Oracle® Financial Services Crime and Compliance Studio

Administration and Configuration Guide

Release 8.0.6.0.0

E91246-01

May 2018

Administration and Configuration Guide, Release 8.0.6.0.0

E91246-01

Copyright © 2018 Oracle and/or its affiliates. All rights reserved.

Primary Author: Nethravathi G

Contributor: Pankaj Chhangwani, Sukant Jain, Shwetha Yatham

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

6 Configuring Interpreters

List of Figures

List of Tables

Document Control

This section provides the revision details of the document.

Version Number	Revision Date	Changes Done
8.0.6.0.0	Created: May 2018	Created first version of Crime and Compliance Studio Administration Guide for 8.0.6.0.0 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*

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.
Italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates the following: ▪ 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
T2T	Table to Table

Table 0–2 Abbreviations and their meaning

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

1

About Oracle Financial Services Crime and Compliance Studio

This chapter provides a brief overview of the Oracle Financial Services Crime and Compliance Studio application in terms of its architecture and operations.

The following sections are covered in this chapter:

- Overview of Crime and Compliance Studio
- PGX Graphs and Data Source Configuration

Overview of Crime and Compliance Studio

The Crime and compliance Studio application is used to fight financial crime in financial institutions. Studio performs pattern discovery, data mashups, graph analytics, and data visualization. It can also be used to write and execute code, and view the result.

In Studio, the data is partitioned as scenarios and then moved into HIVE using the Sqoop component. The partitioned data in Hive is used for Parallel Graph Analytics (PGX) or data analysis. To ensure better performance, the data is moved on a daily basis.

Studio uses Parallel Graph Analytics (PGX) to discover new and suspicious patterns in data. Algorithms are employed to gain new insights from historical alert data in order to prioritize alerts generated by detection engines. Studio users are provisioned in the production instance.

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.

2

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

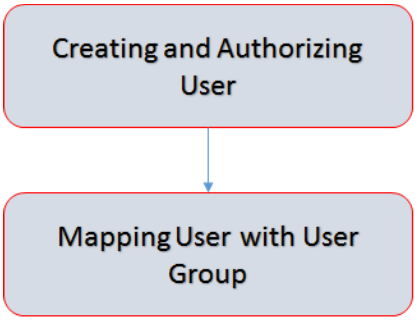


Table 2–1 lists the various actions involved in the user administration process flow:

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_rls 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.

3

Managing Studio Batches

This chapter provides information on creating batches that are required for Crime and Compliance Studio. Batches move data from ATOMIC schema to Hive schema on a daily basis in Studio. After completion of data movement, an OFSAA Globalgraph is created and loaded into PGX.

To create batches required for Studio, perform the following:

- Preparing for Studio Batch Creation
- Creating SSH Connection
- Creating Batch for Studio

Preparing for Studio Batch Creation

This section provides necessary information to review before creating batch required for Crime and Compliance Studio.

1. Copy the following files from the Studio installed path to the OFSAA installed path:
   - Copy all the jars from the `<Studio_Installed_Path>/ficdb/fccm_studio_DM_lib` path to the `<FIC_HOME of OFSAA_Installed_Path>/ficdb/lib` path.
   - Copy the `FCCM_DataStudio_LOAD.sh` file located in the `<Studio_Installed_Path>/ficdb/bin` path to the `<OFSAA_Installed_Path>/ficdb/bin` path.
2. Update the `<INFODOM>_TFM.xml` file located in the `ftpshare/<INFODOM>/erwin/fipxml/` by adding the following code:

```
<TASK>

<NAME>FCCM_DATASTUDIO_LOAD</NAME>

<SHORTDESCRIPTION>FCCM_DATASTUDIO_LOAD</SHORTDESCRIPTION>

<LONGDESCRIPTION>FCCM_DATASTUDIO_LOAD</LONGDESCRIPTION>

<TASKTYPE>DT</TASKTYPE>

<LASTMODIFIEDBY>AMADMIN</LASTMODIFIEDBY>

<LASTMODIFIEDDATE/>

<CREATORID>AMADMIN</CREATORID>

<CREATORDATE>9/18/2017</CREATORDATE>

<ISFLOWCHARTUPLOADED>N</ISFLOWCHARTUPLOADED>

<FLOWCHARTFILENAME/>
```

```
<STEPS>

<STEP>

<NAME>FCCM_DATASTUDIO_LOAD_1</NAME>

<SHORTDESCRIPTION/>

<LONGDESCRIPTION/>

<STEPTYPE>P</STEPTYPE>

<PRECEDENCESTEP/>

<PRECEDENCETYPE/>

<ACTION/>

<SRCFILE>./FCCM_DATASTUDIO_LOAD.sh</SRCFILE>

<SQLBLOCK/>

<JOINCONDITION/>

<SOURCEDATASET/>

<SOURCEATTRIBUTES/>

<DESTINATIONTABLE/>

<PARAMETERS/>

</STEP>

</STEPS>

</TASK>
```

3. Log in to the config schema and enter details in the following tables:

   - `date_task_master` table

     For sample data, see Table 3–1, Table 3–2, and Table 3–3

   - `date_task_step_precedence` table

     For sample data, see Table 3–4 and Table 3–5

Table 3–1 date_task_master

V_INFO_DOMAIN	V_TASK_ID	V_STEP_ID	V_STEP_STATUS
<FCCMINFO>	FCCM_DATASTUDIO_LOAD	FCCM_DATASTUDIO_LOAD_1	F

Table 3–2 (contd.) date_task_master

V_STEP_TYPE	V_EXT_PROC_FILE	V_STEP_PARAMETERS	V_PROCEDURE_NAM
P	./FCCM_DATASTUDIO_LOAD.sh	(RUNID,PHID,EXEID,RUNSK,INFODOM_NAME,TABLE_NAME)	FCCM_DATASTUDIO_DATAGRAPHLOAD

Table 3–3 (contd.) date_task_master

V_USED_COLUMNS	V_USED_TABLES	V_JOIN_CONDITION	V_TFM_TYPE
			DT

Table 3–4 date_task_step_precedence

V_INFO_DOMAIN	V_TASK_ID	V_STEP_ID
<FCCMINFO>	FCCM_DATASTUDIO_LOAD	FCCM_DATASTUDIO_LOAD_1

Table 3–5 (contd.) date_task_step_precedence

V_PARENT_STEP_ID	V_PARENT_STEP_STATUS	V_PARENT_STEP_PREC_BASIS
	S	S

Creating SSH Connection

To create SSH connection, perform below steps:

1. Go to Linux command line interface.
2. Navigate to the Home directory.
3. Execute the following command:

   ```
   ssh-keygen
   ```

   You will be prompted to press Enter key.
4. Press Enter for all the prompts.
5. Execute the following command:

   ```
   ssh-copy-id -i ~/.ssh/id_rsa.pub <Machine_UserName>@<remote-host>
   ```

   Here <remote-host> refers to the `SQOOP_TRG_HOSTNAME` parameter value in the `InstallConfig.xml` file. For more information, see Crime and Compliance Studio Installation Guide.

Creating Batch for Studio

Studio uses batch to move data from ATOMIC schema to Hive schema on a daily basis.

To create batches required for Studio, perform below steps:

1. Log in to the OFSAA application as `ecmadmn` user.
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 **Operations > Batch Maintenance**.

   The **Batch Maintenance** window is displayed.
4. Click add icon in the **Batch Name** section.

   The **Batch Maintenance** window is displayed.

5. Enter the **Batch Name** and **Batch Description**.
6. Click **Save**.

   A new batch is created.
7. Select the newly created batch in the **Batch Name** section and click add icon in the **Task Details** section.

   The **Task Definition** window is displayed.
8. Enter the **Task ID** and **Description** in the **Task Definition** window.
9. Select **Transform Data** in the **Components** drop-down field in the **Task Definition** window.

   Based on the values entered in the **Task Definition** section, the **Dynamic Parameters List** section displays the corresponding properties and values.
10. Select **FCCM_DATASTUDIO_LOAD** in the **Rule Name** drop-down field in the **Dynamic Parameters List** section.
11. Enter required date in the **Parameter List** field.
12. Click **Save**.

    The task is defined for the newly created batch.

4

Configuring Pre-Seeded Graph

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

Configuring Pre-seeded Graph in Studio

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

1. Move the `SYSTEM_OFSAAGLOBALGRAPH_SAMPLE_PGB.PGB` file located in the `<Studio_INSTALLED_PATH>/datastudio` path to the `hdfs` path.
2. Modify the `SYSTEM_OFSAAGLOBALGRAPH_SAMPLE_CONFIG.JSON` file located in the `<Studio_INSTALLED_PATH>/datastudio` path as follows:

   Modify the following piece of code:

   ```
   hdfs:/user/ofsaa/SYSTEM_OFSAAGLOBALGRAPH_2015-12-12_PGB.PGB
   ```

   to the following:

   ```
   hdfs:<Path where PGB file is placed in hdfs>/SYSTEM_OFSAAGLOBALGRAPH_
   SAMPLE_PGB.PGB
   ```

3. Copy the modified code from Step 2.
4. Log in to the Crime and Compliance Studio application.
5. Navigate to the **GRAPHS** section and click **Create Graph Configuration**.

   The **Create new Graph Configuration** dialog box is displayed.

6. Enter the **Name** as **FCCGlobalGraph**, and select the value **PGB** from the **Format** drop-down field.
7. Click the plain icon appearing on the left and paste the modified code copied from Step 2 to the **Json Configuration** field by replacing the existing content.
8. Click **Create**.

   The pre-seeded graph is configured in Data Studio.

5 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 -srcalias 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.

7

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:

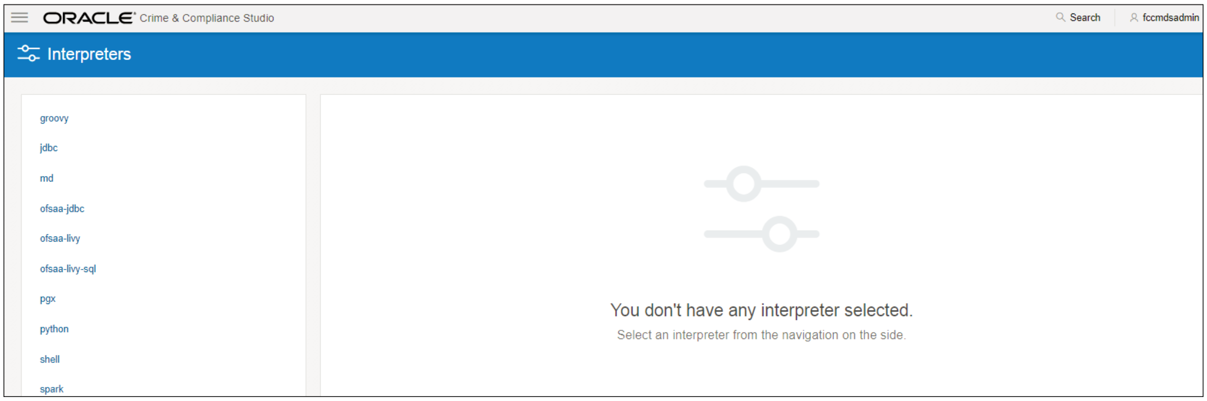
1. Click the menu icon in the upper-left corner in the **Home** page.

   The menu items are listed.

2. Click **Interpreters**.

   The **Interpreters** page is displayed.

Figure 6–1 Interpreters Page



You don't have any interpreter selected.

Select an interpreter from the navigation on the side.

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:

- md Interpreter
- ofsaa-jdbc Interpreter
- ofsaa Interpreter
- ofsaa-sql Interpreter
- pgx Interpreter
- python Interpreter

md Interpreter

The configuration for the md interpreter is given as follows:

Table 6–1 md Interpreter Values

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

ofsaa-jdbc Interpreter

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

Table 6–2 ofsaa-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. Foe example: root
default.password	Enter the default password.
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.

ofsaa Interpreter

The configuration for the ofsaa interpreter is given as follows:

Table 6–3 ofsaa 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://whf00awx.in.oracle.com:7007**
ofsaa.service.url	Enter the OFSAA metadata server URL in this field. For example: **http://whf00awx.in.oracle.com:6080/metaservice**

Table 6–3 ofsaa Interpreter Values

Field	Description
zeppelin.livy.url	Enter the Livy URL in this field. Livy is an interface between Data Studio ans Spark. For example: **http://whf00awx.in.oracle.com: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.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.

ofsaa-sql Interpreter

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

Table 6–4 ofsaa-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://whf00awx.in.oracle.com:7007**
ofsaa.service.url	Enter the OFSAA metadata server URL in this field. For example: **http://whf00awx.in.oracle.com:6080/metaservice**

Table 6–4 ofsaa-sql Interpreter Values

Field	Description
zeppelin.livy.url	Enter the Livy URL in this field. Livy is an interface between Data Studio ans Spark. For example: **http://whf00awx.in.oracle.com: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 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.

pgx Interpreter

The configuration for the pgx interpreter is given as follows:

Table 6–5 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. For example: **http://localhost:7007**
pgx.trustStore	Enter the zeppelin.livy.url URL in this field. For example: **http://whf00awx.in.oracle.com:8998**

Table 6–5 pgx Interpreter Values

Field	Description
pgx.keyStore	Enter the Zeppelin session creation timeout in seconds.
pgx.password	Enter the pgx password.
pgx.accessToken	
pgx.prettyprint	
pgx.visualizePgqlResults	
pgx.maxResults	Enter the maximum number of results that needs to be fetched.

python Interpreter

The configuration for the python interpreter is given as follows:

Table 6–6 python Interpreter Values

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