

**Oracle® Financial Services Crime and Compliance
Studio**

Administration and Configuration Guide

Release 8.0.7.0.0

E91246-01

February 2019

Administration and Configuration Guide, Release 8.0.7.0.0

E91246-01

Copyright © 2019 Oracle and/or its affiliates. All rights reserved.

Primary Author: Nethravathi G

Contributor: Pankaj Chhangwani, Amit Prasad, Partik Davda

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, 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 on 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. 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.

Contents

Document Control	ix
About this Guide	xi
Who Should Use this Guide.....	xi
Scope of this Guide	xi
How this Guide is Organized.....	xii
Where to Find More Information.....	xii
Conventions Used in this Guide	xii
Abbreviations Used in this Guide.....	xii
1 About Oracle Financial Services Crime and Compliance Studio	
Introduction to Crime and Compliance Studio.....	1-2
Salient Features of Crime and Compliance Studio	1-2
Architecture of Crime and Compliance Studio.....	1-3
PGX Graphs and Data Source Configuration.....	1-3
2 Managing User Administration	
Managing Identity and Authorization	2-1
Identity and Authorization Process Flow.....	2-1
Creating and Authorizing User	2-2
Mapping User with User Group	2-2
Granting Permissions	2-2
3 Managing Studio Batches	
Preparing for Batches.....	3-1
Performing Batches	3-1
Data Movement and Graph Loading for Oracle Database	3-1
Moving Data	3-2
Loading Graph	3-2
Data Movement and Graph Loading for Big Data Environment	3-2
Executing Published Scenario Notebook.....	3-3
Creating and Executing Run Executable	3-3
4 Configuring Pre-Seeded Graph and Notebook	
Configuring Pre-seeded Graph in Studio	4-1
Configuring Pre-seeded Notebooks in Studio	4-1
5 Configuring Security for PGX	
Prepare Certificates	5-1
Create Self-Signed Server Certificate	5-1

Create New Keystore.....	5-2
Extract the Certificate	5-2
Import Existing Certificate or Install Certificate from a Certificate Authority	5-3
Prepare Client Keystore	5-3
Prepare Client Truststore	5-4
Configure PGX Web Server.....	5-4
Test Connection Using PGX Client Shell.....	5-4

6 Configuring Interpreters

Accessing Interpreters	6-1
Create New Interpreter Variant.....	6-2
Configure Interpreters.....	6-2
fcc-jdbc Interpreter	6-3
fcc-ore Interpreter.....	6-3
fcc-pyspark Interpreter.....	6-5
fcc-python Interpreter.....	6-6
fcc-spark-scala Interpreter	6-7
fcc-spark-sql Interpreter	6-8
md Interpreter.....	6-9
pgx Interpreter.....	6-9

List of Figures

1-1	Crime and Compliance Studio in FCCM Architecture	1-1
1-2	Crime and Compliance Studio Architecture.....	1-3
2-1	Managing Identity and Authorization Process Flow	2-1
6-1	Interpreters Page	6-1

List of Tables

0-1	Conventions Used in this Guide.....	3-xii
0-2	Abbreviations and their meaning.....	3-xii
2-1	User Administration Process Flow.....	2-2
2-2	Roles and User Groups in Studio	2-2
3-1	Adding Run Definition	3-4
3-2	List of parameter for batch execution	3-5
3-3	Adding Fire Run Details	3-7
6-1	fcc-jdbc interpreter Values.....	6-3
6-2	fcc-ore Interpreter Values	6-4
6-3	fcc-pyspark Interpreter Values	6-5
6-4	fcc-python Interpreter Values	6-6
6-5	fcc-spark-scala Interpreter Values	6-7
6-6	fcc-spark-sql Interpreter Values.....	6-8
6-7	md Interpreter Values	6-9
6-8	pgx Interpreter Values.....	6-9

Document Control

This section provides the revision details of the document.

Version Number	Revision Date	Changes Done
8.0.7.0.0	Created: Feb 2019	Created first version of Crime and Compliance Studio Administration Guide for 8.0.7.0.0 Release.

This document provides functional information about the Crime and Compliance Studio application and enables you to navigate through the various sections of the application. The latest copy of this guide can be accessed from the Oracle Help Center ([OHC](#)) Documentation Library.

About this Guide

This guide explains the concepts for the Oracle Financial Services Crime and Compliance Studio application and provides comprehensive instructions for system administration, as well as for daily operations and maintenance. This section focuses on the following topics:

- [Who Should Use this Guide](#)
- [Scope of this Guide](#)
- [How this Guide is Organized](#)
- [Where to Find More Information](#)
- [Conventions Used in this Guide](#)
- [Abbreviations Used in this Guide](#)

Who Should Use this Guide

This guide is intended for administrators and implementation consultants. Their roles and responsibilities, as they operate within Studio, include the following:

- **Implementation Consultant:** Installs and configures Crime and Compliance Studio application at a specific deployment site. The consultant also installs and upgrades any additional Oracle Financial Services solution sets, and requires access to deployment-specific configuration information. For example, machine names and port numbers.
- **System Administrator:** Configures and maintains the system. The System Administrator maintains user accounts and roles, monitors data management, archives data, loads data feeds, and performs post-processing tasks. In addition, the System Administrator also reloads cache.

Scope of this Guide

This guide describes the physical and logical architecture of Crime and Compliance Studio application. It also provides instructions for maintaining and configuring Studio, its subsystem components, and any third-party software required for operation.

Crime and Compliance Studio provides an open and scalable infrastructure that supports rich, end-to-end functionality across all Oracle Financial Services solution sets. Studio's extensible, modular architecture enables a customer to deploy new solution sets readily as the need arises.

How this Guide is Organized

The Administration Guide includes the following chapters:

- [About Oracle Financial Services Crime and Compliance Studio](#) provides a brief overview of the Crime and Compliance Studio application and its components.
- [Managing User Administration](#) provides a brief overview on creating users and mapping users with user groups.
- [Managing Studio Batches](#) provides information on creating and executing batches required for Studio.

Where to Find More Information

This section identifies additional documents related to Crime and Compliance Studio application. You can access the following documents from Oracle Help Center ([OHC](#)) Documentation Library:

- *Oracle Financial Services Crime and Compliance Studio User Guide*
- *Oracle Financial Services Crime and Compliance Studio Installation Guide*
- *Oracle Financial Services Crime and Compliance Studio ReadMe Guide*

Conventions Used in this Guide

The following table lists the conventions used in this guide and their associated meanings:

Table 0–1 Conventions Used in this Guide

Convention	Meaning
Boldface	Boldface type indicates graphical user interface elements associated with an action (menu names, field names, options, button names), or terms defined in text or glossary.
<i>Italic</i>	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
monospace	Monospace type indicates the following: <ul style="list-style-type: none">▪ Directories and subdirectories▪ File names and extensions▪ Process names▪ Code sample, that includes keywords, variables, and user-defined program elements within text
<variable>	Substitute input value

Abbreviations Used in this Guide

The following table lists the abbreviations used in this guide:

Table 0–2 Abbreviations and their meaning

Abbreviation	Meaning
OFS	Oracle Financial Services

Table 0–2 Abbreviations and their meaning

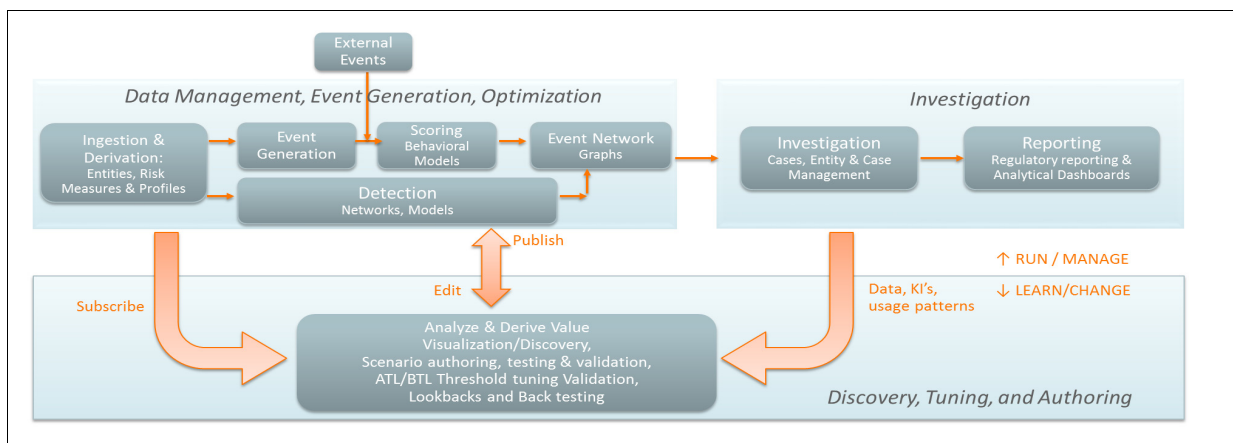
Abbreviation	Meaning
T2T	Table to Table
AAI	Analytical Applications Infrastructure
PGX	Parallel Graph AnalytiX
PGQL	Property Graph Query Language
LHS	Left Hand Side

About Oracle Financial Services Crime and Compliance Studio

Oracle's products and technology fight financial crime at financial institutions with the aid of the following methods:

- **Discovery, Tuning, and Authoring:** This represents a set of products and capabilities which are used to perform ad-hoc visual analysis and research to discover new and emerging patterns in financial crime. In addition to pattern discovery, these products also enable Data Scientists to publish or operationalize patterns discovered in ad-hoc analysis.

Figure 1-1 Crime and Compliance Studio in FCCM Architecture



This chapter provides complete functional details about Oracle Financial Services Crime and Compliance Studio application.

This chapter includes the following sections:

- [Introduction to Crime and Compliance Studio](#)
- [Salient Features of Crime and Compliance Studio](#)
- [Architecture of Crime and Compliance Studio](#)
- [PGX Graphs and Data Source Configuration](#)

Introduction to Crime and Compliance Studio

In order to effectively monitor anti-money laundering and anti-fraud programs in financial institutions, the most challenging need is to quickly identify and adapt to the changing patterns of financial crime. The ability to discover new and emerging criminal behavioral patterns, coupled with the facility to rapidly deploy as models, is a critical requirement.

Oracle Financial Services Crime and Compliance Studio is an integrated and comprehensive analytics toolkit designed to rapidly discover and model new financial crime patterns. Studio interacts with the database, processes the data, and generates patterns in various formats using interpreters. Studio provides secure access to an institution's financial crime data with pre-defined scenarios, out-of-the-box graph queries, and visualizations.

Data scientists and analysts can use Studio to interactively explore financial crime data and gain insights into new and emerging financial crime patterns and trends.

Studio uses Graph Analytics and Graph Query methods to analyze historic data available in the database, and forecast the generated patterns using various interpreters. Studio also uses Machine Learning Algorithms to gain insights from historical alert data, in order to prioritize the alerts generated by the detection engines.

Studio uses interpreters such as PGX, PGQL, GreenMarl, OFSAA Interpreter, OFSAA SQL Interpreter, and Markdown to generate patterns in various formats such as Table, Area Chart, Bar Chart, Funnel Chart, Line Chart, Pie Chart, Pyramid Chart, Tree Map, Sun Burst, Tag Cloud, Box Plot Chart, Scatter Plot Chart, and Raw Code.

Salient Features of Crime and Compliance Studio

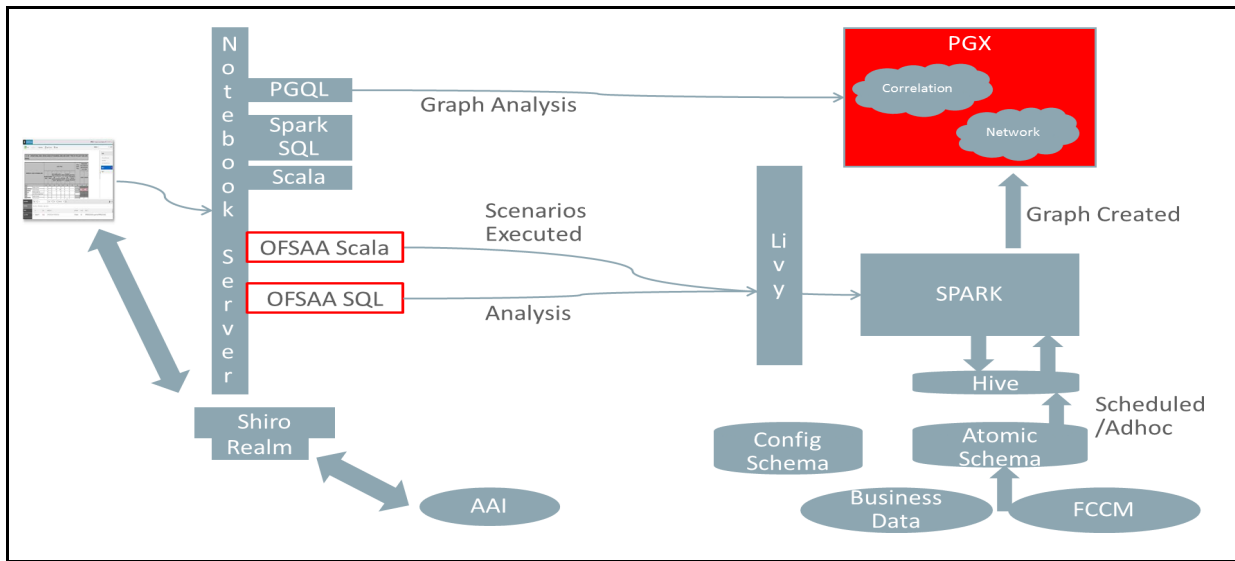
The salient features of Studio are as follows:

- Offers a unified tool for Graph Analytics, Data Visualization, Machine Learning, Scenario Authoring, Pattern Discovery, Data Mashups and testing for financial crime data
- Engineered to work with Apache Spark, the most prevalent analytics engine on Big Data
- Engineered to work with Apache Zeppelin, a web based notebook that enables interactive data analysis
- Supports Polyglot Scenario Authoring to author new scenarios in SQL, Scala, Python or R language
- Embedded with highly scalable in-memory Graph Analytics Engine (PGX)
- Enterprise ready with underlying OFSAA frameworks
- Engineered to work with earlier 8.x releases of Oracle Financial Crime and Compliance Management Anti Money Laundering (AML) and Fraud applications
- Integrated with Oracle Financial Crime Application Data and readily usable across the enterprise financial crime data lake. This can automatically load Oracle AML and Fraud data into the data lake and mash-up Studio data with third party data for discovery and modeling

Architecture of Crime and Compliance Studio

The following diagram depicts the architecture of Studio application:

Figure 1–2 Crime and Compliance Studio Architecture



PGX Graphs and Data Source Configuration

You can view the graphs that are created using Data Studio in the Crime and Compliance Studio interface. You can refer the data source to load graph in Data Studio. After loading the graph, you can execute PGQL query on the newly created graph.

You can also create customized graphs in Spark shell and configure data source for the newly created customized graphs in Studio. To create customized graphs, the data source must be configured manually. Information for the data source is available in the PGX log file located in the path where PGX is installed.

Managing User Administration

This chapter provides information on creating users who can access the Studio application, and executing batches that are required for Studio. Creation of users and execution of batches must be performed in the OFSAA environment.

User administration involves creating and managing users, and providing access to Crime and Compliance Studio based on assigned roles.

The following sections are covered in this chapter:

- [Managing Identity and Authorization](#)
- [Granting Permissions](#)

Managing Identity and Authorization

This section provides information on creating, mapping and authorizing users, and providing access to Studio application.

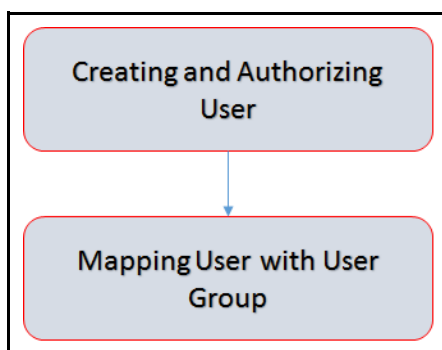
This section covers the following topics:

- [Identity and Authorization Process Flow](#)
- [Creating and Authorizing User](#)
- [Mapping User with User Group](#)

Identity and Authorization Process Flow

[Figure 2-1](#) shows the process flow of identity management and authorization.

Figure 2-1 *Managing Identity and Authorization Process Flow*



[Table 2-1](#) lists the various actions involved in the user administration process flow:

Table 2–1 User Administration Process Flow

Action	Description
Creating and Authorizing User	Create a user by providing the user name, user designation, and the date during which the user is active in Studio.
Mapping User with User Group	Map user with a user group that provides the user with the privileges of the mapped user group.

Creating and Authorizing User

The users with SYSADMN and SYSAUTH functional roles can respectively create and authorize users in the Studio. For more information on creating and authorizing users, see [Oracle Financial Services Analytical Applications Infrastructure User Guide](#).

Mapping User with User Group

This section provides information on mapping users with user groups. A user is mapped with a user group, and the user group is associated with a role. Each role comprises of certain predefined privileges.

Upon mapping a user to a user group, the user is granted with the privileges that are defined for the role of the user group. The SYSADM user maps a user to a user group in Studio.

[Table 2–2](#) describes the Roles and the corresponding User Groups in Studio.

Table 2–2 Roles and User Groups in Studio

Role	User Groups
DSADMIN	DSADMINGRP
DSINTER	DSINTERGRP
DSUSER	DSUSERGRP

Granting Permissions

1. Log in to Oracle Database from sys as a SYSDBA user.
2. Execute the following command:

```
grant execute dbms_ols to <Studio DB Username>
```

The Execute permission is granted to VPD.

3. Execute the following command:

```
grant create any context to <STUDIO_DB_USER_NAME>;
```

The Create permission is granted to context.

Managing Studio Batches

This chapter provides information on creating batches that are required for Crime and Compliance Studio. Batches are used for moving data, loading graph, and running notebook. This helps to move data from Oracle Database or Big data on daily basis in the Studio.

This chapter covers the following sections:

- [Preparing for Batches](#)
- [Performing Batches](#)
- [Creating and Executing Run Executable](#)

Preparing for Batches

Before you perform the batches, perform the following to prepare for batches:

1. Copy all the jars from `<FCC_STUDIO_INSTALLATION_PATH>/ficdb/lib` path into the `<FIC_HOME of OFSAA_Installed_Path>/ficdb/lib` path.
2. Copy the `NBExecutor.txt` file from the `<FCC_STUDIO_INSTALLATION_PATH>/ficdb/bin` path to the `<FIC_HOME of OFSAA_Installed_Path>/ficdb/bin` path.
3. Enter the following details in the `NBExecutor.txt` file:
username=<Studio Username>
password=<Studio Password>

Note: The username must be of a user who has permission to run Notebook.

Performing Batches

The following are the types of batches based on the applications:

- [Data Movement and Graph Loading for Oracle Database](#)
- [Data Movement and Graph Loading for Big Data Environment](#)
- [Executing Published Scenario Notebook](#)

Data Movement and Graph Loading for Oracle Database

This section covers the following topics:

- [Moving Data](#)
- [Loading Graph](#)

Moving Data

You can move data from *BD atomic* to *Studio schema* in batches as per the requirement. You can run this on daily basis to update daily data or after an interval to move data for any interval of time.

To create batches, follow these steps:

1. Copy the required 'FCCM_Studio_DB_DataMove.sh' files from <FCC_STUDIO_INSTALLATION_PATH>/ficdb/bin into <FIC_HOME of OFSAA_Installed_Path>/ficdb/bin path.
2. To create and execute run executable, see [Creating and Executing Run Executable](#) section.

Note: Alternatively it can be run from terminal using the following:

```
./FCCM_Studio_DB_DataMove.sh A B C 20151210 SNAPSHOT_  
DT=20180530,DATAMOVEMENTCODE=ALL
```

Where 'batchId' is 'A', 'ficmisdate' is '20151210' and 'userparams' are 'SNAPSHOT_DT' and 'DATAMOVEMENT' as '20180530' and 'ALL' respectively.

For more information on loading data in slice, see the Post Installation Configuration/ Oracle DB Data movement/Performing Data Movement in [Installation Guide](#).

Loading Graph

Graph load is used to create the graph from the underlying data. It gives the .pgb file and config.json of the OFSAAGLOBALGRAPH, which are further used in studio to view or query using PGQL and PGX interpreters.

To create batches, follow these steps

1. Copy the required 'FCCM_Studio_GraphLoad.sh' files from <FCC_STUDIO_INSTALLATION_PATH>/ficdb/bin into <FIC_HOME of OFSAA_Installed_Path>/ficdb/bin path.
2. To create and execute run executable, see [Creating and Executing Run Executable](#) section.

Note: Alternatively, you can run from the Terminal using the following:

```
./FCCM_Studio_GraphLoad.sh A B C 20151210 SNAPSHOT_DT=2018-05-30
```

Where 'batchId' is 'A', 'ficmisdate' is '20151210' and 'userparams' are 'SNAPSHOT_DT' as '20180530'.

Data Movement and Graph Loading for Big Data Environment

You can move data from *BD atomic* to *Big data* environment and then to create the .pgb file and config.json file of the graph. These files can be further used to view graph and query using PGQL and PGX.

To create batches, follow these steps

1. Copy the required 'FCCM_Studio_SqoopJob.sh' files from <FCC_STUDIO_INSTALLATION_PATH>/ficdb/bin into <FIC_HOME of OFSAA_Installed_Path>/ficdb/bin path.
2. To create and execute run executable, see [Creating and Executing Run Executable](#) section.

Note: Alternatively, you can run from the Terminal using the following:

```
./FCCM_Studio_SqoopJob.sh A B C 20150618 SNAPSHOT_
DT=20181219,DATAMOVEMENTCODE=ALL
```

Where 'batchId' is 'A', 'ficmisdate' is '20150618' and 'userparams' 'SNAPSHOT_DT' and 'DATAMOVEMENT' as '20181219' and 'ALL' respectively.

Executing Published Scenario Notebook

The published scenario notebook can be scheduled for execution with a set of threshold values as seemed required for generating alert or trends

To create batches, follow these steps

1. Copy the required 'FCCM_Studio_NotebookExecution.sh' file from <FCC_STUDIO_INSTALLATION_PATH>/ficdb/bin into <FIC_HOME of Installed OFSAA >/ficdb/bin path.
2. To create and execute run executable, see [Creating and Executing Run Executable](#) section.

Note: Alternatively, you can run from the Terminal using the following:

```
./FCCM_Studio_NotebookExecution.sh "notebookID" "null" "scenarioID"
"thresholdsetID" "null" "BATCH_ID"
```

If it's not a scenario notebook then you can execute the notebook as follows:

```
./FCCM_Studio_NotebookExecution.sh "notebookID" "null" "null"
>null" "paramkey1~~value1,paramkey2~~value2".
```

For example, if you have used ficmisdate as a paramkey1 and lookbackperiod as paramvalue2 with value1 as 20-09-2018 and value2 as 30 in your notebook then extraparams should be written as ficmisdate~~20-09-2018,lookbackperiod~~30

Creating and Executing Run Executable

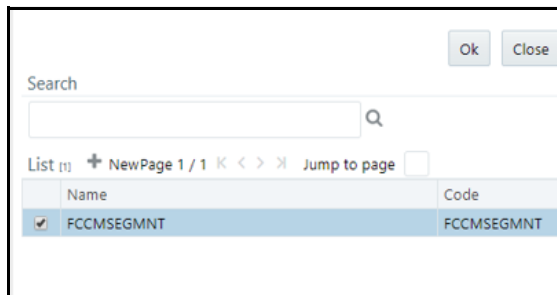
Perform the following to create run executable and execute it for Studio:

1. Log in to the OFSAA application with a user who has the privilege to create run executable.
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. From the Navigation List, navigate to **Common Tasks > Rule Run Framework > Run**.
The **Run Definition** page is displayed.
4. Click **New** on the List Tool bar.

The **Rule Run Framework** window is displayed.

- Under the **Linked To** tool bar, click the button next to **Folder**.

The **Folder Selector** dialog



- Select the folder that is to be linked to the run executable.
- Click **OK**.
- In the **Master Information** tool bar, enter the following details:

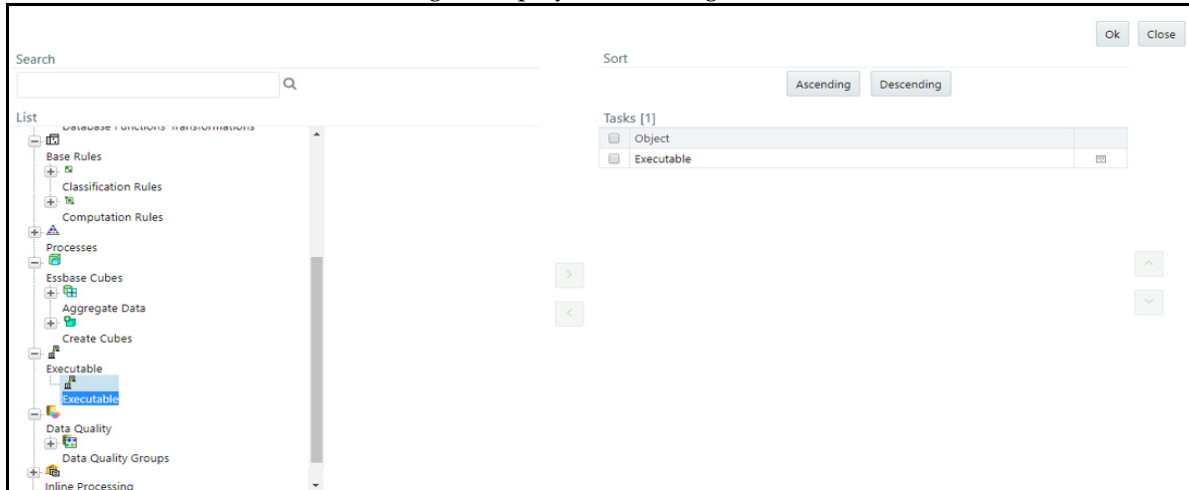


Table 3–1 Adding Run Definition

Fields	Description
Code	Enter the Code of the process.
Name	Enter the Name of the process.
Type	Select Type for the process.

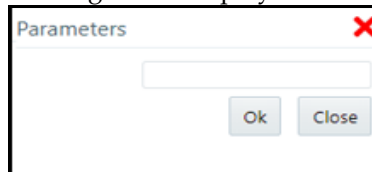
- Click **OK**.
- Click **Selector** on the List tool bar. From the options displayed, select **Job**.
The Job Details page is displayed.
- Click **Executable** on the list. From the options displayed, select **Executable**.

The Executable gets displayed on the right.



12. Select **Executable** from the Tasks list, click the button next to the Executable option.

The **Parameters** dialog box is displayed.



13. Enter the parameters in the following format to create run executable:

"FCCM_Studio_NotebookExecution.sh ", "notebookID", "outputParagraphID", "scenarioID", "thresholdsetID", "extraparams"

The following are required parameters.

Table 3–2 List of parameter for batch execution

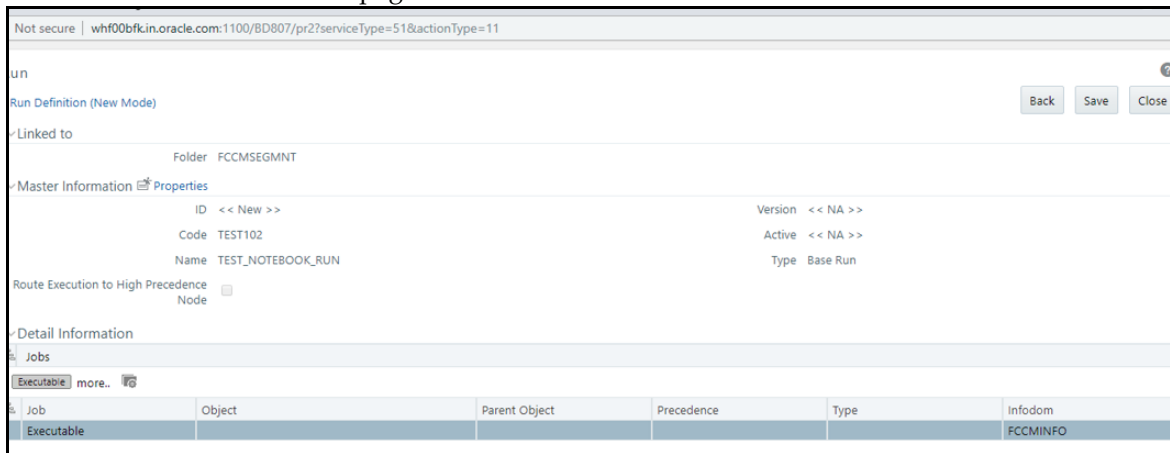
	File Name	Parameters	Description
1	FCCM_Studio_DB_DataMove.sh		Data movement for Oracle database
		"FCCM_Studio_DB_DataMove.sh", "batchId", "ficmisdate", "userparams"	
2	FCCM_Studio_GraphLoad.sh		Graph loading for Oracle database
		"FCCM_Studio_GraphLoad.sh", "batchId", "ficmisdate", "userparams"	
3	FCCM_Studio_SqoopJob.sh		Data movement and graph load in Big Data Environment
		"FCCM_Studio_SqoopJob.sh", "batchId", "ficmisdate", "userparams"	
4	FCCM_Studio_NotebookExecution.sh		For batch execution of Scenario notebook
		notebookId	ID of the required notebook
		outputParagraphId	It will always be "null".

Table 3–2 List of parameter for batch execution

File Name	Parameters	Description
	scenarioId	ID of Scenario
	thresholdsetId	ID of threshold set with which notebook will run
	sessionId	ID of session in which notebook will run
	extraParams	For scenario notebook, it will be "null", but for notebook execution, it depends on the paramkeys used in the notebook
"FCCM_Studio_NotebookExecution.sh", "notebookID", "outputParagraphID", "scenarioID", "thresholdsetId", "extraparams"		

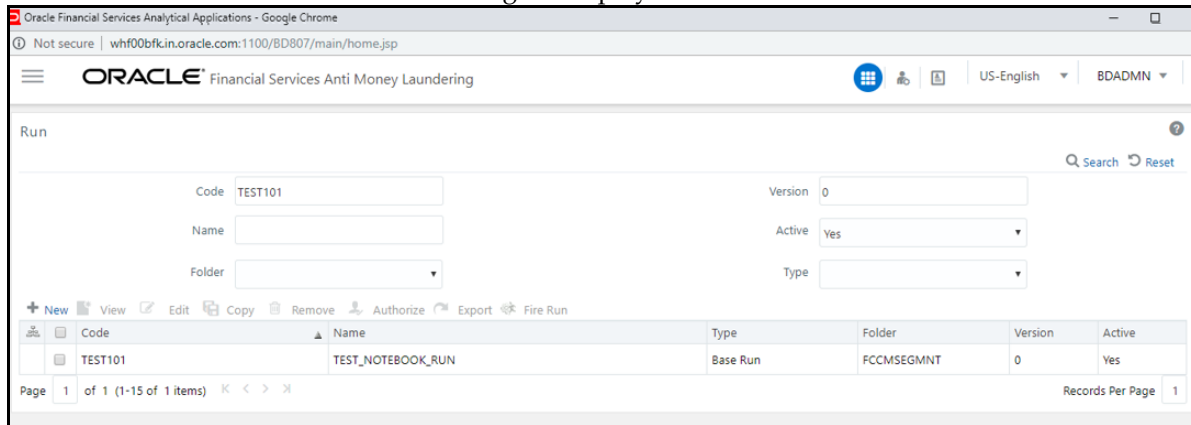
14. Click **OK**.

The run executable is displayed in the Detail Information section in the Run Definition page.



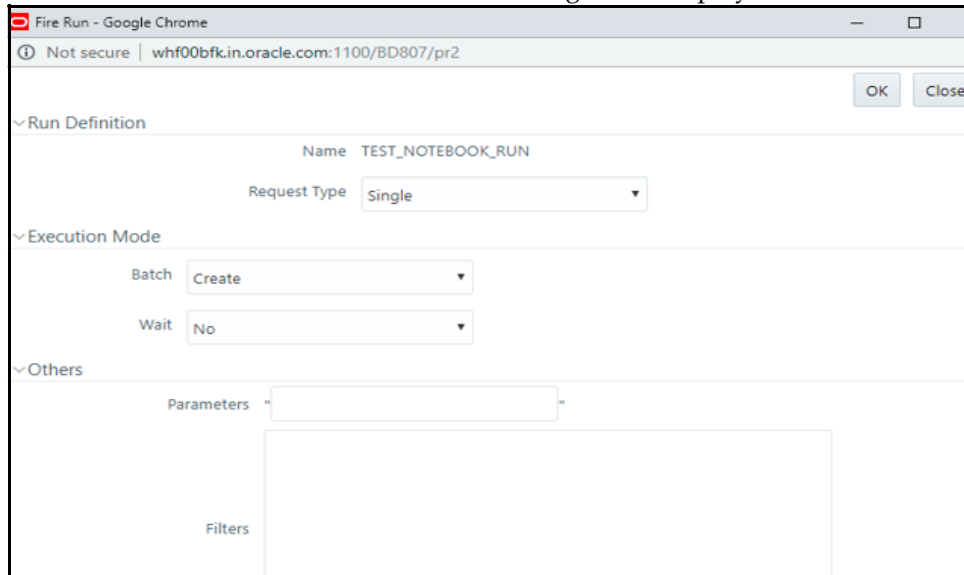
15. Click **Save**.

A confirmation message is displayed. The Run executable is created.



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.



17. Enter the following details:

Table 3-3 Adding Fire Run Details

Fields	Description
Request Type	Select Request Type based on the following options: <ul style="list-style-type: none"> • Single: If the batch has to be executed once. • Multiple: If the batch has to be executed multiple times at different intervals.
Batch	Select Batch. It has the following options: <ul style="list-style-type: none"> • Create • Create and Execute From these options, select Create & Execute.
Wait	Select Wait. It has the following options: <ul style="list-style-type: none"> • 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 ran in the step 18. Select the Information Date and Batch Run ID from the drop-down.

21. Click on **Start Monitoring** in Batch Run Details.

The Batch Run ID and Batch Status details are displayed in Batch Status details section.

Configuring Pre-Seeded Graph and Notebook

This chapter provides information on configuring pre-seeded graph in Data Studio.

Configuring Pre-seeded Graph in Studio

To configure pre-seeded graph Studio, perform below steps:


1. Navigate to <Studio_Installation_Path>/notebooks and graph/graphs path.
2. Modify the SYSTEM_OFSAAGLOBALGRAPH_2018_05_30_CONFIG.JSON file as follows:

Modify the following piece of code:

```
/scratch/ofsaweb/STUDIO0807_WBFK/OFS_FCCM_STUDIO/SYSTEM_
OFSAAGLOBALGRAPH_2018_05_30_PGB.PGB
```

to the following:

```
<Studio_Installation_Path>/notebooks and graph/graphs/SYSTEM_
OFSAAGLOBALGRAPH_2018_05_30_PGB.PGB
```

3. Copy the complete code from the modified SYSTEM_OFSAAGLOBALGRAPH_2018_05_30_CONFIG.JSON file.
4. Log in to the Crime and Compliance Studio application.
5. Navigate to the **Graphs** section and click **Create a Graph**.
The **Create new Graph Configuration** dialog box is displayed.
6. Enter the **Name** as **TESTGRAPH**, and select the value **PGB** from the **Format** drop-down field.
7. Click the plain icon  appearing on the left and paste the copied code from SYSTEM_OFSAAGLOBALGRAPH_2018_05_30_CONFIG.JSON to the **Json Configuration** field by replacing the existing content.
8. Click **Create**.

The pre-seeded graph is configured in Studio.

Configuring Pre-seeded Notebooks in Studio

To configure graphs and pre-seeded notebooks, in Data Studio, perform below steps:

1. Navigate to <Studio_Installation_Path>/notebooks and graph/notebooks path.
2. Copy the required notebooks to your local machine.

3. Log in to the Crime and Compliance Studio application.
4. Navigate to the **Notebooks** page.
5. Click the **Import Notebooks** icon on the top right corner.
The Import Notebook dialog box is displayed.
6. Choose the required files and click **Import**.
The notebooks are successfully imported and a success message is displayed to indicate the same.

Configuring Security for PGX

The PGX web server enables two-way SSL/TLS by default. The PGX server enforces TLS 1.2 and disables certain cipher suites known to be vulnerable to attacks. Upon TLS handshake, both server and client present certificates to each other which are used to validate the authenticity of the other party. Client certificates are additionally used to authorize client applications.

This chapter includes the following sections.

- [Prepare Certificates](#)
- [Prepare Client Keystore](#)
- [Prepare Client Truststore](#)
- [Configure PGX Web Server](#)
- [Test Connection Using PGX Client Shell](#)

Prepare Certificates

Note:

Disabling SSL/TLS

You can skip this part if you turn off SSL/TLS in single-node or multi-node PGX server configuration. However, we strongly recommend to leave SSL/TLS turned on for any production deployment.

You must create server certificate which will be validated by the client upon SSL/TLS handshake. You can either create a self-signed server certificate or import a certificate from a certificate authority.

This section includes the following:

- [Create Self-Signed Server Certificate](#)
- [Import Existing Certificate or Install Certificate from a Certificate Authority](#)

Create Self-Signed Server Certificate

Note: Do not use self-signed certificates in production deployments. For production, you should obtain a certificate from a certificate authority which is trusted by your organization.

You can create self-signed certificate to the keytool command-line utility, which is part of the Java Development Kit (JDK) that you already installed.

Perform the following to create self-signed server certificate:

- [Create New Keystore](#)
- [Extract the Certificate](#)

Create New Keystore

Perform the following to create a new keystore.

1. Create a new keystore containing a self-signed certificate by executing the following command:

```
keytool -genkey -alias pgx -keyalg RSA -keystore server_keystore.jks
```

The command prompts for keystore password, general information of the certificate (which will be displayed to clients who attempt to connect to the PGX web server) and the key password. The keystore password is for the keystore file itself and the key password is for the certificate. This is because JKS keystore files can store more than one certificate (identified by the provided alias).

2. Upon prompt, enter the first name, last name, and host name of the host you will deploy the PGX server on.

Note: If the host name in the certificate does not match the host name the server is listening on, client applications will reject the connection.

Note:

In distributed mode: use the host which starts the web server

For the multi-node setup, use the first host name in the list of `pgx_hostnames` as the host name for your certificates. Only the first host in this list will start a http server.

Extract the Certificate

The PGX server requires both the server certificate and server private key in the PKCS12 (PEM) format.

Perform the following to extract the certificate and private key from the JKS file:

1. Convert the generated `server_keystore.jks` file into a `server_keystore.p12` file by executing the following:

```
keytool -importkeystore -srckeystore server_keystore.jks -destkeystore
server_keystore.p12 -srcaias pgx \
    -srcstoretype jks -deststoretype pkcs12
```

The command will prompt with both the source and destination keystore password.

2. Enter the source and destination keystore password.

A file `server_keystore.p12` is generated in the current directory.

3. Extract certificate and private key from that `server_keystore.p12` file by executing the following openssl commands:

```
openssl pkcs12 -in server_keystore.p12 -nokeys -out server_cert.pem
openssl pkcs12 -in server_keystore.p12 -nodes -nocerts -out server_key.pem
```

The `server_cert.pem` and `server_key.pem` are generated in the current directory.

Import Existing Certificate or Install Certificate from a Certificate Authority

Refer [Tomcat TLS/SSL documentation](#) on how to import existing certificates or on how to install a certificate from a certificate authority into keystore files.

Prepare Client Keystore

Note:

Disabling two-way SSL/TLS

You can skip this part if you turn off client authentication in single-node or multi-node PGX server configuration. However, we strongly recommend to leave two-way SSL/TLS turned on for any production deployment.

For two-way SSL/TLS to work, you have to create one certificate for each client application you allow access to your PGX server. You must first create a keystore file for the client.

1. Execute the following to create a keystore file for the client:

```
keytool -genkey -alias pgx -keyalg RSA -keystore client_keystore.jks
```

The above command prompts with a keystore password, general information of the certificate and the key password. Note down the general information in the certificate (distinguished name string) as you will need this information in the next section for the PGX server authorization configuration.

2. You must sign the certificate inside the client keystore with the server private key which will make the client certificate to be accepted by the server. For which you must first create a sign request file `client.csr` by executing the following:

```
keytool -certreq -keystore client_keystore.jks -storepass <keystore_password> -alias pgx -keyalg RSA -file client.csr
```

3. Sign the `client.csr` file by providing both the server's certificate and private key files to the following openssl command:

```
openssl x509 -req -CA server_cert.pem -CAkey server_key.pem -in client.csr -out client_certificate.pem -days 365 -CAcreateserial
```

Above command generates a signed client certificate file `client_certificate.pem` which will be accepted by the server for the next 365 days. You can modify the `-days` parameter as per your needs.

4. Import both the server certificate as well as the signed client certificate back into the client keystore file by executing the following:

```
keytool -import -noprompt -trustcacerts -keystore client_keystore.jks -file server_cert.pem -alias pgxserver
```

```
keytool -import -noprompt -trustcacerts -keystore client_keystore.jks -file client_certificate.pem -alias pgx
```

Prepare Client Truststore

Use the same `client_keystore.jks` file for both the client keystore (which certificate to present to the server) and the client trust store (which server certificates to trust). If you have used a self-signed server certificate, you also have to import the server certificate's trust authority (CA) into the client keystore, else the client will reject the server certificate. Note that if you are using the PGX client shell, a range of well-known certificate authorities are trusted already by default by the client-side Java virtual machine.

Configure PGX Web Server

Specify the paths to the `server_cert.pem` and the `server_key.pem` files in the single-node or multi-node PGX server configurations. You can also specify a list of certificate authorities which will be trusted by the server.

Test Connection Using PGX Client Shell

If you started the web server with a self-signed certificate, you must first configure the client to accept the certificate. As the certificate is self-signed and not issued from a trusted certificate authority, the PGX client would reject it otherwise. You can set the trust store of the PGX client shell via the `--truststore` command-line option. Similarly, we have to specify the path to the keystore (`--keystore`) which contains the certificate the client will present to the server for authentication and authorization as well as the keystore password (`--password`).

Note: Do not accept self-signed certificates from unknown sources. Do not accept certificates from sources other than yourself.

Assuming the PGX web server listens on the default port 7007 on the same machine and you created the keystores as described above, you can test the connection by executing the following:

```
cd $PGX_HOME  
  
./bin/pgx --base_url https://localhost:7007 --truststore client_  
keystore.jks --keystore client_keystore.jks --password <keystore_password>
```

If the shell starts up without any error, you successfully connected to the PGX web server securely over two-way TLS/SSL.

Configuring Interpreters

An interpreter is a program that directly reads and executes the instructions written in a programming or scripting language without previously compiling the high level language code into machine language program.

The various interpreters Studio are PGX, PGQL, GreenMarl, OFSAA Interpreter, OFSAA SQL Interpreter, Markdown and so on.

This chapter includes the following topics:

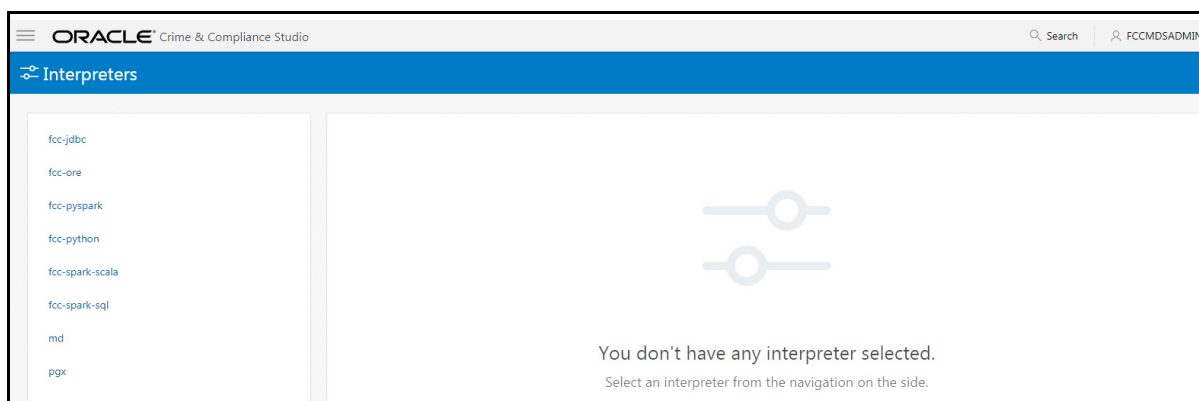
- [Accessing Interpreters](#)
- [Create New Interpreter Variant](#)
- [Configure Interpreters](#)

Accessing Interpreters

To access interpreters:

1. Click the menu icon in the upper-left corner in the **Home** page.
The menu items are listed.
2. Click **Interpreters**.
The **Interpreters** page is displayed.

Figure 6–1 Interpreters Page



3. Click the interpreter that you want to access from the list displayed on the LHS.
The default interpreter variant configured is displayed on the RHS.

4. Modify the required values.
5. Click **Update**.

The modified values are updated for the interpreter.

Create New Interpreter Variant

In Studio, you can either use a default interpreter variant or create new variant for an interpreter. You can create more than one variant for an interpreter.

To create a new variant for an interpreter:

1. Navigate to the **Interpreters** page.
2. Click the interpreter for which you want to create a new variant from the list displayed on the LHS.

The default interpreter variant is displayed on the RHS.

3. Click the following icon to create new variant for the selected interpreter:



The **Create New Variant** dialog box is displayed.

4. Enter the **Variant Name**.
5. Click **Create**.

A new variant is created with name, **<Interpreter Type>.<Variant Name>**.

6. Configure required values for the various properties.
7. Click **Update**.

A new variant is created for the interpreter.

Configure Interpreters

This section provides details of various interpreters in Studio and the configurations for each interpreter.

The various interpreters in Studio are as follows:

- [fcc-jdbc Interpreter](#)
- [fcc-ore Interpreter](#)
- [fcc-pyspark Interpreter](#)
- [fcc-python Interpreter](#)
- [fcc-spark-scala Interpreter](#)
- [fcc-spark-sql Interpreter](#)
- [md Interpreter](#)
- [pgx Interpreter](#)

fcc-jdbc Interpreter

The configuration for the ofsaa-jdbc is given as follows:

Table 6–1 fcc-jdbc interpreter Values

Field	Description
default.url	Enter the ofsaa jdbc URL in this field. For example: jdbc:mysql://localhost:5554/world
default.user	Enter the name of the default user in this field. For example: root
default.password	Enter the default password.
pgx.baseUri	Enter the pgx.baseUri URL in this field. This is the location where the data is pushed. For example: http://##HOSTNAME##:7007
ofsaa.metaservice.url	Enter the metaservice URL in this field. For example: http://##HOSTNAME##:7045/metaservice Here, ##HOSTNAME## refers to the server name or IP where fcc-studio will be installed.
ofsaa.sessionservice.url	Enter the session service URL in this field. For example: http://##HOSTNAME##:7047/sessionservice Here, ##HOSTNAME## refers to the server name or IP where fcc-studio will be installed.
default.completer.ttlInSeconds	Enter the time to live sql completer in seconds.
default.driver	Enter the default ofsaa JDBC driver name. For example: com.mysql.jdbc.Driver
default.completer.schemaFilters	Enter comma separated schema filters to get metadata for completions.
zeppelin.jdbc.precode	Enter the snippet of code that executes after initialization of the interpreter.
default.splitQueries	This field indicates the presence of default split queries. Enter "true" or "false".
common.max_count	Enter the maximum number of SQL result to display.
zeppelin.jdbc.auth.type	Enter the default jdbc authentication type.
zeppelin.jdbc.concurrent.use	Enter to enable or disable concurrent use of JDBC connections. Enter "true" or "false".
zeppelin.jdbc.concurrent.max_connection	Enter the number of maximum connections allowed.
zeppelin.jdbc.keytab.location	Enter the keytab location.
zeppelin.jdbc.principal	Enter the principal name to load from the keytab.

fcc-ore Interpreter

The configuration for the fcc-ore interpreter is given as follows:

Table 6–2 fcc-ore Interpreter Values

Field	Description
dummy.property.name	Enter any name
ore.host	Enter the host name of DB server where fcc-ore interpreter wants to connect.
ore.port	Enter the port number of DB server where fcc-ore interpreter wants to connect.
ore.sid	Enter the SID of DB server where fcc-ore interpreter wants to connect.
ore.user	Enter the schema name where fcc-ore interpreter wants to connect.
ore.password	Enter the schema password where fcc-ore interpreter wants to connect.
ore.service_name	Enter the Service Name of DB server where fcc-ore interpreter wants to connect.
ore.conn_string	Enter the DB connection URL with which fcc-ore interpreter can make the connection to the schema. This field can be left blank.
ore.type	Enter the fcc-ore interpreter type as Oracle.
ore.all	Indicates all tables are synced to fcc-ore interpreter. Enter the value as True.
ofsaaservice.url	Enter the metaservice URL in this field. For example: http://##HOSTNAME##:7045/metaservice Here, ##HOSTNAME## refers to the server name or IP where fcc-studio will be installed.
ofsaasession.url	Enter the session service URL in this field. For example: http://##HOSTNAME##:7047/session Here, ##HOSTNAME## refers to the server name or IP where fcc-studio will be installed.
http_proxy	Enter the Proxy server using which connection to internet is established. This value is used to set the initial setting that makes the environment compatible to download the libraries available in R. For example: www-proxy-hqdc.us.oracle.com:80
https_proxy	Enter the Proxy server using which connection to internet can be established. For example: www-proxy-hqdc.us.oracle.com:80
repo_cran	Indicates the CRAN URL from where R libraries are downloaded to install R packages. For example: https://cran.r-project.org/

Table 6–2 fcc-ore Interpreter Values

Field	Description
libpath	Indicates the custom library path from where R packages will be installed via Studio and will be added to R lib Path. Enter the path to be mentioned under home directory where studio is installed. For example: If you want the packages to be available under /home/user/library, and Studio is installed at /home/user/datstudio, then mention /library as the libpath.
rendering.row.limit	Indicates the number of rows to be shown in the fcc-ore interpreter output. For example: 1000
rendering.numeric.format	Indicates the Number of digits to round off. Example: %.2f
rendering.include.row.name	Indicates whether to include row names. Example: false
rserve.host	Indicates the Rserve host.
rserve.port	Indicates the Rserve port.
rserve.user	Indicates the Rserve username.
rserve.password	Indicates the Rserve password.
rserve.secure.login	Enter TRUE to enforce secure login.
rserve.plain.qap.disabled	Indicates whether plain QAP is disabled in the server or not. If disabled, the connection will always be attempted using SSL. For example: False
rserve.try.wrap	Enter False.
rendering.knitr.image.width	Indicates the image width specification for ore output. Example: 60
rendering.knitr.options	Enter Knitr output rendering option. Foe example: out.format = 'html', comment = NA, echo = FALSE, results = 'verbatim', message = F, warning = F, dpi = 300

fcc-pyspark Interpreter

The configuration for the fcc-pyspark interpreter is given as follows:

Table 6–3 fcc-pyspark Interpreter Values

Field	Description
pgx.baseUrl	Enter the pgx.baseUrl URL in this field. This is the location where the data is pushed. For example: http://##HOSTNAME##:7007
ofsa.metaservice.url	Enter the metaservice URL in this field. For example: http://##HOSTNAME##:7045/metaservice Here, ##HOSTNAME## refers to the server name or IP where fcc-studio will be installed.

Table 6–3 fcc-pyspark Interpreter Values

Field	Description
ofsa.session.service.url	Enter the session service URL in this field. For example: http://##HOSTNAME##:7047/session.service Here, ##HOSTNAME## refers to the server name or IP where fcc-studio will be installed.
zeppelin.livy.url	Enter the Livy URL in this field. Livy is an interface between Data Studio and Spark. For example: http://##HOSTNAME##:8998
zeppelin.livy.session.create_timeout	Enter the Zeppelin session creation timeout in seconds.
livy.spark.driver.cores	Enter the number of Number of driver cores to use for the driver process.
livy.spark.driver.memory	Enter the amount of memory to use for the driver process.
livy.spark.executor.instances	Enter the number of executors to launch for the current session.
livy.spark.executor.cores	Enter the number of Number of executor cores to use for the driver process.
livy.spark.executor.memory	Enter the amount of memory to use for the executor process.
livy.spark.dynamicAllocation.enabled	This field indicates whether Dynamic Allocation is enabled or not. Enter "true" or "false".
livy.spark.dynamicAllocation.cachedExecutorIdleTimeout	Enter the cached execution timeout in seconds.
livy.spark.dynamicAllocation.minExecutors	Enter the minimum number of required Dynamic Allocation executors.
livy.spark.dynamicAllocation.initialExecutors	Enter the initial Dynamic Allocation executors.
livy.spark.dynamicAllocation.maxExecutors	Enter the maximum number of required Dynamic Allocation executors.
zeppelin.livy.principal	Enter the principal name to load from the keytab.
zeppelin.livy.keytab	Enter the keytab location.
zeppelin.livy.pull_status.interval.millis	Enter the data pull interval in milliseconds.
livy.spark.jars.packages	Enter to add extra libraries to livy interpreter.
zeppelin.livy.displayApplicationInfo	This field indicates whether the application information needs to be displayed or not. Enter "true" or "false".
zeppelin.livy.spark.sql.maxResult	Enter the maximum number of results that needs to be fetched.

fcc-python Interpreter

The configuration for the python interpreter is given as follows:

Table 6–4 fcc-python Interpreter Values

Field	Description
zeppelin.python	Enter the Python installed path.

Table 6–4 fcc-python Interpreter Values

Field	Description
zeppelin.python.maxResult	Enter the maximum number of results that needs to be fetched.

fcc-spark-scala Interpreter

The configuration for the ofsaas interpreter is given as follows:

Table 6–5 fcc-spark-scala Interpreter Values

Field	Description
pgx.baseUrl	Enter the pgx.baseUrl URL in this field. This is the location where the data is pushed. For example: http://##HOSTNAME##:7007
ofsaas.metaservice.url	Enter the metaservice URL in this field. For example: http://##HOSTNAME##:7045/metaservice Here, ##HOSTNAME## refers to the server name or IP where fcc-studio will be installed.
ofsaas.sessionservice.url	Enter the session service URL in this field. For example: http://##HOSTNAME##:7047/sessionservice Here, ##HOSTNAME## refers to the server name or IP where fcc-studio will be installed.
zeppelin.livy.url	Enter the Livy URL in this field. Livy is an interface between Data Studio and Spark. For example: http://##HOSTNAME##:8998
zeppelin.livy.session.create_timeout	Enter the Zeppelin session creation timeout in seconds.
livy.spark.driver.cores	Enter the number of Number of driver cores to use for the driver process.
livy.spark.driver.memory	Enter the amount of memory to use for the driver process.
livy.spark.executor.instances	Enter the number of executors to launch for the current session.
livy.spark.executor.cores	Enter the number of Number of executor cores to use for the driver process.
livy.spark.executor.memory	Enter the amount of memory to use for the executor process.
livy.spark.dynamicAllocation.enabled	This field indicates whether Dynamic Allocation is enabled or not. Enter "true" or "false".
livy.spark.dynamicAllocation.cachedExecutorIdleTimeout	Enter the cached execution timeout in seconds.
livy.spark.dynamicAllocation.minExecutors	Enter the minimum number of required Dynamic Allocation executors.
livy.spark.dynamicAllocation.initialExecutors	Enter the initial Dynamic Allocation executors.
livy.spark.dynamicAllocation.maxExecutors	Enter the maximum number of required Dynamic Allocation executors.
zeppelin.livy.principal	Enter the principal name to load from the keytab.
zeppelin.livy.keytab	Enter the keytab location.

Table 6–5 fcc-spark-scala Interpreter Values

Field	Description
zeppelin.livy.pull_status.interval.millis	Enter the data pull interval in milliseconds.
livy.spark.jars.packages	Enter to add extra libraries to livy interpreter.
zeppelin.livy.displayAppInfo	This field indicates whether the application information needs to be displayed or not. Enter “true” or “false”.
zeppelin.livy.spark.sql.maxResult	Enter the maximum number of results that needs to be fetched.

fcc-spark-sql Interpreter

The configuration for the ofsa-sql interpreter is given as follows:

Table 6–6 fcc-spark-sql Interpreter Values

Field	Description
pgx.baseUrl	Enter the pgx.baseUrl URL in this field. This is the location where the data is pushed. For example: http://##HOSTNAME##:7007
ofsa.metaservice.url	Enter the metaservice URL in this field. For example: http://##HOSTNAME##:7045/metaservice Here, ##HOSTNAME## refers to the server name or IP where fcc-studio will be installed.
ofsa.sessionservice.url	Enter the session service URL in this field. For example: http://##HOSTNAME##:7047/sessionservice Here, ##HOSTNAME## refers to the server name or IP where fcc-studio will be installed.
zeppelin.livy.url	Enter the Livy URL in this field. Livy is an interface between Data Studio and Spark. For example: http://##HOSTNAME##:8998
zeppelin.livy.session.creation.timeout	Enter the Zeppelin session creation timeout in seconds.
livy.spark.driver.cores	Enter the number of Number of driver cores to use for the driver process.
livy.spark.driver.memory	Enter the amount of memory to use for the driver process.
livy.spark.executor.instances	Enter the number of executors to launch for the current session.
livy.spark.executor.cores	Enter the number of Number of executor cores to use for the driver process.
livy.spark.executor.memory	Enter the amount of memory to use for the executor process.
livy.spark.dynamicAllocation.enabled	This field indicates whether Dynamic Allocation is enabled or not. Enter “true” or “false”.
livy.spark.dynamicAllocation.cachedExecutorIdleTimeout	Enter the cached execution timeout in seconds.
livy.spark.dynamicAllocation.minExecutors	Enter the minimum number of required Dynamic Allocation executors.

Table 6–6 fcc-spark-sql Interpreter Values

Field	Description
livy.spark.dynamicAllocation.initialExecutors	Enter the initial Dynamic Allocation executors.
livy.spark.dynamicAllocation.maxExecutors	Enter the maximum number of required Dynamic Allocation executors.
zeppelin.livy.principal	Enter the principal name to lead from the keytab.
zeppelin.livy.keytab	Enter the keytab location.
zeppelin.livy.pull_status.interval.millis	Enter the data pull interval in milliseconds.
livy.spark.jars.packages	Enter to add extra libraries to livy interpreter.
zeppelin.livy.displayAppInfo	This field indicates whether the application information needs to be displayed or not. Enter “true” or “false”.
zeppelin.livy.spark.sql.maxResult	Enter the maximum number of results that needs to be fetched.

md Interpreter

The configuration for the md interpreter is given as follows:

Table 6–7 md Interpreter Values

Field	Description
markdown.parser.type	Enter the markdown parser type.

pgx Interpreter

The configuration for the pgx interpreter is given as follows:

Table 6–8 pgx Interpreter Values

Field	Description
pgx.timeout	Enter the pgx session creation timeout in seconds.
pgx.baseUrl	Enter the pgx.baseUrl URL in this field. This is the location where the data is pushed. For example: http://##HOSTNAME##:7007
pgx.trustStore	Enter the zeppelin.livy.url URL in this field. For example: http://##HOSTNAME##:8998
pgx.keyStore	Enter the Zeppelin session creation timeout in seconds.
pgx.password	Enter the pgx password.
pgx.accessToken	Enter the pgx access token.
pgx.prettyprint	Enter the pgx pretty print.
pgx.visualizePgqlResults	Enter the pgx visualize pgql results.
pgx.maxResults	Enter the maximum number of results that needs to be fetched.