM	Workspace Summary Access	The user mapped to this function can access the Workspace Summary Pages
WKSP_LNK_ACC	Workspace Link Access	The user mapped to this function can access the Workspace Links
WKSP_AUTH	Workspace Authorization	The user mapped to this function can authorize Workspace
WKSP_VIW	Workspace View	The user mapped to this function can view Workspace
WKSP_ADD	Workspace Add	The user mapped to this function can add Workspace
WKSP_CPY	Workspace Copy	The user mapped to this function can copy Workspace
WKSP_DEL	Workspace Delete	The user mapped to this function can delete Workspace
WKSP_EDIT	Workspace Edit	The user mapped to this function can edit Workspace
FLDR_LNK_ACC	Folder Link Access	The user mapped to this function can access the Folder Links
FLDR_AUTH	Folder Authorization	The user mapped to this function can authorize Folder
FLDR_VIW	Folder View	The user mapped to this function can view the Folder
FLDR_CPY	Folder Copy	The user mapped to this function can copy Folder
FLDR_EDIT	Folder Edit	The user mapped to this function can edit the Folder
ADMINSCR	Administration Screen	The user mapped to this function can access the Administration Screen
FUNCMAINT	Function Maintenance Screen	The user mapped to this function can access the Function Maintenance Screen
FUNCROLE	Function Role Map Screen	The user mapped to this function can access the Function Role Map Screen

Table 66: Compliance Studio Functions

Role Code	Role Name	Description
ROLEMAINT	Role Maintenance Screen	The user mapped to this function can access the Role Maintenance Screen
UGWKSPMAP	User Group Workspace Map Screen	The user mapped to this function can access the User Group Workspace Map Screen
UGFLROLMAP	User Group Folder Role Map Screen	The user mapped to this function can access the User Group Folder Role Map Screen
UGMAINT	User Group Maintenance Screen	The user mapped to this function can access the User Group Maintenance Screen
UGMAP	User Group User Map Screen	The user mapped to this function can access the User Group User Map Screen
UGROLMAP	User Group Role Map Screen	The user mapped to this function can access the User Group Role Map Screen
USRACTREP	User Activity Reports Screen	The user mapped to this function can access the User Activity Reports Screen
USRATTUP	User Attribute Upload Screen	The user mapped to this function can access the User Attribute Upload Screen
USRMAINT	User Maintenance Screen	The user mapped to this function can access the User Maintenance Screen
USRATH	User Authorization Screen	The user mapped to this function can access the User Authorization Screen
FUNC_SUMM	Function Summary	-
FUNC_VIEW	Function View	-
FUNC_ADD	Function Add	-
FUNC_MOD	Function Modify	-
FUNC_DEL	Function Delete	-
FUNC_MAP	Function Map	-
FUNC_PURGE	Function Purge	-
ROLE_SUMM	Role Summary	-
ROLE_VIEW	Role View	-
ROLE_ADD	Role Add	-
ROLE_MOD	Role Modify	-

Table 66: Compliance Studio Functions

Role Code	Role Name	Description
ROLE_DEL	Role Delete	-
ROLE_MAP	Role Map	-
ROLE_PURGE	Role Purge	-
ROLE_AUTH	Role Authorize	-
GRP_SUMM	Group Summary	-
GRP_VIEW	Group View	-
GRP_ADD	Group Add	-
GRP_MOD	Group Modify	-
GRP_DEL	Group Delete	-
GRP_MAP	Group Map	-
GRP_PURGE	Group Purge	-
GRP_AUTH	Group Authorize	-
USR_SUMM	User Summary	-
USR_VIEW	User View	-
USR_ADD	User Add	-
USR_MOD	User Modify	-
USR_DEL	User Delete	-
USR_MAP	User Map	-
USR_PURGE	User Purge	-
USR_AUTH	User Authorize	-
SRVC_SUMM	Service Summary	-
SRVC_VIEW	Service View	-
APP_SUMM	Application Summary	-
APP_VIEW	Application View	-
WRKSP_SUMM	Workspace Summary	-
WRKSP_VIEW	Workspace View	-
WRKSP_ADD	Workspace Add	-
WRKSP_MOD	Workspace Modify	-
WRKSP_DEL	Workspace Delete	-
FLDR_SUMM	Folder Summary	-
FLDR_VIEW	Folder View	-
FLDR_ADD	Folder Add	-

Table 66: Compliance Studio Functions

Role Code	Role Name	Description
FLDR_MOD	Folder Modify	-
FLDR_DEL	Folder Delete	-
DTSRC_SUMM	Datasource Summary	-
DTSRC_VIEW	Datasource View	-
ADMIN_LINK	Admin Link	-
BATCH_ADD	Batch Add Function	Batch add function in scheduler service
BATCH_DEL	Batch Delete Function	Batch delete function in scheduler service
BATCH_MOD	Batch Modify Function	Batch modify the function in scheduler service
BATCH_VIEW	Batch View Function	Batch view function in scheduler service
BATCH_SCH	Batch Schedule Function	Batch schedule function in scheduler service
BATCH_SUMM	Batch Summary Function	Batch summary function in scheduler service
BATCH_AUTH	Batch Authorize Function	Batch authorize function in scheduler service
BATCH_PURGE	Batch Purge Function	Batch purge function in scheduler service
BATCH_MON	Batch Monitor Function	Batch monitor function in scheduler service
BATCH_EXEC	Batch Execute Function	Batch execution function in scheduler service
BATCH_COPY	Batch Copy Function	Batch Copy function in scheduler service
MDLCNFSUMM	Model Configuration Summary	This function gives access to Model Configuration Summary
MDLSUMM	Model Summary	This function gives access to the Model Summary
MDLVIEW	Model View	This function gives access to view Model
MDLTRACE	Model Trace	This function gives access to trace Model
MDLADD	Model Add	This function gives access to add Model
MDLCOPY	Model Copy	This function gives access to copy Model

Table 66: Compliance Studio Functions

Role Code	Role Name	Description
MDLEDIT	Model Edit	This function gives access to edit Model
MDLDEL	Model Delete	This function gives access to delete Model
MDLAPPROVE	Model Approve	This function gives access to approve Model
MDLLOCK	Model Lock	This function gives access to the lock Model
MDLEXE	Model Execute	This function gives access to execute Model
MDLREVIEW	Model Review	This function gives access to review Model
MDLDEPL	Model Deploy	This function gives access to deploying Model
MDLPURGE	Model Purge	This function gives access to purge Model
SBADD	Sandbox Add	This function gives access to add Sandbox
DSADD	Datasource Add	The user mapped to this function can add Datasource
DSEDIT	Datasource Edit	The user mapped to this function can edit Datasource
DSDELETE	Datasource Delete	The user mapped to this function can delete Datasource
DSVIEW	Datasource View	The user mapped to this function can view Datasource
DSSUMM	Datasource Access	The user mapped to this function can access the Datasource summary
MDAPPROVER	MDAPPROVER	To grant approver role
MDREQUESTER	MDREQUESTER	To grant requester role

14.10.3 Permissions in Notebook Server

Set of actions in the Notebook Server. For example, limited_read, read, and write. A Role can have a single or multiple permissions. The Table 67 describes the Preconfigured Permissions.

Table 67: Notebook Server Permissions

Name	Description
*	Do all of the following names
create_notebook	Create a notebook
delete_all	Delete all notebooks in the workspace view

Table 67: Notebook Server Permissions

export_all	Export all notebooks in the Workspace view
graph_create	Create a graph in the Graphs tab
import_notebook	Import a notebook
view_dashboard_tab	View the Tasks tab
view_permissions_tab	View the Permissions tab
view_interpreter_tab	View the Interpreters tab
view_credentials_tab	View the Credentials tab
create_credential	Create a credential
view_visualization_template_tab	View the Visualization Templates tab
visualization_template_create	Create a visualization template
graph_delete	Delete a graph
graph_share	Share a graph
graph_update	Update a graph
graph_view	View a graph
interpreter_create_variant	Create a new interpreter variant
interpreter_update_variant	Update a variant of an interpreter
interpreter_view	View an interpreter
interpreter_variant_execute	Execute an interpreter variant
interpreter_variant_delete	Delete an interpreter variant
interpreter_variant_view	View an interpreter variant
job_cancel	Cancel a job
job_view	View a job
add_relation	Add a relation to a notebook
Attach	(Deprecated) Attach a notebook
Clear	Clear all results in a notebook
Clone	Clone a notebook
Delete	Delete a notebook
Detach	(Deprecated) Detach a notebook
Export	Export a notebook
Iframe	Open a notebook in the iframe view
invalidate_session	Invalidate the session of a notebook
Layout	Change the layout of a notebook
paragraph_comment	Comment on paragraphs in a notebook
paragraph_create	Create a new paragraph in a notebook
paragraph_delete	Delete the paragraphs in a notebook
paragraph_execute	Execute the paragraphs in a notebook
paragraph_modify	Modify the paragraphs in a notebook
paragraph_move	Move the paragraphs in a notebook
paragraph_view	View the paragraphs in a notebook
remove_relation	Remove a relation from a notebook
Rename	Rename a notebook
run_all	Run all paragraphs in a notebook

Table 67: Notebook Server Permissions

schedule_notebook	Schedule a notebook
Share	Share a notebook
set_readonly	Set a notebook to read-only
Snapshot	Take a snapshot (immutable copy) of a notebook
Style	Change the style of a notebook
Template	Add a template to a notebook
toggle_show_code	Toggle the Show Code button in a notebook
toggle_show_result	Toggle the Show Result button in a notebook
Update	Update a notebook
View	View a notebook
view_code	View the code of the paragraphs of a notebook
view_result	View the result of the paragraphs in a notebook
view_sessions	View the sessions of a notebook
create_group	Create a group
create_permission_template	Create a permission template
create_role	Create a role
delete_group	Delete a group
delete_permission_template	Delete a permission template
delete_role	Delete a role
update_group	Update a group
update_permission_template	Update a permission template
update_role	Update a role
update_user	Update a user
view_group	View the Groups section in the Permissions screen
view_permission_template	View the Permission Templates section in the Permissions screen
view_role	View the Roles section in the Permissions screen
view_user	View the Users section in the Permissions screen
view_credential	View a credential and download its file in the credentials screen
use_credential	Use a credential to connect to a data source
delete_credential	Delete a credential from the credentials screen
visualization_template_view	View a visualization template
visualization_template_update	Update a visualization template
visualization_template_delete	Delete a visualization template
visualization_template_share	Share a visualization template
templates_view	View the templates Menu
review_approve (deprecated)	Users can approve the manual similarity edge
review_request (deprecated)	Users can request for approving manual similarity edge
Approve	Users can approve scenario notebook

14.10.4 Group - Role Mapping

Table 68 describes the Preconfigured Groups and the corresponding Roles.

Table 68: Role Mapping

Group Code	Group Name	Role Code	Role Name
DSREDACTGRP	DSREDACTGRP	DSREDACT	DSREDACT
DSUSRGRP	Datastudio User	DSADMIN	DSADMIN
IDNTYADMN	Identity Administrator group	ADMIN_LINK	Admin Link Role
		BATCH_ADV	Batch Advance Role
		BATCH_WRITE	Batch Write Role
		FUNC_ADV	Function Advanced Role
		GRP_ADV	Group Advanced Role
		ROLE_ADV	Role Advanced Role
		USR_ADV	User Advanced Role
IDNTYAUTH	Identity Authorizer group	ADMIN_LINK	Admin Link Role
		FUNC_READ	Function Read Role
		GRP_AUTH	Group Authorize Role
		GRP_READ	Group Read Role
		ROLE_AUTH	Role Authorize Role
		ROLE_READ	Role Read Role
		USR_AUTH	User Authorize Role
MDLAPPR	Modeling Approver	DSAPPROVER	DSAPPROVER
		DSINTER	DSINTER
		MDLACCESS	Model Access
		MDLAUTH	Model Authorize
		MDLDEPLOY	Model Deployment
		MDLREAD	Model Read
		WKSPACC	Workspace Access
		WKSPREAD	Workspace Read
MDLBATCHUSR	Modeling Batch User	DSBATCH	DSBATCH

Table 68: Role Mapping

Group Code	Group Name	Role Code	Role Name
MDLREV	Modeling Reviewer	DSUSER	DSUSER
		MDLACCESS	Model Access
		MDLREAD	Model Read
		MDLREVIEW	Model Review
		WKSPACC	Workspace Access
		WKSPREAD	Workspace Read
MDLUSR	Modeling User	BATCH_ADV	Batch Advance Role
		DSACCESS	Datasource Access
		DSREAD	Datasource Read
		DSUSER	DSUSER
		DSWRITE	Datasource Write
		MDLACCESS	Model Access
		MDLADV	Model Advanced
		MDLEXE	Model Execute
		MDLREAD	Model Read
		MDLWRITE	Model Write
		WKSPACC	Workspace Access
		WKSPREAD	Workspace Read
SANDBOXADM	Sandbox Administrator	SBADMIN	Sandbox Admin
WKSPADMIN	Workspace Administrator	DSADMIN	DSADMIN
		IDMGMTADVN	Identity MGMT advanced
		WKSPACC	Workspace Access
		WKSPAUTH	Workspace Authorize
		WKSPREAD	Workspace Read
		WKSPWRITE	Workspace Write

14.10.5 Role - Function Mapping

Table 69 describes the pre-configured roles and the corresponding Functions.

Table 69: Role - Function Mapping

Role Code	Role Name	Function Code	Function Name
ADMIN_LINK	Admin Link Role	ADMIN_LINK	Admin Link
APP_READ	Application Read Role	APP_SUMM	Application Summary
		APP_VIEW	Application View
BATCH_ADV	Batch Advance Role	BATCH_ADD	Batch Add Function
		BATCH_COPY	Batch Copy Function
		BATCH_DEL	Batch Delete Function
		BATCH_EXEC	Batch Execute Function
		BATCH_MOD	Batch Modify Function
		BATCH_PURGE	Batch Purge Function
		BATCH_SCH	Batch Schedule Function
		BATCH_SUMM	Batch Summary Function
		BATCH_VIEW	Batch View Function
		FUNC_SUMM	Function Summary
BATCH_AUTH	Batch Authorization Role	BATCH_AUTH	Batch Authorize Function
		BATCH_SUMM	Batch Summary Function
		BATCH_VIEW	Batch View Function
		FUNC_SUMM	Function Summary
BATCH_MAINT	Batch Maintenance Role	BATCH_MOD	Batch Modify Function
		BATCH_SUMM	Batch Summary Function
		BATCH_VIEW	Batch View Function
		FUNC_SUMM	Function Summary

Table 69: Role - Function Mapping

Role Code	Role Name	Function Code	Function Name
BATCH_OPER	Bath Operation Role	BATCH_EXEC	Batch Execute Function
		BATCH_SCH	Batch Schedule Function
		BATCH_SUMM	Batch Summary Function
		BATCH_VIEW	Batch View Function
		FUNC_SUMM	Function Summary
BATCH_READ	Batch Read Role	BATCH_SUMM	Batch Summary Function
		BATCH_VIEW	Batch View Function
		FUNC_SUMM	Function Summary
BATCH_WRITE	Batch Write Role	BATCH_ADD	Batch Add Function
		BATCH_COPY	Batch Copy Function
		BATCH_MOD	Batch Modify Function
		BATCH_SUMM	Batch Summary Function
		BATCH_VIEW	Batch View Function
		FUNC_SUMM	Function Summary
DSACCESS	Datasource Access	DSSUMM	Datasource Access
DSAPPROVER	DSAPPROVER	MDAPPROVER	MDAPPROVER
DSREAD	Datasource Read	DSVIEW	Datasource View
DSUSER	DSUSER	MDREQUESTER	MDREQUESTER
DSWRITE	Datasource Write	DSADD	Datasource Add
		DSDELETE	Datasource Delete
		DSEDIT	Datasource Edit
DTSRC_READ	Datasource Read Role	DTSRC_SUMM	Datasource Summary
		DTSRC_VIEW	Datasource View
FLDR_ADV	Folder Advanced Role	FLDR_ADD	Folder Add
		FLDR_DEL	Folder Delete
		FLDR_MOD	Folder Modify
		FLDR_SUMM	Folder Summary
		FLDR_VIEW	Folder View

Table 69: Role - Function Mapping

Role Code	Role Name	Function Code	Function Name
FLDR_READ	Folder Read Role	FLDR_SUMM	Folder Summary
		FLDR_VIEW	Folder View
FLDR_WRITE	Folder Write Role	FLDR_ADD	Folder Add
		FLDR_MOD	Folder Modify
		FLDR_SUMM	Folder Summary
		FLDR_VIEW	Folder View
FLDRACC	Folder Access	FLDR_LNK_ACC	Folder Link Access
FLDRAUTH	Folder Authorize	FLDR_AUTH	Folder Authorization
FLDRREAD	Folder Read	FLDR_VIW	Folder View
FLDRWRITE	Folder Write	FLDR_CPY	Folder Copy
		FLDR_EDIT	Folder Edit
FUNC_ADV	Function Advanced Role	FUNC_ADD	Function Add
		FUNC_DEL	Function Delete
		FUNC_MAP	Function Map
		FUNC_MOD	Function Modify
		FUNC_PURGE	Function Purge
		FUNC_SUMM	Function Summary
		FUNC_VIEW	Function View
FUNC_READ	Function Read Role	FUNC_SUMM	Function Summary
		FUNC_VIEW	Function View
FUNC_WRITE	Function Write Role	FUNC_ADD	Function Add
		FUNC_MOD	Function Modify
		FUNC_SUMM	Function Summary
		FUNC_VIEW	Function View
GRP_ADV	Group Advanced Role	GRP_ADD	Group Add
		GRP_DEL	Group Delete
		GRP_MAP	Group Map
		GRP_MOD	Group Modify
		GRP_PURGE	Group Purge
		GRP_SUMM	Group Summary
		GRP_VIEW	Group View

Table 69: Role - Function Mapping

Role Code	Role Name	Function Code	Function Name
GRP_AUTH	Group Authorize Role	GRP_AUTH	Group Authorize
		GRP_SUMM	Group Summary
		GRP_VIEW	Group View
GRP_READ	Group Read Role	GRP_SUMM	Group Summary
		GRP_VIEW	Group View
GRP_WRITE	Group Write Role	GRP_ADD	Group Add
		GRP_MOD	Group Modify
		GRP_SUMM	Group Summary
		GRP_VIEW	Group View
IDMGMTACC	Identity MGMT access	ADMINSCR	Administration Screen
IDMGMTADVN	Identity MGMT advanced	ADMINSCR	Administration Screen
		FUNCMAINT	Function Maintenance Screen
		FUNCROLE	Function Role Map Screen
		ROLEMAINT	Role Maintenance Screen
		UGFLROLMAP	User Group Folder Role Map Screen
		UGMAINT	User Group Maintenance Screen
		UGMAP	User Group User Map Screen
		UGROLMAP	User Group Role Map Screen
		UGWKSPMAP	User Group Workspace Map Screen
		USRACTREP	User Activity Reports Screen
		USRATTUP	User Attribute Upload Screen
		USRMAINT	User Maintenance Screen
IDMGMTAUTH	Identity MGMT authorize	ADMINSCR	Administration Screen
		USRATH	User Authorization Screen
IDMGMTREAD	Identity MGMT read	ADMINSCR	Administration Screen

Table 69: Role - Function Mapping

Role Code	Role Name	Function Code	Function Name
IDMGMTWRIT	Identity MGMT write	ADMINSCR	Administration Screen
		ROLEMAINT	Role Maintenance Screen
		UGFLROLMAP	User Group Folder Role Map Screen
		UGMAINT	User Group Maintenance Screen
		UGMAP	User Group User Map Screen
		UGROLMAP	User Group Role Map Screen
		UGWKSPMAP	User Group Workspace Map Screen
		USRACTREP	User Activity Reports Screen
		USRATTUP	User Attribute Upload Screen
		USRMAINT	User Maintenance Screen
MDLACCESS	Model Access	MDLCNFSUMM	Model Configuration Summary
		MDLSUMM	Model Summary
MDLADMIN	Model Admin	MDLPURGE	Model Purge
MDLADV	Model Advanced	MDLEXE	Model Execute
		MDLLOCK	Model Lock
MDLAUTH	Model Authorize	MDLAPPROVE	Model Approve
MDLDEPLOY	Model Deployment	MDLDEPL	Model Deploy
MDLREAD	Model Read	MDLTRACE	Model Trace
		MDLVIEW	Model View
MDLREVIEW	Model Review	MDLREVIEW	Model Review
MDLWRITE	Model Write	MDLADD	Model Add
		MDLCOPY	Model Copy
		MDLDEL	Model Delete
		MDLEDIT	Model Edit

Table 69: Role - Function Mapping

Role Code	Role Name	Function Code	Function Name
ROLE_ADV	Role Advanced Role	ROLE_ADD	Role Add
		ROLE_DEL	Role Delete
		ROLE_MAP	Role Map
		ROLE_MOD	Role Modify
		ROLE_PURGE	Role Purge
		ROLE_SUMM	Role Summary
		ROLE_VIEW	Role View
ROLE_AUTH	Role Authorize Role	ROLE_AUTH	Role Authorize
		ROLE_SUMM	Role Summary
		ROLE_VIEW	Role View
ROLE_READ	Role Read Role	ROLE_SUMM	Role Summary
		ROLE_VIEW	Role View
ROLE_WRITE	Role Write Role	ROLE_ADD	Role Add
		ROLE_MOD	Role Modify
		ROLE_SUMM	Role Summary
		ROLE_VIEW	Role View
SBADMIN	Sandbox Admin	SBADD	Sandbox Add
SRVC_READ	Service Read Role	SRVC_SUMM	Service Summary
		SRVC_VIEW	Service View
USR_ADV	User Advanced Role	USR_ADD	User Add
		USR_DEL	User Delete
		USR_MAP	User Map
		USR_MOD	User Modify
		USR_PURGE	User Purge
		USR_SUMM	User Summary
		USR_VIEW	User View
USR_AUTH	User Authorize Role	USR_AUTH	User Authorize
		USR_SUMM	User Summary
		USR_VIEW	User View
USR_READ	User Read Role	USR_SUMM	User Summary
		USR_VIEW	User View

Table 69: Role - Function Mapping

Role Code	Role Name	Function Code	Function Name
USR_WRITE	User Write Role	USR_ADD	User Add
		USR_MOD	User Modify
		USR_SUMM	User Summary
		USR_VIEW	User View
WKSPACC	Workspace Access	WKSP_LNK_ACC	Workspace Link Access
		WKSP_SUMM	Workspace Summary Access
WKSPAUTH	Workspace Authorize	WKSP_AUTH	Workspace Authorization
WKSPREAD	Workspace Read	WKSP_VIW	Workspace View
WKSPWRITE	Workspace Write	WKSP_ADD	Workspace Add
		WKSP_CPY	Workspace Copy
		WKSP_DEL	Workspace Delete
		WKSP_EDIT	Workspace Edit
WRKSP_ADV	Workspace Advanced Role	WRKSP_ADD	Workspace Add
		WRKSP_DEL	Workspace Delete
		WRKSP_MOD	Workspace Modify
		WRKSP_SUMM	Workspace Summary
		WRKSP_VIEW	Workspace View
WRKSP_READ	Workspace Read Role	WRKSP_SUMM	Workspace Summary
		WRKSP_VIEW	Workspace View
WRKSP_WRITE	Workspace Write Role	WRKSP_ADD	Workspace Add
		WRKSP_MOD	Workspace Modify
		WRKSP_SUMM	Workspace Summary
		WRKSP_VIEW	Workspace View

14.10.6 Role - Permission Mapping

Table 70 describes the Preconfigured Roles and the corresponding Permissions.

NOTE The additional Role, **DSREDACT** in Notebook Server, along with the following roles:

- DSADMIN
- DSBATCH
- DSINTER
- DSUSER
- DSAPPROVER

Table 70: Role - Permission Mapping

Permissions	Roles			
	DSADMIN	DSBATCH	DSINTER	DSUSER	DSAPPRROVER
*	Yes				
create_notebook	Yes	Yes	Yes	Yes	
delete_all	Yes	Yes	Yes		
export_all	Yes	Yes	Yes		
graph_create	Yes	Yes	Yes	Yes	
import_notebook	Yes	Yes	Yes	Yes	
view_dashboard_tab	Yes	Yes	Yes	Yes	
view_permissions_tab	Yes		Yes		
view_interpreter_tab	Yes	Yes	Yes	Yes	
view_credentials_tab	Yes	Yes	Yes		
create_credential	Yes	Yes	Yes		
view_visualization_template_tab	Yes	Yes	Yes	Yes	
visualization_template_create	Yes	Yes	Yes	Yes	
graph_delete	Yes	Yes			
graph_share	Yes	Yes			
graph_update	Yes	Yes			
graph_view	Yes	Yes			
interpreter_create_variant	Yes		Yes		
interpreter_update_variant	Yes		Yes		
interpreter_view	Yes	Yes	Yes	Yes	

Table 70: Role - Permission Mapping

interpreter_variant_execute	Yes	Yes	Yes	Yes	
interpreter_variant_delete	Yes		Yes		
interpreter_variant_view	Yes	Yes	Yes	Yes	
job_cancel	Yes	Yes			
job_view	Yes	Yes	Yes	Yes	
add_relation	Yes	Yes	Yes	Yes	
Attach	Yes				
Clear	Yes	Yes	Yes	Yes	
Clone	Yes	Yes	Yes	Yes	
Delete	Yes	Yes	Yes	Yes	
Detach	Yes				
Export	Yes	Yes	Yes	Yes	
Iframe	Yes	Yes	Yes	Yes	
invalidate_session	Yes	Yes	Yes	Yes	
Layout	Yes	Yes	Yes	Yes	
paragraph_comment	Yes	Yes	Yes	Yes	
paragraph_create	Yes	Yes	Yes	Yes	
paragraph_delete	Yes	Yes	Yes	Yes	
paragraph_execute	Yes	Yes	Yes	Yes	
paragraph_modify	Yes	Yes	Yes	Yes	
paragraph_move	Yes	Yes	Yes	Yes	
paragraph_view	Yes	Yes	Yes	Yes	
remove_relation	Yes	Yes	Yes	Yes	
Rename	Yes	Yes	Yes	Yes	
run_all	Yes	Yes	Yes	Yes	
schedule_notebook	Yes	Yes			
Share	Yes	Yes	Yes	Yes	
set_readonly	Yes	Yes	Yes	Yes	
Snapshot	Yes	Yes	Yes	Yes	
Style	Yes	Yes	Yes	Yes	
Template	Yes	Yes	Yes	Yes	
toggle_show_code	Yes	Yes	Yes	Yes	

Table 70: Role - Permission Mapping

toggle_show_result	Yes	Yes	Yes	Yes	
Update	Yes	Yes	Yes	Yes	
View	Yes	Yes	Yes	Yes	
view_code	Yes	Yes	Yes	Yes	
view_result	Yes	Yes	Yes	Yes	
view_sessions	Yes	Yes	Yes	Yes	
create_group	Yes		Yes		
create_permission_template	Yes		Yes		
create_role	Yes		Yes		
delete_group	Yes		Yes		
delete_permission_template	Yes		Yes		
delete_role	Yes		Yes		
update_group	Yes		Yes		
update_permission_template	Yes		Yes		
update_role	Yes		Yes		
update_user	Yes		Yes		
view_group	Yes		Yes		
view_permission_template	Yes		Yes		
view_role	Yes		Yes		
view_user	Yes		Yes		
view_credential	Yes		Yes		
use_credential	Yes		Yes		
delete_credential	Yes		Yes		
visualization_template_view	Yes	Yes	Yes	Yes	
visualization_template_update	Yes	Yes	Yes	Yes	
visualization_template_delete	Yes	Yes	Yes	Yes	
visualization_template_share	Yes	Yes	Yes	Yes	
Approve	Yes	Yes			
review_request	Yes			Yes	
review_approve	Yes				Yes

14.11 Setting Memory of Entity Resolution and Matching Services

To increase the memory of entity resolution and matching services, perform the following steps:

1. Log in to the server where Compliance Studio is installed.
2. Navigate to <`COMPLIANCE_STUDIO_INSTALLATION_PATH>/bin` directory .
3. Open the `compliance-studio.sh` file, and edit the function `start_services()`
4. In entity resolution, update the memory in the `JAVA_OPTS` to a higher value according to your requirement.

   For example,

   ```
   export JAVA_OPTS="-Xms12g -Xmx24g"
   ```

   Code-block:

   ```
   entity-resolution
     export JAVA_OPTS="-Xms4g -Xmx8g"
       export LD_LIBRARY_PATH="$COMPLIANCE_STUDIO_INSTALLATION_PATH/deployed/python-packages/saneVirtualEnv/lib/python3.6/site-packages/jep:$COMPLIANCE_STUDIO_INSTALLATION_PATH/deployed/python-packages/saneVirtualEnv/lib/":$LD_LIBRARY_PATH
       export PATH_ORG=$PATH
       export PATH=$DEPLOY_APP_HOME/python-packages/saneVirtualEnv/bin:$PATH
       export TNS_ADMIN=$TNS_ADMIN_PATH
       sh "$DEPLOY_APP_HOME"/entity-resolution/bin/entity-resolution >"$LOGS_FOLDER"/entity-resolution.log &
       unset JAVA_OPTS
       export PATH=$PATH_ORG
       ;;
   ```

5. In the matching service, update the memory in the **JAVA_OPTS** to a higher value according to your requirement.

   For example,

   ```
   export JAVA_OPTS="-Xms12g -Xmx24g"
   ```

   Code-block:

   ```
   matching-service
       export JAVA_OPTS="-Xms6g -Xmx12g"
       export LD_LIBRARY_PATH="$COMPLIANCE_STUDIO_INSTALLATION_PATH/deployed/python-packages/saneVirtualEnv/lib/python3.6/site-packages/jep:$COMPLIANCE_STUDIO_INSTALLATION_PATH/deployed/python-packages/saneVirtualEnv/lib/":$LD_LIBRARY_PATH
       if ("$ELASTIC_SEARCH_HTTPS_ENABLED"); then
   ```

```
      export JAVA_OPTS="$JAVA_OPTS -
Djavax.net.ssl.trustStore=$DEPLOY_APP_HOME/matching-service/conf/
$ELASTIC_SEARCH_TRUSTSTORE_FILE_NAME

      -
Djavax.net.ssl.trustStorePassword=$ELASTIC_SEARCH_TRUSTSTORE_PASSWORD"

    fi

    export PATH_ORG=$PATH

    export PATH=$DEPLOY_APP_HOME/python-packages/saneVirtualEnv/
bin:$PATH

    export TNS_ADMIN=$TNS_ADMIN_PATH

    sh "$DEPLOY_APP_HOME"/matching-service/bin/matching-service
>"$LOGS_FOLDER"/matching-service.log &

    unset JAVA_OPTS

    export PATH=$PATH_ORG

    ;;
```

14.12 Clean-up Steps When the Create Index and Load Data Job Terminated Manually

Perform the following clean-up steps when you terminate the **Create Index and Load Data** job manually:

> **ATTENTION** Ensure that you have set the DBMS Server Output as **ON**.
>
> To set it via SQL command prompt.
>
> Run the following command in SQL prompt:
>
> `SET SERVEROUTPUT ON;`
>
> To set it via Oracle SQL Developer, perform the following:
>
> 1. Open the Oracle SQL Developer tool.
> 2. Navigate to **View** > **Dbms Output**. The DBMS OUTPUT panel is displayed.
> 3. Click the Plus ➕ icon. The Select Connect window is displayed.
> 4. From the **Connection** drop-down list, select the connection. Click **OK**. The Connection Information Window is displayed.
> 5. Enter the **Username** and **Password** to enable your selected schema connection.

NOTE

- Clean-up is not required when the job fails with a logged exception. In some scenarios, where the batch failed with logged exception but the system is unable to perform any further actions; for instance, a database connectivity issue, wherein the database is unable to provide any more connections. In such situations, the clean-up process would not happen and should be run explicitly.
- Cleanup scripts are executed based on the **n_run_skey** and **fic_mis_date**.

The following clean-up steps are used to clean up the failure for **Day 0 (Initial Run)** as well as subsequent execution:

- `Batch_Details_For_Create_And_Load_Index.sql`: It provides the previous **runSkey** (**n_run_skey** column) and the latest **fic_mis_date** for the given latest **runSkey** as input.
- `Cleanup_ER_Create_And_Load_Data_into_Index.sql`**:** This script accepts the outputs from `Batch_Details_For_Create_And_Load_Index.sql` as input and cleans up failed batch execution.

  Perform the following to run the `Batch_Details_For_Create_And_Load_Index.sql`:

1. Download the script from My Oracle Support - Doc ID 2813823.1.
2. Log in to the Compliance Studio Schema.
3. Copy the script to the machine where you need to execute the script.
4. Run the following command in SQL prompt:

   ```
   @<Fully Qualified path of the Script>/
   Batch_Details_For_Create_And_Load_Index.sql
   ```

   Enter the inputs as per the prompt.

   The input for this script is **runSkey**, for which the batch cleanup is to be performed. You can get the **runSkey** from the batch execution console messages by searching the text "**Current runSkey:**"
5. Press **Enter.**

   On successful execution, the script will exit with the message "Successfully retrieved PREVIOUS_RUN_SKEY and CURRENT_FIC_MIS_DATE".

NOTE You can note the values for **CURRENT_FIC_MIS_DATE**, **CURRENT_RUN_SKEY,** and **PREVIOUS_RUN_SKEY,** respectively. These values are used as inputs to run the `Cleanup_ER_Create_And_Load_Data_into_Index.sql`.

Perform the following to run the `Cleanup_ER_Create_And_Load_Data_into_Index.sql`:

NOTE Ensure you have captured the values for CURRENT_FIC_MIS_DATE, CURRENT_RUN_SKEY, and PREVIOUS_RUN_SKEY, respectively, from the `Batch_Details_For_Create_And_Load_Index.sql` script results.

1. Download the script from My Oracle Support - Doc ID 2813823.1.
2. Log in to the ER Schema (The schema where input tables of Entity Resolution are available).
3. Copy the script to the machine where you need to execute the script.
4. Run the following command in SQL prompt:

   ```
   @<Fully Qualified path of the Script>/
   Cleanup_ER_Create_And_Load_Data_into_Index.sql
   ```

   Enter the inputs as per the prompt.

   The inputs for this script are **CURRENT_FIC_MIS_DATE**, **CURRENT_RUN_SKEY,** and **PREVIOUS_RUN_SKEY** values that were received from `Batch_Details_For_Create_And_Load_Index.sql`.

   To reset the ER Schema for **Day 0** execution, the input value for **V_RESET_FLAG** should be **'Y'** else **'N'** (including single quotes).

NOTE	• The **FCC_ER_FULL** and **History** tables are dropped when you run this script with the **V_RESET_FLAG** value as **'Y'**: • You must drop the **FCC_ER_FULL** table if any changes are made in metadata to alter the table.

5. Press **Enter**.

   On successful execution, the script will exit with the message "Cleanup Successfully Completed".

14.13 Clean-up Steps When the Bulk Similarity Job Terminated Manually

Perform the following clean-up steps when you terminate the ER Bulk Similarity job manually.

To clean up the ER Bulk Similarity job execution in Compliance Studio Schema, perform the following:

1. Log in to the Compliance Studio Schema.
2. Run the following command in SQL prompt to update the status in the **n_run_status** column in the **FCC_BATCH_RUN** table in Compliance Studio Schema for corresponding **runSKey** (**n_run_skey**).

   ```
   UPDATE FCC_BATCH_RUN SET N_RUN_STATUS=2 WHERE
   N_RUN_SKEY=<CURRENT_RUN_SKEY>;
   ```

   ```
   COMMIT;
   ```

   Where `<CURRENT_RUN_SKEY>` is the **runSkey** for which the job is to be re-executed.

   For example,

   ```
   UPDATE FCC_BATCH_RUN SET N_RUN_STATUS=2 WHERE N_RUN_SKEY=112;
   ```

   ```
   COMMIT;
   ```

Perform the following additional clean-up steps for **Day 0** execution:

1. Log in to the ER Schema.

2. Run the following commands in SQL prompt:

```
truncate table FCC_ER_MATCHING;

truncate table FCC_ER_MANUAL_MATCH;

truncate table FCC_ER_MANUAL_MAPPING;
```

14.14 Clean-up Steps When the Data Survival Job Terminated Manually

Perform the following clean-up steps when you terminate the Data Survival job manually.

To clean up the ER Data Survival Engine job execution in Compliance Studio Schema, perform the following:

1. Log in to the Compliance Studio Schema.
2. Run the following command in SQL prompt to update the status in the **n_run_status** column in the **FCC_BATCH_RUN** table in Compliance Studio Schema for corresponding **runSkey** (**n_run_skey**).

   ```
   UPDATE FCC_BATCH_RUN SET N_RUN_STATUS=4 WHERE
   N_RUN_SKEY=<CURRENT_RUN_SKEY>;

   COMMIT;
   ```

   Where `<CURRENT_RUN_SKEY>` is the **runSkey** for which the job is to be re-executed.

   For example,

   ```
   UPDATE FCC_BATCH_RUN SET N_RUN_STATUS=4 WHERE N_RUN_SKEY=112;

   COMMIT;
   ```

Perform the following additional clean-up steps for **Day 0, Day 1/Subsequent Day** execution:

1. Log in to the ER Schema.
2. Run the following commands in SQL prompt for **Day 0**:

   ```
   truncate table FCC_ER_MAPPING;

   truncate table FCC_ER_AUDIT_LOGS;

   truncate table STG_PARTY_MASTER;

   truncate table STG_PARTY_DETAILS;

   truncate table STG_PARTY_EMAIL_MAP;

   truncate table STG_PARTY_PHONE_MAP;

   truncate table STG_ADDRESS_MASTER;

   truncate table STG_PARTY_ADDRESS_MAP;

   truncate table STG_CUSTOMER_IDENTIFCTN_DOC;

   truncate table STG_ADDRESS_MASTER_HIST;

   truncate table STG_CUSTOMER_IDENTIFCTN_DOC_HIST;

   truncate table STG_PARTY_ADDRESS_MAP_HIST;

   truncate table STG_PARTY_DETAILS_HIST;

   truncate table STG_PARTY_EMAIL_MAP_HIST;
   ```

```
truncate table STG_PARTY_MASTER_HIST;
truncate table STG_PARTY_PHONE_MAP_HIST;
truncate table FCC_ER_MAPPING_ARC;
truncate table FCC_ER_MAPPING_BKP;
truncate table FCC_ER_MAPPING_STG_TEMP;
truncate table FCC_ER_MAPPING_TEMP;
truncate table FCC_ER_TEMP_MAPPING;
```

3. Run the following commands in SQL prompt for **Day 1/Subsequent Day**:

```
DELETE /*+ PARALLEL(8) */ FROM STG_PARTY_MATER WHERE FIC_MIS_DATE=
TO_DATE('<FIC_MIS_DATE>','yyyyMMdd');

DELETE /*+ PARALLEL(8) */ FROM STG_PARTY_DETAILS WHERE FIC_MIS_DATE=
TO_DATE('<FIC_MIS_DATE>','yyyyMMdd');

DELETE /*+ PARALLEL(8) */ FROM STG_PARTY_PHONE_MAP WHERE FIC_MIS_DATE=
TO_DATE('<FIC_MIS_DATE>','yyyyMMdd');

DELETE /*+ PARALLEL(8) */ FROM STG_PARTY_EMAIL_MAP WHERE FIC_MIS_DATE=
TO_DATE('<FIC_MIS_DATE>','yyyyMMdd');

DELETE /*+ PARALLEL(8) */ FROM STG_CUSTOMER_IDENTIFCTN_DOC WHERE
FIC_MIS_DATE= TO_DATE('<FIC_MIS_DATE>','yyyyMMdd');

DELETE /*+ PARALLEL(8) */ FROM STG_ADDRESS_MASTER WHERE FIC_MIS_DATE=
TO_DATE('<FIC_MIS_DATE>','yyyyMMdd');

DELETE /*+ PARALLEL(8) */ FROM STG_PARTY_ADDRESS_MAP WHERE FIC_MIS_DATE=
TO_DATE('<FIC_MIS_DATE>','yyyyMMdd');

COMMIT;
```

Where `FIC_MIS_DATE` is the current ficmisdate in yyyyMMdd format.

For example,

```
DELETE /*+PARALLEL(8) */ FROM STG_PARTY_MASTER WHERE FIC_MIS_DATE=
TO_DATE('20151211','yyyyMMdd');

DELETE /*+PARALLEL(8) */ FROM STG_PARTY_DETAILS WHERE FIC_MIS_DATE=
TO_DATE('20151211','yyyyMMdd');

DELETE /*+PARALLEL(8) */ FROM STG_PARTY_PHONE_MAP WHERE FIC_MIS_DATE=
TO_DATE('20151211','yyyyMMdd');

DELETE /*+PARALLEL(8) */ FROM STG_PARTY_EMAIL_MAP WHERE FIC_MIS_DATE=
TO_DATE('20151211','yyyyMMdd');

DELETE /*+PARALLEL(8) */ FROM STG_CUSTOMER_IDENTIFCTN_DOC WHERE
FIC_MIS_DATE= TO_DATE('20151211','yyyyMMdd');

DELETE /*+PARALLEL(8) */ FROM STG_ADDRESS_MASTER WHERE FIC_MIS_DATE=
TO_DATE('20151211','yyyyMMdd');
```

```
DELETE /*+PARALLEL(8) */ FROM STG_PARTY_ADDRESS_MAP WHERE FIC_MIS_DATE=
TO_DATE('20151211','yyyyMMdd');
```

Where `20151211` is the FIC_MIS_DATE of the 11th Dec 2015.

14.15 Clean-up Steps When the Load Data in FCC_ER_OUTPUT Job Terminated Manually

Perform the following clean-up steps when you terminate the Load Data in the FCC_ER_OUTPUT job manually.

ATTENTION Ensure that you have set the DBMS Server Output as **ON**.

To set it via SQL command prompt.

Run the following command in SQL prompt:

```
SET SERVEROUTPUT ON;
```

To set it via Oracle SQL Developer, perform the following:

1. Open the Oracle SQL Developer tool.
2. Navigate to **View** > **Dbms Output**. The DBMS OUTPUT panel is displayed.
3. Click the Plus + icon. The Select Connect window is displayed.
4. From the **Connection** drop-down list, select the connection. Click **OK**. The Connection Information Window is displayed.
5. Enter the **Username** and **Password** to enable your selected schema connection.

To update job execution status in Compliance Studio Schema, perform the following:

1. Log in to the Compliance Studio Schema.
2. Update status in the **n_run_status** column to **6** in the **FCC_BATCH_RUN** table in Compliance Studio Schema for current **runSKey** (**n_run_skey**).

   ```
   UPDATE FCC_BATCH_RUN SET N_RUN_STATUS=6 WHERE
   N_RUN_SKEY=<CURRENT_RUN_SKEY>;
   ```

   ```
   COMMIT;
   ```

   Where `<CURRENT_RUN_SKEY>` is the **runSkey** for which the job is to be executed.

   For example,

   ```
   UPDATE FCC_BATCH_RUN SET N_RUN_STATUS=6 WHERE N_RUN_SKEY=112;
   ```

   ```
   COMMIT;
   ```

To perform clean-up in ER Schema for the failed job:

1. Download `Cleanup_ER_Run_Full_Data_Output.sql` script from My Oracle Support - Doc ID 2813823.1.
2. Log in to ER Schema (The schema where input tables of Entity Resolution are available).

3. Copy the script to the machine where you need to execute the script.
4. Run the following command in SQL prompt:

   ```
   @<Fully Qualified path of the Script>/
   Cleanup_ER_Run_Full_Data_Output.sql
   ```

5. Press **Enter.**

   On successful execution, the script exits with the message: "Cleanup successfully completed."

14.16 Resetting Entity Resolution Back to Day 0

ATTENTION

- This section is applicable only when you wipe out ER-related tables and indexes. This will bring the Entity Resolution back to **Day 0**.
- You can clean up the ER Schema after upgrading from **v8.1.2.0.0** to **v8.1.2.0.1** or restart ER with different rules.

14.16.1 ER Schema Changes

1. Run **Create Index and Load Data** job clean-up script for the latest **runSkey** where this job is executed. For **Create Index and Load Data** job clean-up script, see the Clean-up Steps When the Create Index and Load Data Job Terminated Manually section.

NOTE The value for input parameter **V_RESET_FLAG** should be **'Y'** (including single quotes).

2. Log in to ER Schema.
3. Run the following SQL queries to drop the specified tables:

   ```
   DROP TABLE er_audit;

   DROP TABLE er_jobs;

   DROP TABLE fcc_er_audit_logs;

   DROP TABLE fcc_er_full;

   DROP TABLE fcc_er_gp_comments;

   DROP TABLE fcc_er_m_bkp_config;

   DROP TABLE fcc_er_manual_match;

   DROP TABLE fcc_er_manual_match_temp;

   DROP TABLE fcc_er_mapping_arc;

   DROP TABLE fcc_er_mapping_bkp;

   DROP TABLE fcc_er_mapping_stg_temp;

   DROP TABLE fcc_er_mapping_temp;

   DROP TABLE fcc_er_matching;

   DROP TABLE fcc_er_matching_deleted;

   DROP TABLE fcc_er_matching_temp;
   ```

```
DROP TABLE fcc_er_output;
DROP TABLE fcc_er_temp_mapping;
DROP TABLE fcc_er_affected_similarity_ids;
DROP TABLE fcc_er_mapping_delta;
DROP TABLE fcc_er_batch_check;
DROP TABLE fcc_er_mapping_delta_temp_3;
DROP TABLE fcc_er_mapping_delta_temp_2;
DROP TABLE fcc_er_full_delta_temp_1;
DROP TABLE ER_PERFORMANCE_TIME_PROFILER;
DROP TABLE er_p_parallel_run;
DROP TABLE fcc_er_config;
DROP TABLE fcc_er_manual_map;
DROP TABLE h$stg_address_master_pre;
DROP TABLE h$stg_customer_identifctn_doc_pre;
DROP TABLE h$stg_party_address_map_pre;
DROP TABLE h$stg_party_details_pre;
DROP TABLE h$stg_party_email_map_pre;
DROP TABLE h$stg_party_master_pre;
DROP TABLE h$stg_party_phone_map_pre;
DROP TABLE stg_address_master_dlta;
DROP TABLE stg_customer_identifctn_doc_dlta;
DROP TABLE stg_party_address_map_dlta;
DROP TABLE stg_party_details_dlta;
DROP TABLE stg_party_email_map_dlta;
DROP TABLE stg_party_master_dlta;
DROP TABLE stg_party_phone_map_dlta;
DROP TABLE stg_address_master_hist;
DROP TABLE stg_customer_identifctn_doc_hist;
DROP TABLE stg_party_address_map_hist;
DROP TABLE stg_party_details_hist;
DROP TABLE stg_party_email_map_hist;
DROP TABLE stg_party_master_hist;
DROP TABLE stg_party_phone_map_hist;
DROP TABLE FCC_ER_MAPPING_DELTA_HIST;
DROP TABLE FCC_ER_MAPPING_DELTA_TEMP_CURR_RUN;
DROP TABLE FCC_ER_MAPPING_DELTA_ARC;
DROP TABLE FCC_ER_MAPPING_HIST;
```

```
DROP TABLE FCC_ER_POPULATE_OUTPUT;
```

NOTE The following indices will be deleted automatically when you drop the respective tables:

- fcc_er_manual_match_ind
- fcc_er_matching_ind
- fcc_er_matching_temp_ind

If indices still exist, run the following SQL queries to drop the Indices:

```
DROP INDEX fcc_er_manual_match_ind;
```

```
DROP INDEX fcc_er_matching_ind;
```

```
DROP INDEX fcc_er_matching_temp_ind;
```

If following tables still exist, run the SQL queries to drop the tables:

- stg_party_master_hist
- stg_party_details_hist
- stg_party_email_map_hist
- stg_party_phone_map_hist
- stg_address_master_hist
- stg_party_address_map_hist
- stg_customer_identifctn_doc_hist

Run the following command to drop tables if they exist in the database:

```
DROP MATERIALIZED VIEW <STG_TABLE_NAME>_HIST;
```

For example,

```
DROP MATERIALIZED VIEW STG_PARTY_MASTER_HIST;
```

Run the following command to drop tables only when table names are changed to LOG_<STG_TABLE_NAME>_HIST.

```
DROP MATERIALIZED VIEW LOG ON <STG_TABLE_NAME>_HIST;
```

For example,

```
DROP MATERIALIZED VIEW LOG ON STG_PARTY_MASTER_HIST;
```

4. Run the following SQL queries to drop the specified sequences:

```
DROP SEQUENCE fcc_er_bkp_config_id_seq;
```

```
DROP SEQUENCE fcc_er_audit_seq;
```

```
DROP SEQUENCE data_survival_realtime_sequence;
```

5. Run the following SQL queries to truncate the specified tables:

```
TRUNCATE TABLE fcc_er_manual_mapping;
```

```
TRUNCATE TABLE fcc_er_mapping;
```

```
TRUNCATE TABLE stg_party_master;
TRUNCATE TABLE stg_party_details;

TRUNCATE TABLE stg_party_email_map;

TRUNCATE TABLE stg_party_phone_map;

TRUNCATE TABLE stg_address_master;

TRUNCATE TABLE stg_party_address_map;

TRUNCATE TABLE stg_customer_identifctn_doc;
```

6. After executing steps 1 to 5, verify if any temporary tables exist using the below query:

```
SELECT TEMP.TABLE_NAME

FROM

(SELECT TABLE_NAME FROM USER_TABLES A

WHERE (A.TABLE_NAME LIKE '%_CHUNK%' OR A.TABLE_NAME LIKE '%_PRE_P_%' OR
A.TABLE_NAME LIKE '%_DLTA_%' OR A.TABLE_NAME LIKE '%_OUT_P%')) TEMP

INNER JOIN

(SELECT SUBSTR(TABLE_NAME, 1, LENGTH(TABLE_NAME)-4) TABLE_NAME FROM
USER_TABLES WHERE TABLE_NAME LIKE 'STG_%_PRE') STAGE

ON TEMP.TABLE_NAME LIKE '%'|| STAGE.TABLE_NAME ||'%';
```

If any tables exist, drop those tables after confirming manually that the table belongs to Entity Resolution.

14.16.2 Compliance Studio Schema Changes

To truncate batch run tables, perform the following:

1. Log in to Compliance Studio Schema.
2. Check the **FCC_BATCH_RUN** table in the Compliance Studio schema and if there are any records exist, run the following command to truncate the table before executing the ER jobs:

```
truncate table FCC_BATCH_RUN;
```

14.16.3 Elastic Search Changes

To clean up ER staging indexes, perform the following:

1. Log in to the server where Compliance Studio is installed.
2. Run the following **curl** command:

```
curl -XDELETE http://hostname:port/load-to-elastic-search/idx/
deleteIndex/<Index name>
```

For example,

```
curl -XDELETE http://testserver.oracle.com:7053/load-to-elastic-search/
idx/deleteIndex/stg_party_812
```

3. Repeat **Step 2** if multiple ER indexes, run with respective staging index names.

14.17 Utility Scripts

14.17.1 Data Slicing Utility Script

The Data Slicing Utility is a SQL script to perform data slicing (slicing the data into different chunks or data units) according to the user input (FIC_MIS_DATE). It helps faster turn-around time for individual batches as the load is moderately low.

FIC_MIS_DATE is the execution identifier for Entity Resolution, and it is easy to distribute records into different FIC_MIS_DATE values.

You can perform the data slicing for a high volume of data, which takes a long time and more resources based on your database performance.

> **NOTE** This utility is used for slicing the data in the following input tables of the out-of-the-box rules for Entity Resolution:
>
> - STG_PARTY_MASTER_PRE
> - STG_PARTY_DETAILS_PRE
> - STG_PARTY_EMAIL_MAP_PRE
> - STG_PARTY_PHONE_MAP_PRE
> - STG_CUSTOMER_IDENTIFCTN_DOC_PRE
> - STG_PARTY_ADDRESS_MAP_PRE
> - STG_ADDRESS_MASTER_PRE

The utility distributes the data into logical units based on the criteria (user input), resulting in multiple data chunks.

- It accepts comma-separated **FIC_MIS_DATE** as user input.

  For example. 20150101,20150102,20150103
- It distributes the records across the **FIC_MIS_DATE** equally. The last slice should contain additional records if there are any.

> **ATTENTION** It is recommended that you must split the data into slices of a maximum of 10 million records.
>
> Here is a scenario of data slicing:
>
> - Input data volume: **50 million**
> - Size of slice on which job has to execute: **10 million**
> - Total number of slices: **5** (different comma-separated **FIC_MIS_DATE)**

After the utility completes the distribution, you can perform the ER batch execution as follows:

1. Provide the chunk as **Day 0** load corresponding to the respective **FIC_MIS_DATE**.
2. Provide subsequent chunks such as **Day 1**, **Day 2**, etc. These chunks are treated as delta loads (delta having only new records).

To execute the utility script, perform the following:

1. Download `DataSlicingUtility.sql` script from My Oracle Support (Doc ID 2813823.1).

2. Log in to the ER Schema. The schema (input tables of Entity Resolution) is available.
3. Copy the script to the machine where you need to execute the script.
4. Run the following command in SQL prompt:

   ```
   @<Fully Qualified path of Utility Script>/DataSlicingUtility.sql
   ```

5. Enter the values according to the following prompt:

   ```
   Enter value for fic_mis_date:
   ```

   You need to enter comma-separated **FIC_MIS_DATE** in **YYYYMMDD** format.

   For example, 20150101,20150102,20150103

6. Press **Enter**.
   - On successful execution, the utility scripts exits with a success message "FIC_MIS_DATEs have applied for all <list of fic_mis_dates> slices"

     For example,

     ```
     SQL> @<path of the script>/DataSlicingUtility.sql

     Enter value for fic_mis_date:
     20150107,20150108,20150109,20150110,20150115

     old  24: FIC_MIS_DATES:='&FIC_MIS_DATE';

     new  24:
     FIC_MIS_DATES:='20150107,20150108,20150109,20150110,20150115';

     PL/SQL procedure successfully completed.
     ```

   - On failure, displays the appropriate error message.
7. You can validate the results of successful execution:
   - For each input table, check the count of records against **FIC_MIS_DATE**.

     Run the following commands to check the count in each input table. Perform the same for all input tables:

     ```
     SELECT DISTINCT FIC_MIS_DATE, COUNT(*) FROM <INPUT TABLE NAME> GROUP
     BY FIC_MIS_DATE;
     ```

     For example,

     ```
     SELECT DISTINCT FIC_MIS_DATE, COUNT(*) FROM STG_PARTY_MASTER_PRE
     GROUP BY FIC_MIS_DATE;
     ```

   - Ensure that complete information for a particular party is included in the same slice.

     For example, for any **V_PARTY_ID**, there should be the same **FIC_MIS_DATE** tagged in each input table.

14.18 Disable the User in Compliance Studio after SSO Login

To revoke the mapped CS Groups for a particular user in the Compliance Studio, follow these steps:

In SAML IDCS, Admin has to remove the Groups for a particular user.

1. Login to IDCS as **Admin**.
2. Navigate to **Users** tab and select the **User**.

3. Navigate to **Groups** tab and select the groups to be revoked.
4. Click **Revoke** Button.
5. Click **Save** to modify the changes.

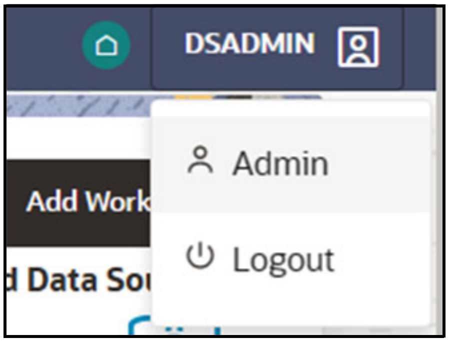
In Compliance Studio,

1. Login to Compliance Studio as **Admin User**.

NOTE	Admin users should have access to Identity Management.

2. Navigate to **Identity Management** and click **Users**.

Figure 133: Identity Management



3. Select the same user of the Groups that are removed from the IDCS.
4. Navigate to **Mapped Groups** tab and select the Groups to be revoked.
5. Click **Unmap**.
6. Login as another **Admin User** who can authorize the above changes.

NOTE	Any other user with admin access can authorize.

7. Navigate to Identity Management as **Authorizing User.**
8. Click **Users** and select the same user of the Groups that are removed from the IDCS.
9. Navigate to **Mapped Groups** tab and move the toggle switch to the right to enable **Authorization View**.
10. Select all the groups and click **Authorize** button.
11. Restart the Compliance Studio.

OFSAA Support

Raise a Service Request (SR) in My Oracle Support (MOS) for queries related to OFSAA applications.

Send Us Your Comments

Oracle welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this manual?

If you find any errors or have any other suggestions for improvement, indicate the title and part number of the documentation along with the chapter/section/page number (if available) and contact the Oracle Support.

Before sending us your comments, you might like to ensure that you have the latest version of the document wherein any of your concerns have already been addressed. You can access My Oracle Support site which has all the revised/recently released documents.

ORACLE®