Oracle® Financial Services Crime and Compliance Management Studio

User Guide

Release 8.0.5.0.0

E91355-01

November 2017

User Guide, Release 8.0.5.0.0

E91355-01

Primary Author: Aneesh Kurian

Contributing Author:

Contents

List of Figures

List of Tables

Document Control

This section provides the revision details of the document.

Version Number	Revision Date	Changes Done
1.0	Created: November 2017	Created first version of OFS FCCM Studio User Guide.

This document provides you functional information about the Oracle Financial Services Crime and Compliance Management Studio (OFS CCMS) application 8.0.5.0.0 and enables you to navigate through the various sections of the application. The latest copy of this guide can be accessed from OHC Documentation Library.

Preface

This section provides supporting information for the Oracle Financial Services Crime and Compliance Management Studio (OFS CCMS) application 8.0.5.0.0 User Guide and includes the following topics:

- Summary
- Audience
- Related Documents
- Conventions
- Abbreviations

Summary

You can find the latest copy of this document in OHC Library which includes all the recent additions/revisions (if any) done till date.

Audience

Oracle Financial Services Crime and Compliance Management Studio User Guide is intended for end users such as Data Analysts and Data Scientists.

Related Documents

This section identifies additional documents related to OFS CCMS.

OFSAAI Related Documents

Following documents are available in OHC Documentation Library.

- *Oracle Financial Services Analytical Applications Infrastructure User Guide*

OFS CCM Studio Application Related Documents

Following documents are available in OHC.

- *Oracle Financial Services Crime and Compliance Management Studio Installation Guide*
- *Oracle Financial Services Crime and Compliance Management Studio Admin Guide*
- *Oracle Financial Services Crime and Compliance Management Studio Release Notes*

Conventions

The following text conventions are used in this document:

Table 0–1 Conventions used in this guide

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

Abbreviations

The following table lists the abbreviations used in this document:

Table 0–2 Abbreviations and their meaning

Abbreviation	Meaning
OFS CCMS	Oracle Financial Services Crime and Compliance Management Studio
FCCM	Financial Crime and Compliance Management
OFSAA	Oracle Financial Services Analytical Applications
SQL	Structured Query Language

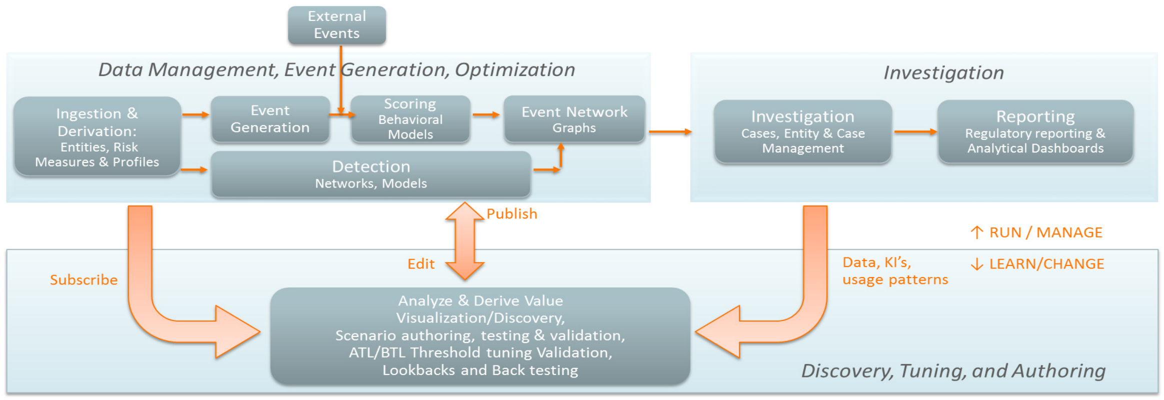
1

Oracle Financial Services Crime and Compliance Management 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 OFS CCMS in FCCM Architecture


External Events
Data Management, Event Generation, Optimization
Ingestion & Derivation: Entities, Risk Measures & Profiles
Event Generation
Scoring Behavioral Models
Event Network Graphs
Detection Networks, Models
Investigation
Investigation Cases, Entity & Case Management
Reporting Regulatory reporting & Analytical Dashboards
Publish
Edit
Subscribe
↑ RUN / MANAGE
↓ LEARN/CHANGE
Data, KI's, usage patterns
Analyze & Derive Value Visualization/Discovery, Scenario authoring, testing & validation, ATL/BTL Threshold tuning Validation, Lookbacks and Back testing
Discovery, Tuning, and Authoring

This chapter provides complete functional details about Oracle Financial Services Crime and Compliance Management Studio (OFS CCMS) application.

This chapter includes the following topics:

- Introduction to OFS CCMS
- Architecture of OFS CCMS
- Components of OFS CCMS

Introduction to OFS CCMS

OFS CCMS offers an intuitive Graph Analytics and Graph Query interface, which enables Data Scientists and Data Analysts in using historic data to forecast the trends, through various interpreters. OFS CCMS also utilizes Machine Learning Algorithms to gain new insights from historical alert data, in order to prioritize alerts generated by detection engines.

The UI of OFS CCMS interacts with the database, processes the data, and the results are generated and displayed 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 even as Raw Code.

Interpreters facilitate the output generation. There are various interpreters which process the data and generates the output. The OFS OFS CCMS uses the interpreters such as PGX, PGQL, GreenMarl, OFSAA Interpreter, OFSAA SQL Interpreter, and Markdown.

Salient Features of OFS CCMS

Following are the prominent features of OFS CCMS application:

- Big Data Analytics platform (Spark and Graph (PGX))
- Enterprise ready with underlying OFSAA frameworks
- In-memory model execution for rapid, agile development with polyglot language support
- Baseline scenarios included to support quick ramp-up for users
- A platform for new scenario development
- Graph analytics, Data mash-ups, Pattern discovery, and Data visualization
- State-of-the-art Graph engine for network and fraud analytics

OFS CCMS provides a single, unified workbench for Graph Analytics and visualization, Scenario Authoring and testing, Machine Learning, and Data Visualization for Financial Crimes Data. This comes with an embedded high performance graph analytics engine (PGX).

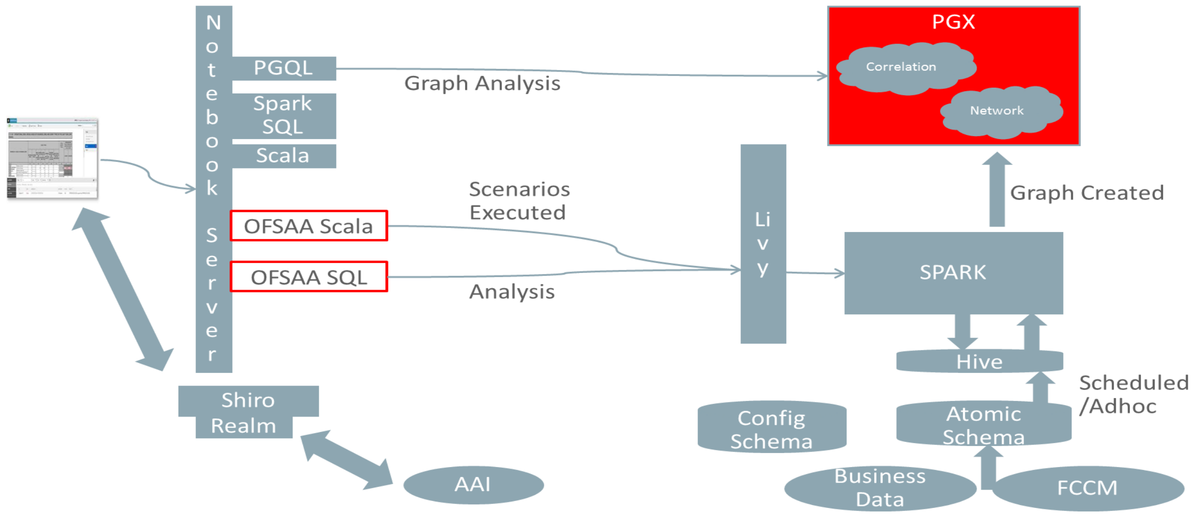
OFS CCMS is seamlessly integrated with Oracle AML and Fraud but engineered as a portal into the enterprise Financial Crimes data lake. This can automatically load Oracle AML and Fraud data into the data lake and 'mash-up' Oracle FCCM data with third party data for discovery and modeling.

OFS CCMS is also fully compliant with open source standards for “Notebooks” (no need to retrain data scientists) and is capable of running any Notebook created for Apache Zeppelin.

Architecture of OFS CCMS

The following diagram depicts the architecture of OFS CCMS application:

Figure 1–2 OFS CCMS Architecture



Components of OFS CCMS

This section provides you an insight to various components of OFS CCMS, which are discussed in various Chapters of this User Guide.

- OFS CCMS Home
- My Notebooks
- Shared Notebooks
- Graphs
- Templates
- Interpreters

2

OFS CCMS Home

This chapter provides complete details about the OFS CCMS Home. This module displays all the existing Notebooks mapped to your role. The details such as Name, Detailed Information, and Tags are displayed for each Notebook.

This chapter includes the following topics:

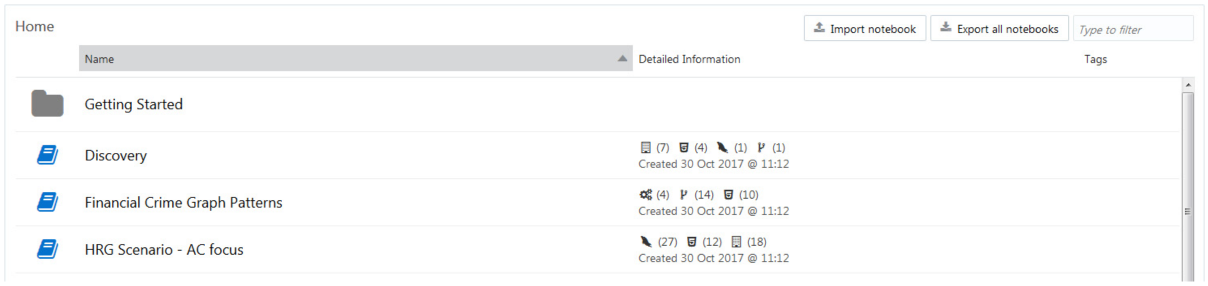
- Accessing FCCM Home
- Components of OFS CCMS Home
- Creating a Notebook
- Creating a Paragraph

Accessing FCCM Home

When you login to the OFS CCMS application, you are directed to the OFS CCMS Home.

The OFS CCMS Home is displayed as depicted in the following figure:

Figure 2–1 OFS CCMS Home

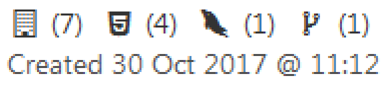


Name	Detailed Information	Tags
Getting Started		
Discovery	(7) (4) (1) (1) Created 30 Oct 2017 @ 11:12	
Financial Crime Graph Patterns	(4) (14) (10) Created 30 Oct 2017 @ 11:12	
HRG Scenario - AC focus	(27) (12) (18) Created 30 Oct 2017 @ 11:12	

Components of OFS CCMS Home

The OFS CCMS Home displays all the existing notebooks along with the details such as created date with time and number of compilations using various interpreters. These details are displayed as depicted in the following sample figure:

Figure 2–2 Notebook Details



The OFS CCMS Home also displays the Tags associated with each Notebooks.

Another prominent set of feature that the OFS CCMS Home provide are the options to import new Notebooks and export existing Notebooks.

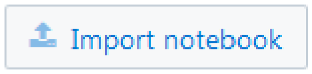
Import Notebooks

The Import Notebooks feature enables you to load Data Studio Notebook (*.dsnb) files from your local machine. Using this feature, you can load any previously saved or exported Notebooks.

To import a Notebook from the OFS CCMS Home, perform the following procedure:

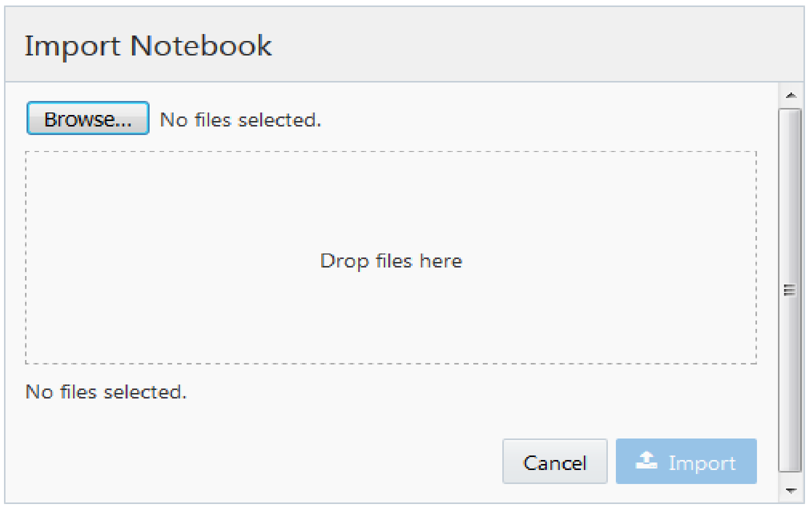
1. Click the following button, from the OFS CCMS Home:

Figure 2–3 Import Notebook - Import Button



The *Import Notebook* window is displayed:

Figure 2–4 Import Notebook



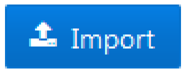
2. Click the **Browse** button to locate the file in your local machine.

   Alternatively, you can drag and drop the files in the area marked as "Drop files here".

   Once the file is selected, it is validated and a message is displayed.

3. Click the following button:

Figure 2–5 Import Button



The file is imported and the Notebook is displayed in OFS CCMS Home.

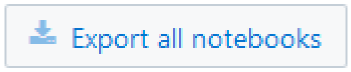
Export Notebooks

This feature enables you to export all the Notebooks or individual Notebooks to your local machine. Note that, if you want to export individual Notebooks, you need to open the Notebook that you want to export and perform export operation.

Perform the following procedure to export all the Notebooks at once, from the:

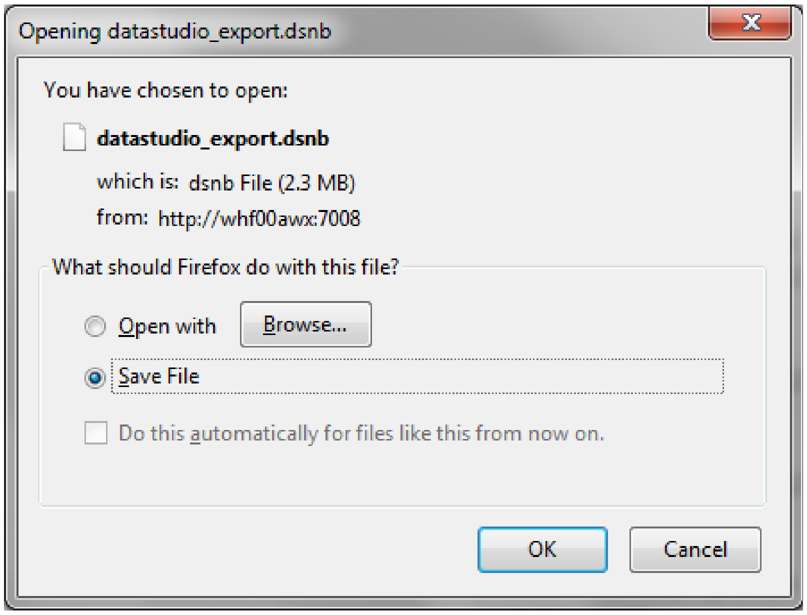
1. Click the following button, from the OFS CCMS Home:

Figure 2–6 Export Notebook - Export Button



All the Notebooks in the OFS CCMS Home are loaded to a file namely **datastudio_ export.dsnb** and displayed for save operation, as depicted in the following figure:

Figure 2–7 Export Notebook - Save Prompt



Opening datastudio_export.dsnb

You have chosen to open:

datastudio_export.dsnb

which is: dsnb File (2.3 MB)

from: http://whf00awx:7008

What should Firefox do with this file?

2. Click **OK** button and select a location to save the file.
3. Click **Save** button.

You can also rename the file before saving.

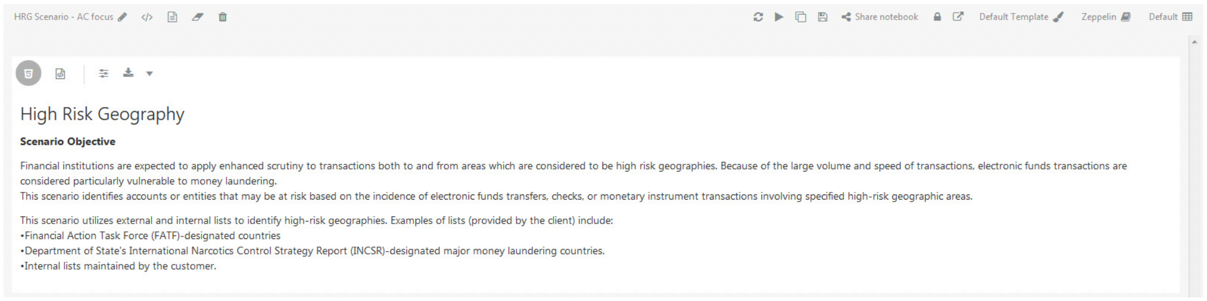
Export Individual Notebooks

To export individual Notebooks, perform the following procedure:

1. Click the Notebook you want to export.

   The Notebook is opened as depicted in the following sample figure:

Figure 2–8 Notebook Display



High Risk Geography

Scenario Objective

Financial institutions are expected to apply enhanced scrutiny to transactions both to and from areas which are considered to be high risk geographies. Because of the large volume and speed of transactions, electronic funds transactions are considered particularly vulnerable to money laundering.
This scenario identifies accounts or entities that may be at risk based on the incidence of electronic funds transfers, checks, or monetary instrument transactions involving specified high-risk geographic areas.

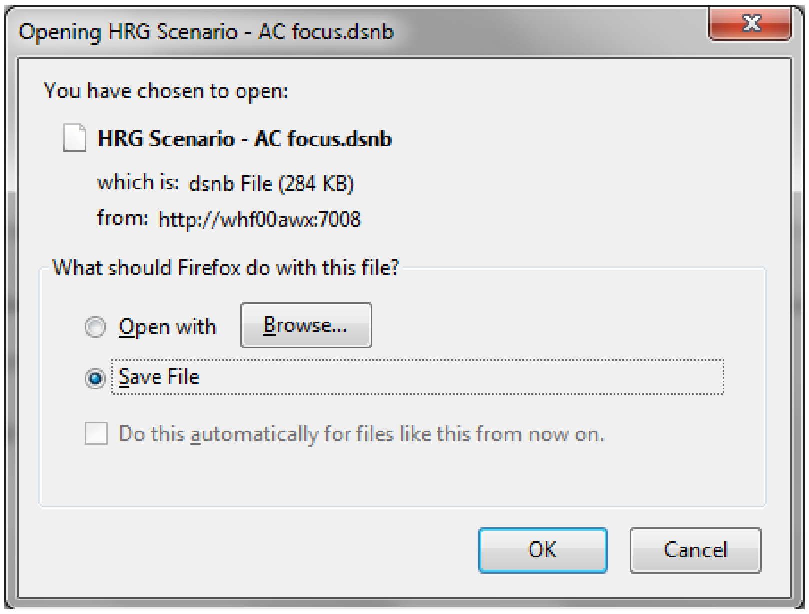
This scenario utilizes external and internal lists to identify high-risk geographies. Examples of lists (provided by the client) include:
•Financial Action Task Force (FATF)-designated countries
•Department of State's International Narcotics Control Strategy Report (INCSR)-designated major money laundering countries.
•Internal lists maintained by the customer.

2. Click the following button:

Figure 2–9 Export Individual Notebook - Export Button

The Notebook is loaded to a file with extension **.dsnb** and displayed for save operation, as depicted in the following sample figure:

Figure 2–10 Export Individual Notebook - Save Prompt


Opening HRG Scenario - AC focus.dsnb

You have chosen to open:

HRG Scenario - AC focus.dsnb

which is: dsnb File (284 KB)

from: http://whf00awx:7008

What should Firefox do with this file?

○ Open with [Browse...]

● Save File

☐ Do this automatically for files like this from now on.

[OK] [Cancel]

3. Click **OK** button and select a location to save the file.
4. Click **Save** button.

   You can also rename the file before saving.

Filter Contents on My Notebooks Module

This feature enables you to filter the Notebooks in the OFS CCMS Home by name. You can enter the keywords in **Type to Filter** field. As you key in the keywords, the list is filtered and is updated.

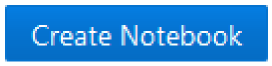
Creating a Notebook

This section describes the process of creating a new Notebook. Notebooks creation is the initial process and this process creates a frame for the Paragraphs which is the basic building block.

You can create a new Notebook from any module of the OFS CCMS application. The Notebook creation process can be initiated by performing the following procedure:

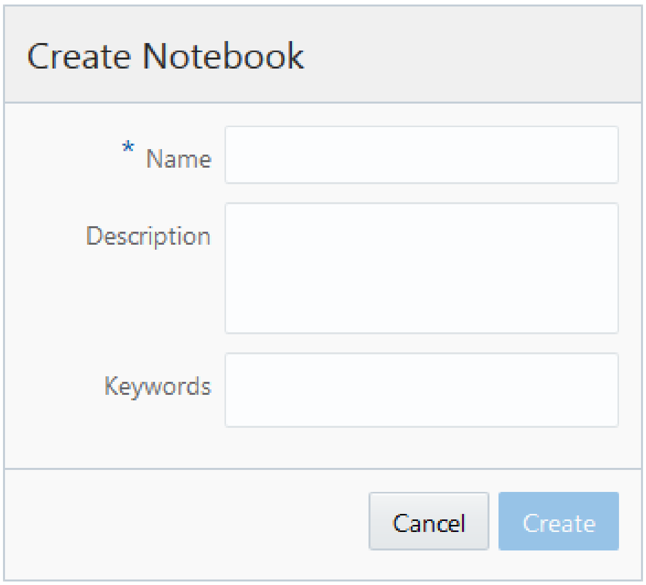
1. Click the following button, present in the application task bar:

Figure 2–11 Create Notebook Button



2. Upon clicking this button, the Create Notebook window is displayed, as depicted in the following figure:

Figure 2–12 Create Notebook



3. Populate this form as tabulated:

Table 2–1 Create Notebook

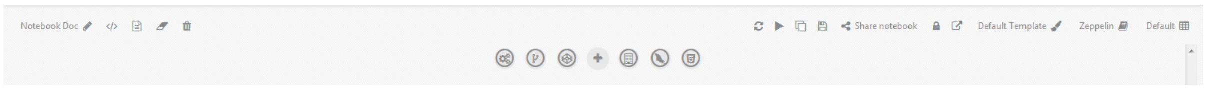
Field	**Description**
Fields marked in asterisks (*) are mandatory.	
Name	Enter a name for the Notebook in this field. The character range of Notebook name is between three and 32. Also, the Name should not start or end with a slash (/).
Description	Enter a description about the Notebook.
Keywords	Enter the keywords through which this Notebook can be searched. These keyword act as search tags.

4. Click the following button:

Figure 2–13 Create Button

The Notebook is created and is displayed as depicted in the following sample figure:

Figure 2–14 Notebook



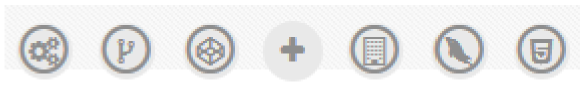
To know more about the actions that can be performed from a Notebook screen, see Manage Notebooks section.

Creating a Paragraph

A Notebook can contain multiple Paragraphs. Paragraphs enable you to enter the required code/query to obtain a result. The content in a Paragraph needs to be interpreter friendly. To ensure this, the Paragraph creation is performed by clicking the suitable interpretor icon. For more details on interpreters, see Interpreters section.

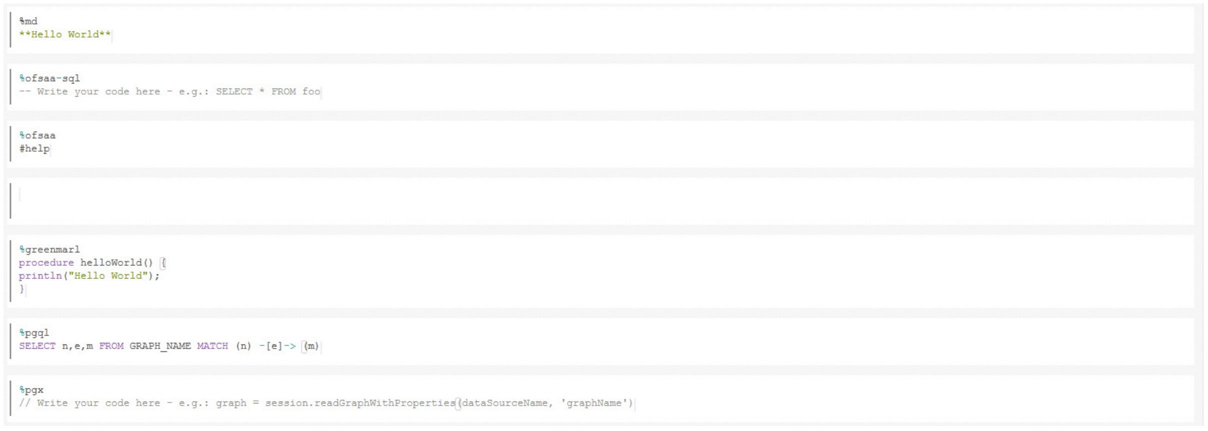
Perform the following procedure to create a blank paragraph:

1. Click one of the following icons in a Notebook:



   Depends on the type of interpreter that you have selected, one of the following Paragraphs is displayed:

Figure 2–15 Paragraph



```
%md
**Hello World**
```

```
%ofsaa-sql
-- Write your code here - e.g.: SELECT * FROM foo
```

```
%ofsaa
#help
```

```
%greenmarl
procedure helloWorld() {
println("Hello World");
}
```

```
%pgql
SELECT n,e,m FROM GRAPH_NAME MATCH (n) -[e]-> (m)
```

```
%pgx
// Write your code here - e.g.: graph = session.readGraphWithProperties(dataSourceName, 'graphName')
```

2. Enter the required code/query in the paragraph.

To know more about the actions that can be performed from a Paragraph screen, see Manage Paragraphs section.

3 My Notebooks

This chapter provides complete details about shared notebooks and Out of the Box Notebooks in OFS CCMS application.

This chapter includes the following topics:

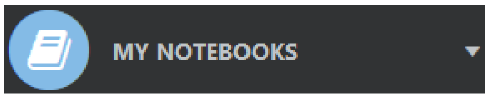
- Introduction
- Access My Notebooks

Introduction

The My Notebooks module displays all the Notebooks created by the currently logged in user.

Access My Notebooks

You can access the My Notebooks module by clicking the following icon and then selecting the required Notebook from the list.



Once you select a Notebook, the selected Notebook is open in the RHS. From here, you can view or edit the Notebook and the underlying Paragraphs.

4 Shared Notebooks

This chapter provides complete details about shared notebooks and Out of the Box Notebooks in OFS CCMS application.

This chapter includes the following topics:

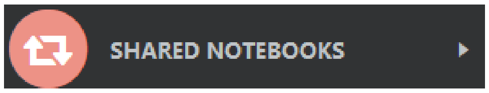
- Introduction
- Access Shared Notebooks
- Out of the Box Notebooks

Introduction

The Shared Notebooks module displays all the out of the box Notebooks and the Notebooks shared with the current user. Notebooks can be shared within the current User Group and other User Groups.

Access Shared Notebooks

You can access the Shared Notebooks module by clicking the following icon.



Upon clicking this icon, the underlying Notebooks are displayed. You can click the required Notebook to view or edit it.

Out of the Box Notebooks

The OFS CCMS Application comes with the following default Notebooks:

Table 4–1 Out of the Box Notebooks

Notebook Name	Description
Discovery	This Notebook lists all the available APIs/Functions available for a user to work with in Studio.
Financial Crime Graph Patterns	This Notebook describes a few patterns of fraud that can be visualized and discovered using Graph/PGX capabilities.
HRG Scenario - AC focus	The Notebook contains the High Risk Geography scenario logic defined in Scala and Spark SQL. This also shows the capabilities to visualize the data at every data point.

Table 4–1 Out of the Box Notebooks

Notebook Name	Description
RMF - All Activity	The Notebook contains the Rapid Movements of Fund scenario logic defined in Scala and Spark SQL. This also shows the capabilities to visualize the data at every data point.
RMF Scenario - CU Focus	The Notebook contains the Rapid Movements Of Fund-Customer Focus, scenario logic defined in Scala and Spark SQL. This also shows the capabilities to visualize the data at every data point.

5 Graphs

This chapter provides complete details about Graphs in OFS CCMS application. You can view the graphs that are created out of Data Studio from the OFS CCMS interface.

If you are creating custom Graphs, the Data Store needs to be configured manually. To know more about configuring Graphs, see *Oracle Financial Services CCM Studio Admin Guide* in OHC Documentation Library.

6 Templates

Templates allow you to specify the same visualization options that you can do directly on a widget, in the CCM Studio application, after pressing the Settings button.

When you create a Notebook with a defined Template and update this template, the changes will propagate to every widget of every Notebooks uses it.

This chapter includes the following topics:

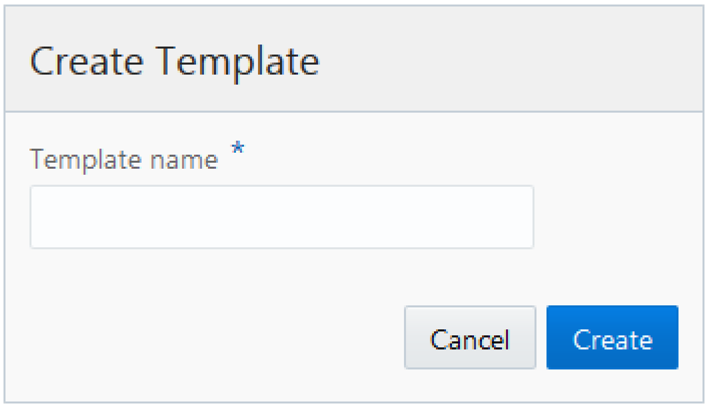
- Create a Template
- Update an Existing Template

Create a Template

To create a new Template, perform the following procedure:

1. Click the Create Template button present in the LHS menu of the Templates section in CCM Studio application.

   The *Create Template* window is displayed:



2. Enter a name for the new Template in the **Template Name** field and click **Create** button.

   The RHS is displayed with all the available settings. These include Treemap, Table, HTML, Raw, Sunburst, Tag Cloud, Exploration, Visualization, Box Plot Chart, Scatter Plot Chart, Area Chart, Bar Chart, Funnel Chart, Line Chart, Pie Chart, and Pyramid Chart.

   For every option, you can set values for General, Visualization, and Text parameters.

3. Click **Update** button to save the changes.

Update an Existing Template

To update the values of an existing Template, perform the following procedure:

1. Click the Template which you want to update in the LHS of the Templates section.

   The details of the Template are displayed in the RHS.

2. Update the required values.
3. Click **Update** button to save the changes.

7 Interpreters

This chapter provides complete details about OFS CCMS Interpreters such as PGX, PGQL, GreenMarl, OFSAA Interpreter, OFSAA SQL Interpreter, and Markdown.

This chapter includes the following topics:

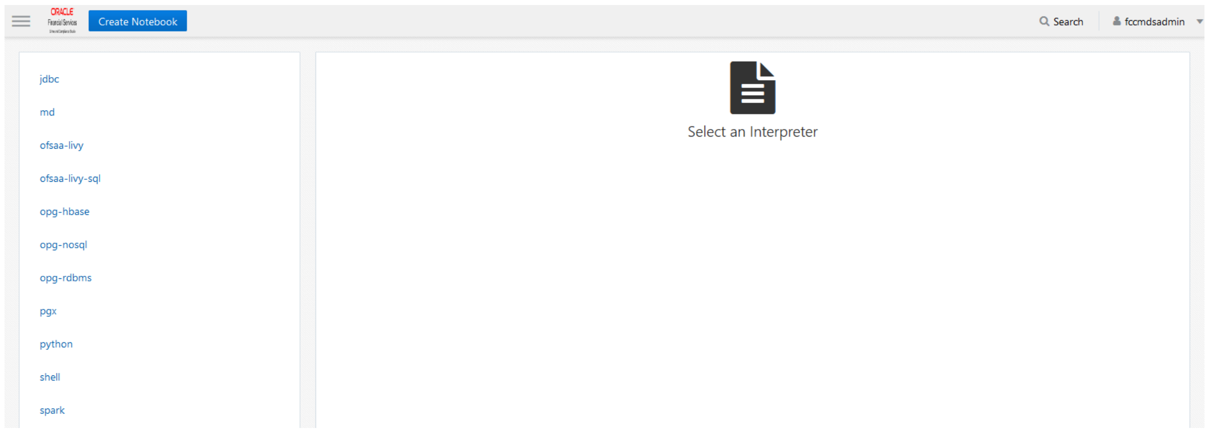
- Accessing Interpreters
- Managing Interpreters

Accessing Interpreters

You can access the Interpreters module by clicking the following icon.

Upon clicking this icon, you are directed to the *Interpreters* page. This page displays all the available Interpreters as depicted in the following sample figure:

Figure 7–1 Interpreters



The Interpreters page displays following available Interpreters in the application:

- jdbc Interpreter
- md Interpreter
- ofsaa-livy Interpreter

- ofsaa-livy-sql Interpreter
- pgx Interpreter
- python Interpreter
- shell Interpreter
- spark Interpreter

You can update the present variable values of these Interpreters and can create new variables from Interpreters screen. For more information about these actions, see Managing Interpreters section.

Managing Interpreters

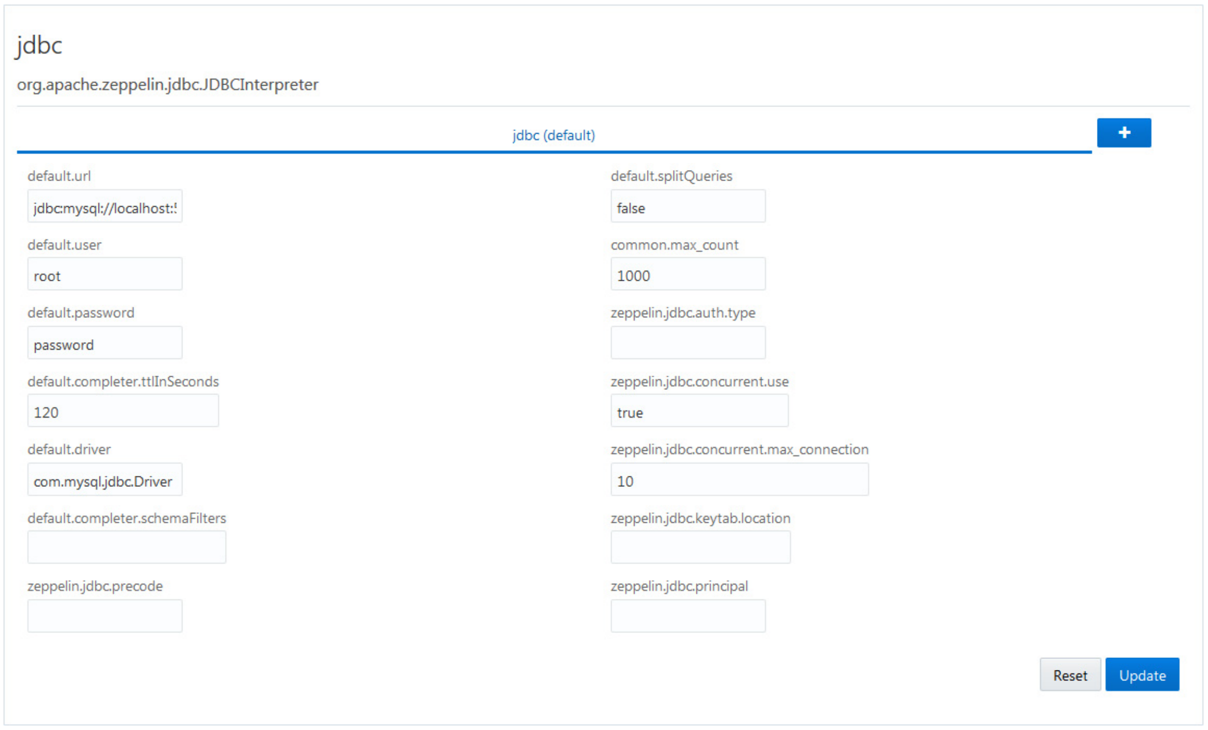
This section enables you to handle various Interpreters. Processes to manage each Interpreter type are detailed in the following sections.

jdbc Interpreter

Managing Default jdbc Interpreter

You can access the jdbc interpreter by clicking **jdbc** link in the LHS menu of *Interpreters* page. Upon clicking this link, the RHS is displayed with default jdbc variant and its corresponding values, as depicted in the following figure:

Figure 7–2 jdbc Interpreter



jdbc

org.apache.zeppelin.jdbc.JDBCInterpreter

default.url: jdbc:mysql://localhost:!	default.splitQueries: false
default.user: root	common.max_count: 1000
default.password: password	zeppelin.jdbc.auth.type:
default.completer.ttlInSeconds: 120	zeppelin.jdbc.concurrent.use: true
default.driver: com.mysql.jdbc.Driver	zeppelin.jdbc.concurrent.max_connection: 10
default.completer.schemaFilters:	zeppelin.jdbc.keytab.location:
zeppelin.jdbc.precode:	zeppelin.jdbc.principal:

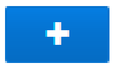
Table 7–1 jdbc Interpreter Values

Field	Description
The permissible character range for these fields is between three and 255.	
default.url	Enter the jdbc URL in this field. For example: **jdbc:mysql://localhost:5554/world**
default.splitQueries	This field indicates the presence of default split queries. Enter “true” or “false”.
default.user	Enter the name of the default user in this field. Foe example: **root**
common.max_count	
default.password	Enter the default password.
zeppelin.jdbc.auth.type	Enter the default jdbc authentication type.
default.completer.ttlInSeconds	
zeppelin.jdbc.concurrent.use	
default.driver	Enter the default driver. For example: **com.mysql.jdbc.Driver**
zeppelin.jdbc.concurrent.max_connection	Enter the number of maximum connections allowed.
default.completer.schemaFilters	
zeppelin.jdbc.keytab.location	
zeppelin.jdbc.keytab.location	
zeppelin.jdbc.precode	
zeppelin.jdbc.principal	

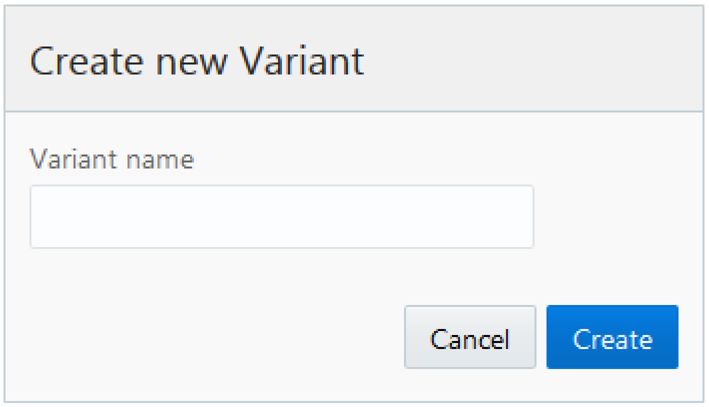
Creating a New Variant

You can create multiple variants for jdbc Interpreter. To create a new variant, perform the following procedure:

1. Click the following button, from the *jdbc Interpreter* page:



The *Create New Variant* popup window is displayed:



2. Enter the name of the variant in the **Variant name** field.
3. Click the following button:

   The new variant is created and is displayed in a tab adjacent to the default variable. The new variant is named **jdbc <variant name>**.

   You can add/update the details of the new variant by filling the fields. For more information, see Table 7–1, "jdbc Interpreter Values"

You can also perform the following actions from the *jdbc Interpreter* page:

- Click **Reset** button to reset the values in all the field to default values.
- Update the required fields and click following button to save the changes:

- Click **X** button adjacent to the variant tab to remove a variant.

Note: You cannot remove the default variant.

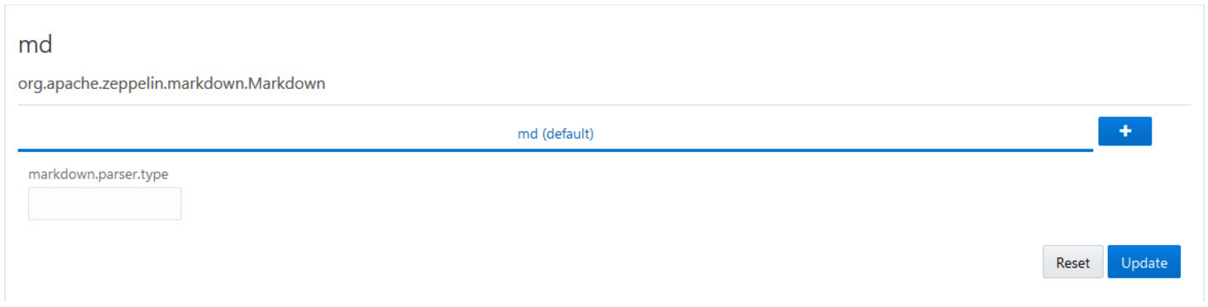
---

md Interpreter

Managing Default md Interpreter

You can access the md interpreter by clicking **md** link in the LHS menu of *Interpreters* page. Upon clicking this link, the RHS is displayed with default md variant and its corresponding values, as depicted in the following figure:

Figure 7–3 md Interpreter



md

org.apache.zeppelin.markdown.Markdown

Enter the value for

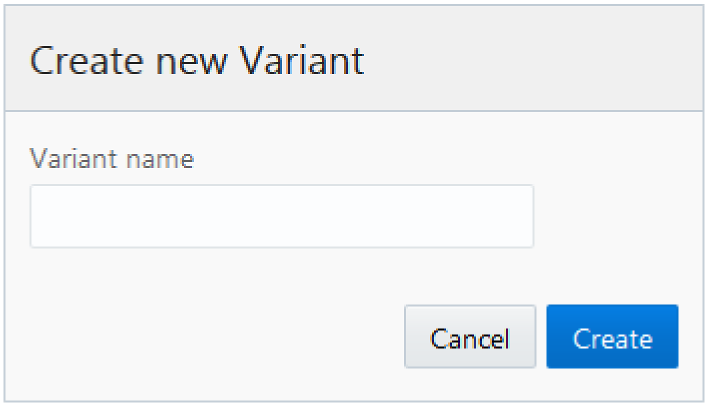
markdown.parser.type in the field.

Creating a New Variant

You can create multiple variants for md Interpreter. To create a new variant, perform the following procedure:

1. Click the following button, from the *md Interpreter* page:

The *Create New Variant* popup window is displayed:



Create new Variant

2. Enter the name of the variant in the **Variant name** field.
3. Click the following button:

The new variant is created and is displayed in a tab adjacent to the default variable. The new variant is named **md <variant name>**.

You can add/update the details of the new variant by filling the necessary value in **markdown.parser.type** field.

You can also perform the following actions from the *md Interpreter* page:

- Click **Reset** button to reset the values in all the field to default values.
- Update the required fields and click following button to save the changes:

- Click **X** button adjacent to the variant tab to remove a variant.

Note: You cannot remove the default variant.

ofsaa-livy Interpreter

ofsaa-livy Interpreter uses built-in functions and Scala functions. Data loaded through these are processed in Spark.

ofsaa-livy Interpreter provides the following functions:

- **loadGraph(code,graphtag [--optional,date])** - to load graph for a particular date in yyyy-mm-dd format.

  Return type is void and should not be assigned to any variable.

  For example: loadGraph("code","graphtag","date");
- **saveGraphAs(graphtag, usertag)** - to create a snapshot of graph against a new tab.

  Return type is void and should not be assigned to any variable.

  For example: saveGraphAs("graphtag","usertag");
- **listGraphs()** - to list all the available graphs in a tabular form.

  Cannot be used in conjunction with any other code.

  For example: listGraphs()
- **getGraphDetails(code)** - to get the graph details in a tabular form.

  Cannot be used in conjunction with any other code.

  For example: getGraphDetails("code")
- **loadDataset(code)** - to load a particular dataset.

  Returns a DataFrame.

  For example: var ds = loadDataset("code"))
- **loadDataset(code,date)** - to load dataset for a particular date in yyyy-mm-dd format.

  Returns a DataFrame.

  For example: var ds = loadDataset("code","date"))
- **loadDataset(code,startDate,endDate)** - to load dataset for a specific time period with date in yyyy-mm-dd format.

  Returns a DataFrame.

  For example: var ds = loadDataset("code","startDate","endDate"))
- **listDatasets()** - to list all the available datasets in a tabular form.

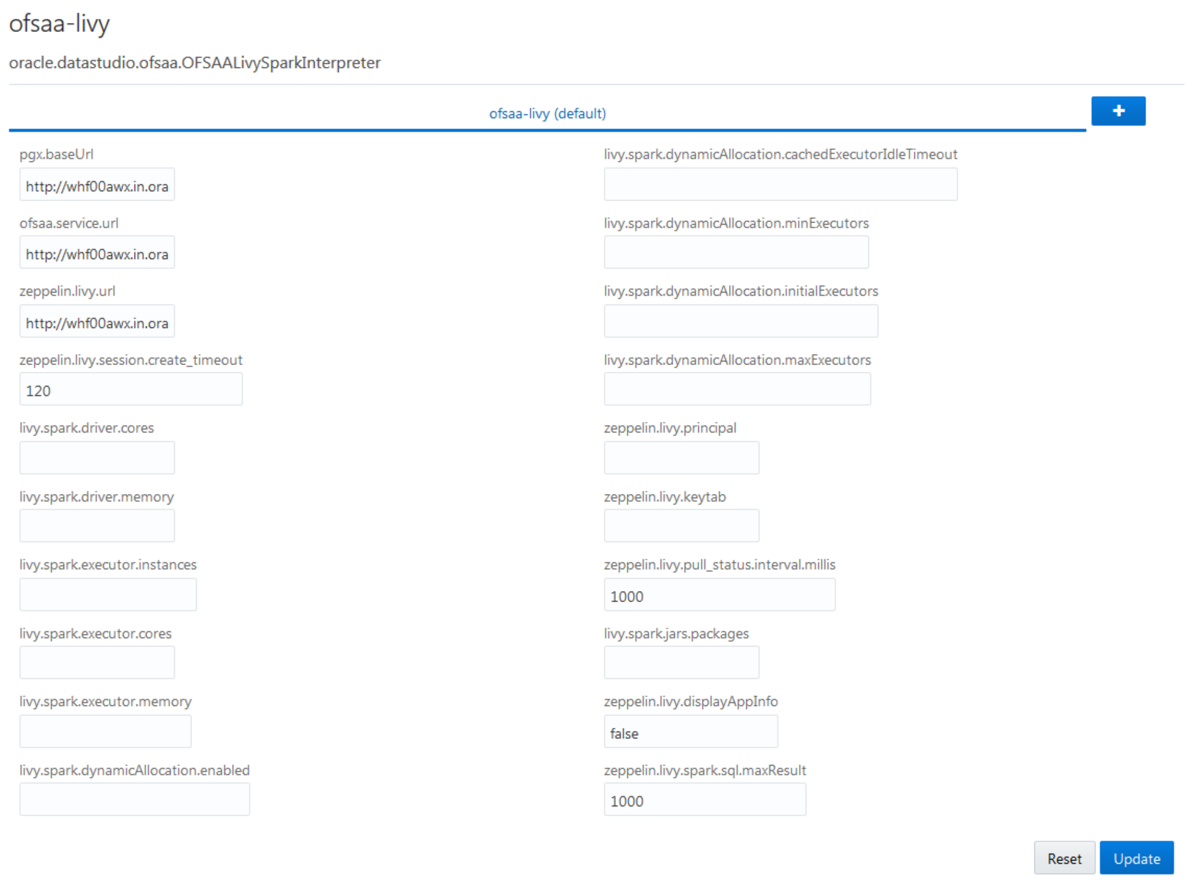
Cannot be used in conjunction with any other code.

For example: listDatasets()

Managing Default ofsaa-livy Interpreter

You can access the ofsaa-livy interpreter by clicking **ofsaa-livy** link in the LHS menu of *Interpreters* page. Upon clicking this link, the RHS is displayed with default ofsaa-livy variant and its corresponding values, as depicted in the following figure:

Figure 7–4 ofsaa-livy Interpreter


ofsaa-livy

oracle.datastudio.ofsaa.OFSAALivySparkInterpreter

pgx.baseUrl	http://whf00awx.in.ora
ofsaa.service.url	http://whf00awx.in.ora
zeppelin.livy.url	http://whf00awx.in.ora
zeppelin.livy.session.create_timeout	120
livy.spark.driver.cores	
livy.spark.driver.memory	
livy.spark.executor.instances	
livy.spark.executor.cores	
livy.spark.executor.memory	
livy.spark.dynamicAllocation.enabled	
livy.spark.dynamicAllocation.cachedExecutorIdleTimeout	
livy.spark.dynamicAllocation.minExecutors	
livy.spark.dynamicAllocation.initialExecutors	
livy.spark.dynamicAllocation.maxExecutors	
zeppelin.livy.principal	
zeppelin.livy.keytab	
zeppelin.livy.pull_status.interval.millis	1000
livy.spark.jars.packages	
zeppelin.livy.displayAppInfo	false
zeppelin.livy.spark.sql.maxResult	1000

Table 7–2 ofsaa-livy Interpreter Values

Field	Description
The permissible character range for these fields is between three and 255.	
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**
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**

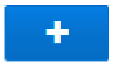
Table 7–2 ofsaa-livy Interpreter Values

Field	Description
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.
zeppelin.jdbc.concurrent.use	
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	
zeppelin.livy.keytab	
zeppelin.livy.pull_status.interval.millis	Enter the data pull interval in milliseconds.
livy.spark.jars.packages	
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.

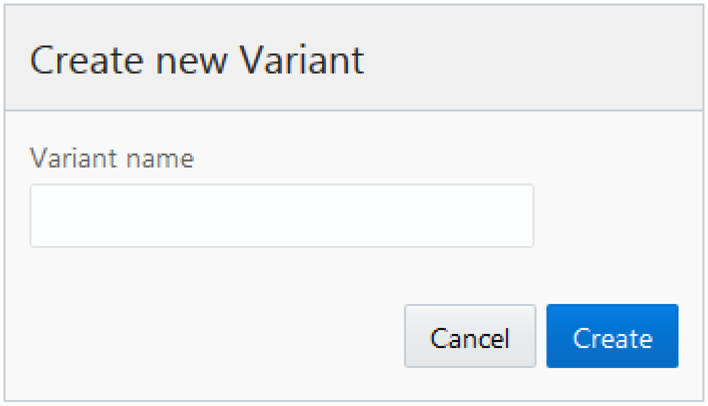
Creating a New Variant

You can create multiple variants for ofsaa-livy Interpreter. To create a new variant, perform the following procedure:

1. Click the following button, from the *ofsaa-livy Interpreter* page:



The *Create New Variant* popup window is displayed:



2. Enter the name of the variant in the **Variant name** field.
3. Click the following button:

   The new variant is created and is displayed in a tab adjacent to the default variable. The new variant is named **ofsaa-livy <variant name>**.

   You can add/update the details of the new variant by filling the fields. For more information, see Table 7–2, "ofsaa-livy Interpreter Values"

You can also perform the following actions from the *ofsaa-livy Interpreter* page:

- Click **Reset** button to reset the values in all the field to default values.
- Update the required fields and click following button to save the changes:

- Click **X** button adjacent to the variant tab to remove a variant.

Note: You cannot remove the default variant.

---

ofsaa-livy-sql Interpreter

ofsaa-livy-sql Interpreter uses built-in functions and Scala functions. Data loaded through these are processed in Spark.

ofsaa-livy-sql Interpreter provides the following functions:

- **loadGraph(code,graphtag [--optional,date])** - to load graph for a particular date in yyyy-mm-dd format.

  Return type is void and should not be assigned to any variable.

  For example: loadGraph("code","graphtag","date");
- **saveGraphAs(graphtag, usertag)** - to create a snapshot of graph against a new tab.

Return type is void and should not be assigned to any variable.

For example: saveGraphAs("graphtag","usertag");

- **listGraphs()** - to list all the available graphs in a tabular form.

  Cannot be used in conjunction with any other code.

  For example: listGraphs()

- **getGraphDetails(code)** - to get the graph details in a tabular form.

  Cannot be used in conjunction with any other code.

  For example: getGraphDetails("code")

- **loadDataset(code)** - to load a particular dataset.

  Returns a DataFrame.

  For example: var ds = loadDataset("code"))

- **loadDataset(code,date)** - to load dataset for a particular date in yyyy-mm-dd format.

  Returns a DataFrame.

  For example: var ds = loadDataset("code","date"))

- **loadDataset(code,startDate,endDate)** - to load dataset for a specific time period with date in yyyy-mm-dd format.

  Returns a DataFrame.

  For example: var ds = loadDataset("code","startDate","endDate"))

- **listDatasets()** - to list all the available datasets in a tabular form.

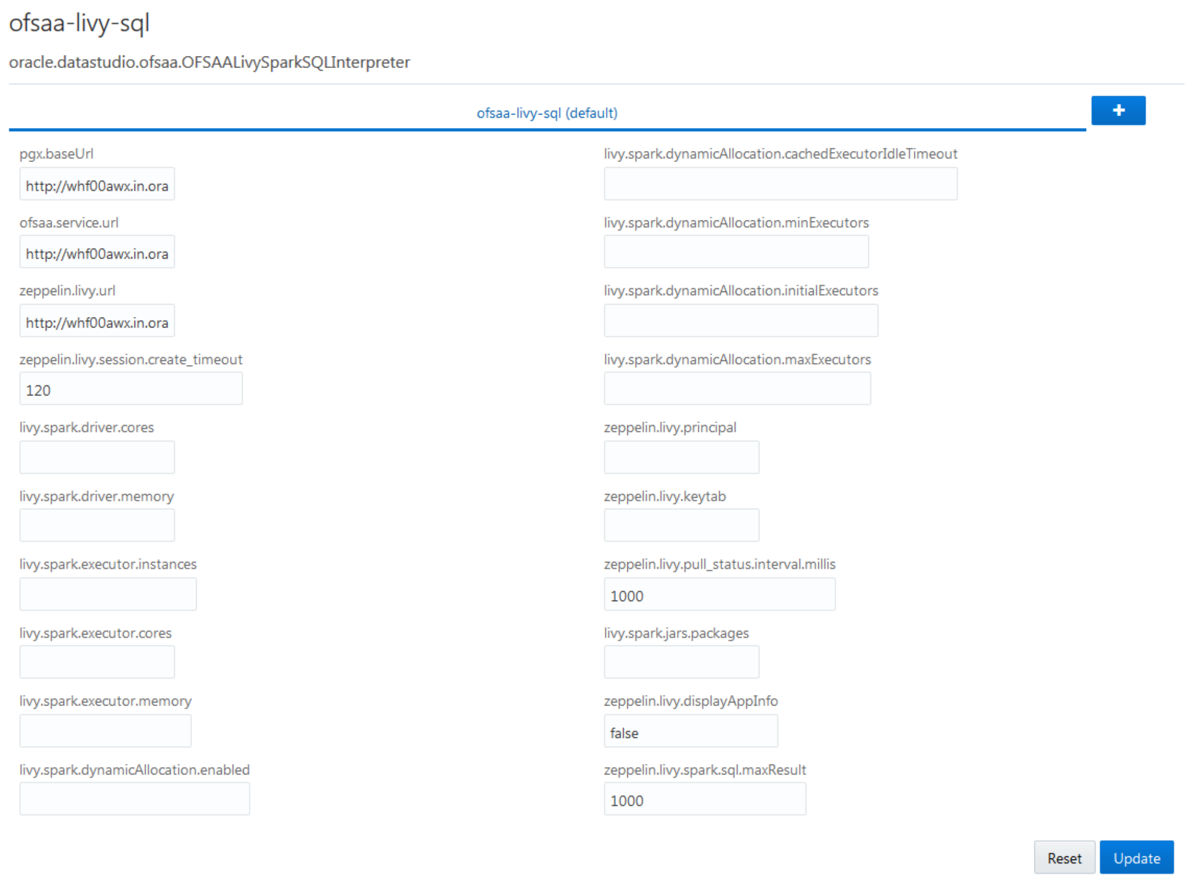
  Cannot be used in conjunction with any other code.

  For example: listDatasets()

Managing Default ofsaa-livy-sql Interpreter

You can access the ofsaa-livy-sql interpreter by clicking **ofsaa-livy-sql** link in the LHS menu of *Interpreters* page. Upon clicking this link, the RHS is displayed with default ofsaa-livy-sql variant and its corresponding values, as depicted in the following figure:

Figure 7–5 ofsaa-livy-sql Interpreter


ofsaa-livy-sql

oracle.datastudio.ofsaa.OFSAALivySparkSQLInterpreter

ofsaa-livy-sql (default)

pgx.baseUrl: http://whf00awx.in.ora	livy.spark.dynamicAllocation.cachedExecutorIdleTimeout:
ofsaa.service.url: http://whf00awx.in.ora	livy.spark.dynamicAllocation.minExecutors:
zeppelin.livy.url: http://whf00awx.in.ora	livy.spark.dynamicAllocation.initialExecutors:
zeppelin.livy.session.create_timeout: 120	livy.spark.dynamicAllocation.maxExecutors:
livy.spark.driver.cores:	zeppelin.livy.principal:
livy.spark.driver.memory:	zeppelin.livy.keytab:
livy.spark.executor.instances:	zeppelin.livy.pull_status.interval.millis: 1000
livy.spark.executor.cores:	livy.spark.jars.packages:
livy.spark.executor.memory:	zeppelin.livy.displayAppInfo: false
livy.spark.dynamicAllocation.enabled:	zeppelin.livy.spark.sql.maxResult: 1000

Table 7–3 ofsaa-livy-sql Interpreter Values

Field	Description
The permissible character range for these fields is between three and 255.	
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**
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.
zeppelin.jdbc.concurrent.use	

Table 7–3 ofsaa-livy-sql Interpreter Values

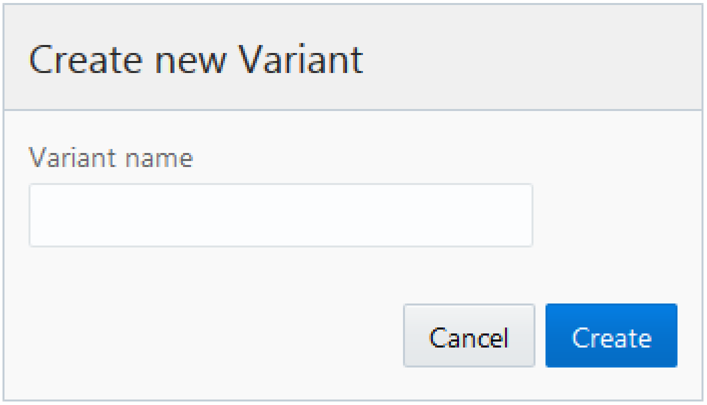
Field	Description
livy.spark.executor.cores	Enter the number of Number of executor cores to use for the driver process.
livy.spark.executor.memory	Enter the amount of memory to use for the executor process.
livy.spark.dynamicAllocation.enabled	This field indicates whether Dynamic Allocation is enabled 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	
zeppelin.livy.keytab	
zeppelin.livy.pull_status.interval.millis	Enter the data pull interval in milliseconds.
livy.spark.jars.packages	
zeppelin.livy.displayAppInfo	This field indicates whether the application information needs to be displayed or not. Enter "true" or "false".

Creating a New Variant

You can create multiple variants for ofsaa-livy-sql Interpreter. To create a new variant, perform the following procedure:

1. Click the following button, from the *ofsaa-livy-sql Interpreter* page:

   The *Create New Variant* popup window is displayed:



2. Enter the name of the variant in the **Variant name** field.

3. Click the following button:

The new variant is created and is displayed in a tab adjacent to the default variable. The new variant is named **ofsaa-livy-sql <variant name>**.

You can add/update the details of the new variant by filling the fields. For more information, see Table 7–3, "ofsaa-livy-sql Interpreter Values"

You can also perform the following actions from the *ofsaa-livy-sql Interpreter* page:

- Click **Reset** button to reset the values in all the field to default values.
- Update the required fields and click following button to save the changes:

- Click **X** button adjacent to the variant tab to remove a variant.

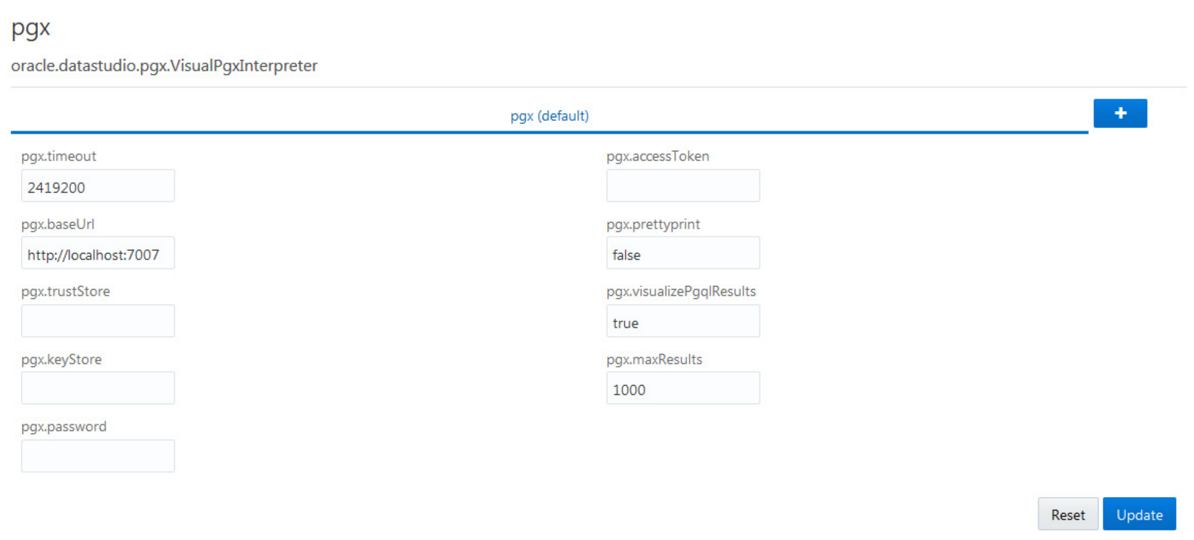
Note: You cannot remove the default variant.

pgx Interpreter

Managing Default pgx Interpreter

You can access the pgx interpreter by clicking pgx link in the LHS menu of *Interpreters* page. Upon clicking this link, the RHS is displayed with default pgx variant and its corresponding values, as depicted in the following figure:

Figure 7–6 pgx Interpreter



pgx

oracle.datastudio.pgx.VisualPgxInterpreter

pgx.timeout	2419200
pgx.baseUrl	http://localhost:7007
pgx.trustStore	
pgx.keyStore	
pgx.password	
pgx.accessToken	
pgx.prettyprint	false
pgx.visualizePgqlResults	true
pgx.maxResults	1000

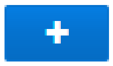
Table 7–4 pgx Interpreter Values

Field	Description
The permissible character range for these fields is between three and 255.	
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**
pgx.keyStore	Enter the Zeppelin session creation timeout in seconds.
pgx.password	
pgx.accessToken	
pgx.prettyprint	
pgx.visualizePgqlResults	
pgx.maxResults	

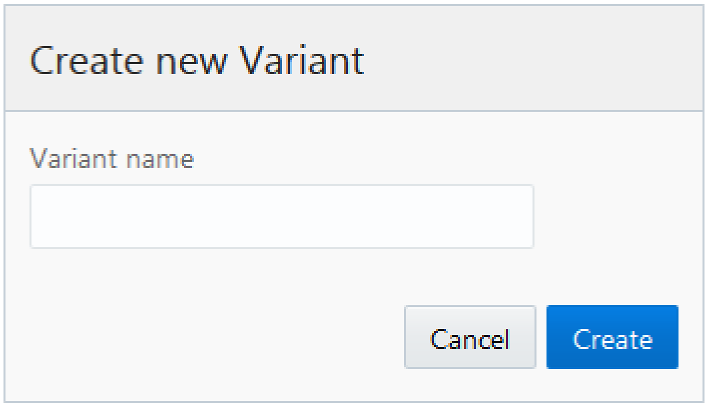
Creating a New Variant

You can create multiple variants for pgx Interpreter. To create a new variant, perform the following procedure:

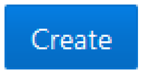
1. Click the following button, from the *pgx Interpreter* page:



The *Create New Variant* popup window is displayed:



2. Enter the name of the variant in the **Variant name** field.
3. Click the following button:



The new variant is created and is displayed in a tab adjacent to the default variable. The new variant is named **pgx <variant name>**.

You can add/update the details of the new variant by filling the fields. For more information, see Table 7–4, "pgx Interpreter Values"

You can also perform the following actions from the *pgx Interpreter* page:

- Click **Reset** button to reset the values in all the field to default values.
- Update the required fields and click following button to save the changes:

- Click **X** button adjacent to the variant tab to remove a variant.

Note: You cannot remove the default variant.

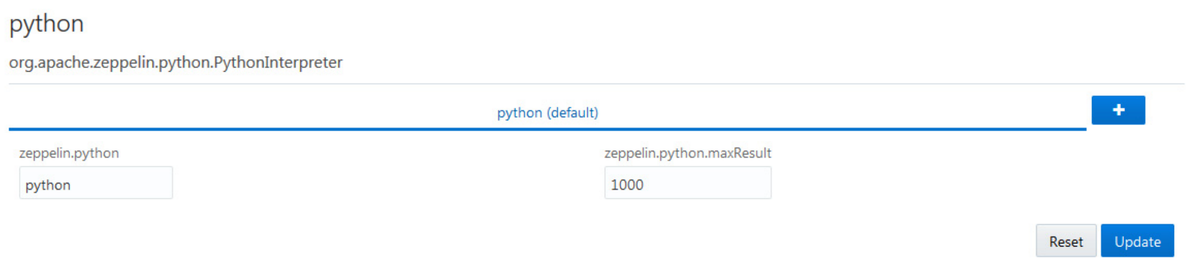
---

python Interpreter

Managing Default python Interpreter

You can access the python interpreter by clicking **python** link in the LHS menu of *Interpreters* page. Upon clicking this link, the RHS is displayed with default python variant and its corresponding values, as depicted in the following figure:

Figure 7–7 python Interpreter


python

org.apache.zeppelin.python.PythonInterpreter

zeppelin.python: python

zeppelin.python.maxResult: 1000

Table 7–5 python Interpreter Values

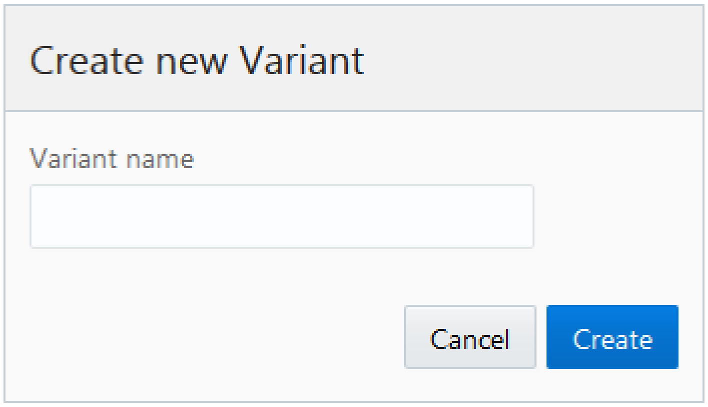
Field	Description
The permissible character range for these fields is between three and 255.	
zeppelin.python	
zeppelin.python.maxResult	

Creating a New Variant

You can create multiple variants for python Interpreter. To create a new variant, perform the following procedure:

1. Click the following button, from the *python Interpreter* page:

The *Create New Variant* popup window is displayed:



2. Enter the name of the variant in the **Variant name** field.
3. Click the following button:

   The new variant is created and is displayed in a tab adjacent to the default variable. The new variant is named **python <variant name>**.

   You can add/update the details of the new variant by filling the fields. For more information, see Table 7–5, "python Interpreter Values"

You can also perform the following actions from the *python Interpreter* page:

- Click **Reset** button to reset the values in all the field to default values.
- Update the required fields and click following button to save the changes:

- Click **X** button adjacent to the variant tab to remove a variant.

Note: You cannot remove the default variant.

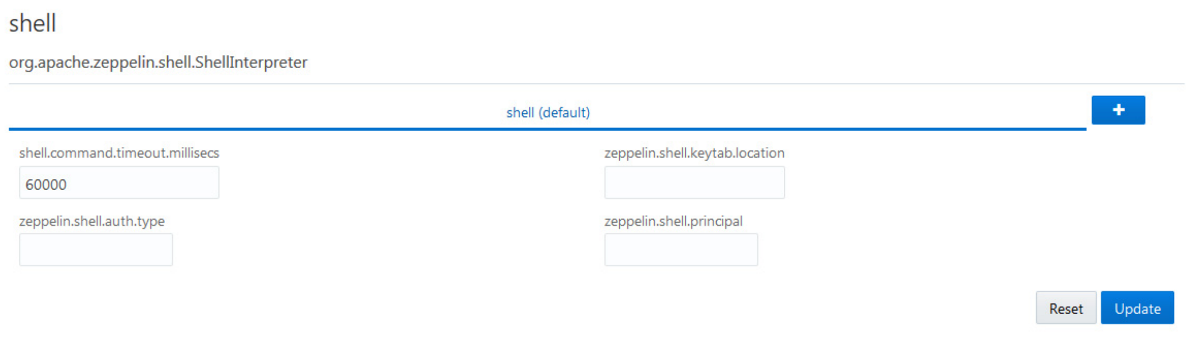
---

shell Interpreter

Managing Default shell Interpreter

You can access the shell interpreter by clicking **shell** link in the LHS menu of *Interpreters* page. Upon clicking this link, the RHS is displayed with default shell variant and its corresponding values, as depicted in the following figure:

Figure 7–8 shell Interpreter


shell

org.apache.zeppelin.shell.ShellInterpreter

shell (default)

shell.command.timeout.millisecs: 60000

zeppelin.shell.keytab.location:

zeppelin.shell.auth.type:

zeppelin.shell.principal:

Table 7–6 shell Interpreter Values

Field	Description
The permissible character range for these fields is between three and 255.	
shell.command.timeout.millisecs	Enter the shell command timeout in milliseconds.
zeppelin.shell.auth.type	
zeppelin.shell.keytab.location	
zeppelin.shell.principal	

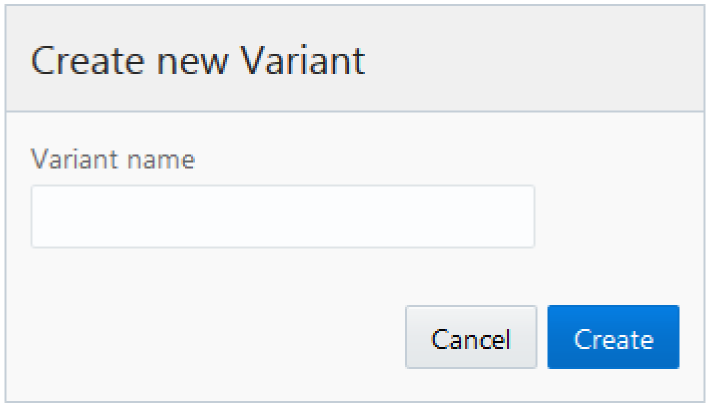
Creating a New Variant

You can create multiple variants for shell Interpreter. To create a new variant, perform the following procedure:

1. Click the following button, from the *shell Interpreter* page:

   +

   The *Create New Variant* popup window is displayed:


Create new Variant

Variant name

2. Enter the name of the variant in the **Variant name** field.

3. Click the following button:

   The new variant is created and is displayed in a tab adjacent to the default variable. The new variant is named **shell <variant name>**.

   You can add/update the details of the new variant by filling the fields. For more information, see Table 7–6, "shell Interpreter Values"

You can also perform the following actions from the *shell Interpreter* page:

- Click **Reset** button to reset the values in all the field to default values.
- Update the required fields and click following button to save the changes:

- Click **X** button adjacent to the variant tab to remove a variant.

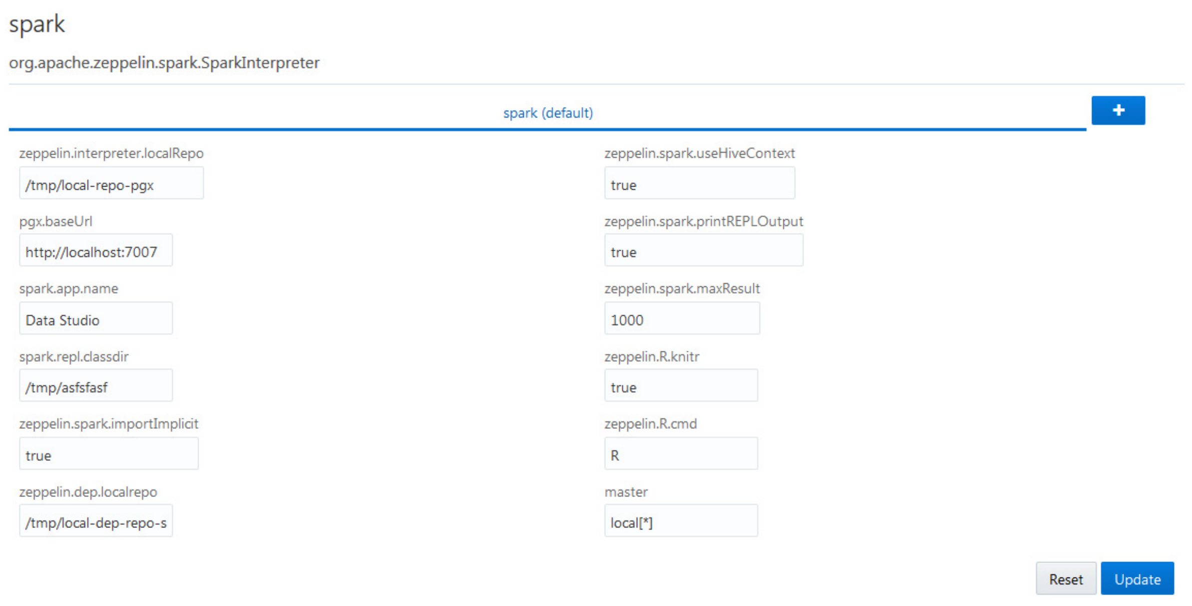
Note: You cannot remove the default variant.

spark Interpreter

Managing Default spark Interpreter

You can access the spark interpreter by clicking **spark** link in the LHS menu of *Interpreters* page. Upon clicking this link, the RHS is displayed with default spark variant and its corresponding values, as depicted in the following figure:

Figure 7–9 spark Interpreter



spark

org.apache.zeppelin.spark.SparkInterpreter

spark (default)

zeppelin.interpreter.localRepo	/tmp/local-repo-pgx
pgx.baseUrl	http://localhost:7007
spark.app.name	Data Studio
spark.repl.classdir	/tmp/asfsfasf
zeppelin.spark.importImplicit	true
zeppelin.dep.localrepo	/tmp/local-dep-repo-s
zeppelin.spark.useHiveContext	true
zeppelin.spark.printREPLOutput	true
zeppelin.spark.maxResult	1000
zeppelin.R.knitr	true
zeppelin.R.cmd	R
master	local[*]

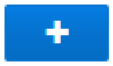
Table 7–7 spark Interpreter Values

Field	Description
The permissible character range for these fields is between three and 255.	
zeppelin.interpreter.localRepo	Enter the zeppelin local repository path in this field. For example: **/tmp/local-repo-pgx**
pgx.baseUrl	Enter the pgx.baseUrl URL in this field. For example: **http://localhost:7007**
spark.app.name	Enter the spark application name in this field.
spark.repl.classdir	
zeppelin.spark.importImplicit	
zeppelin.dep.localrepo	
zeppelin.spark.useHiveContext	
zeppelin.spark.printREPLOutput	
zeppelin.spark.maxResult	
zeppelin.R.knitr	
zeppelin.R.cmd	Enter the cached execution timeout in seconds.
master	

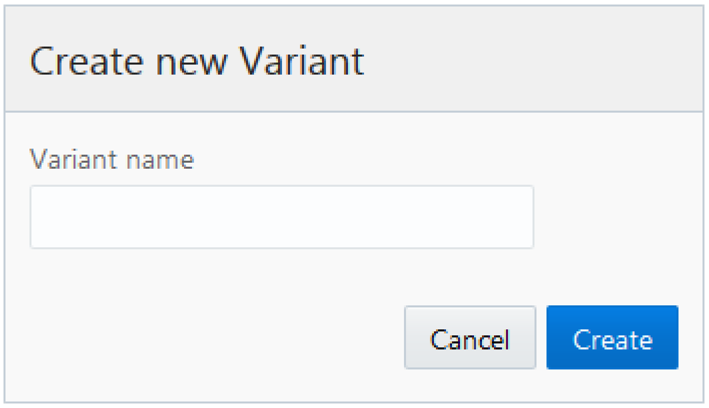
Creating a New Variant

You can create multiple variants for spark Interpreter. To create a new variant, perform the following procedure:

1. Click the following button, from the *spark Interpreter* page:



The *Create New Variant* popup window is displayed:



Create new Variant

Variant name

2. Enter the name of the variant in the **Variant name** field.
3. Click the following button:

The new variant is created and is displayed in a tab adjacent to the default variable. The new variant is named **spark <variant name>**.

You can add/update the details of the new variant by filling the fields. For more information, see Table 7–7, "spark Interpreter Values"

You can also perform the following actions from the spark *Interpreter* page:

- Click **Reset** button to reset the values in all the field to default values.
- Update the required fields and click following button to save the changes:

- Click **X** button adjacent to the variant tab to remove a variant.

Note: You cannot remove the default variant.

A
Workspace Configuration

This section covers the following topics:

- Interpreters
- Manage Paragraphs
- Result View Options

Interpreters

Interpreters facilitate the output generation. OFS CCMS provides you the options to choose between various interpreters which process the data and generates the output. The OFS CCMS uses the following interpreters:

- PGX
- PGQL
- GreenMarl
- OFSAA Interpreter
- OFSAA SQL Interpreter
- Markdown

PGX

You can click the following icon displayed above a Paragraph to use the PGX

interpreter. Upon clicking this icon, a field is enabled to write your code lines. The first line of the code is auto populated with the term "%pgx".

Example A–1 Sample Code

```
%pgx
session.getGraphs().get("PanamaFCCGlobalGraph4")
```

PGQL

You can click the following icon displayed above a Paragraph to use the PGQL

Interpreter. Upon clicking this icon, a field is enabled to write your code lines. The first line of the code is auto populated with the term "%pgql".

Example A–2 Sample Code

```
%pgql
SELECT n,e,m from fccmgraph_11Dec MATCH (n) -[e]-> (m)
```

GreenMarl

You can click the following icon displayed above a Paragraph to use the GreenMarl

Interpreter. Upon clicking this icon, a field is enabled to write your code lines. The first line of the code is auto populated with the term "%greenmarl".

Example A–3 Sample Code

```
%greenmarl
procedure helloWorld()
{
println("Hello World");
}
```

OFSAA Interpreter

You can click the following icon displayed above a Paragraph to use the OFSAA

Interpreter. Upon clicking this icon, a field is enabled to write your code lines. The first line of the code is auto populated with the term "%ofsaa".

Example A–4 Sample Code

```
%ofsaa
loadGraph("LGRAPH","lg","2017-01-01")
```

OFSAA SQL Interpreter

You can click the following icon displayed above a Paragraph to use the PGQL

Interpreter. Upon clicking this icon, a field is enabled to write your code lines. The first line of the code is auto populated with the term "%ofsaa-sql".

Example A–5 Sample Code

```
%ofsaa-sql
select * from ACCOUNT
```

Markdown

You can click the following icon displayed above a Paragraph to use the PGQL

Interpreter. Upon clicking this icon, a field is enabled to write your code lines. The first line of the code is auto populated with the term "%md".

Example A–6 Sample Code

```
%md
**Hello World**
```

Manage Notebooks

This section provides you the details of all the buttons that are used to manage Notebooks.

Table A–1 Manage Notebooks

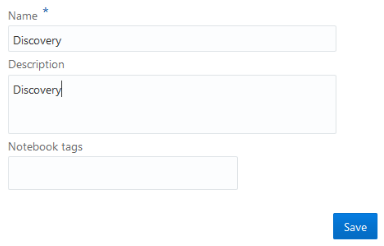
Button/Icon	Action/Description
✎	Click this button to edit the basic details of the selected Notebook. Upon clicking this button, the following window is displayed:  Name * Discovery Description Discovery Notebook tags Add/update the required details. Enter three or more characters, up to a maximum of 32. The Notebook name cannot be empty and cannot start or end with a slash (/). Click **Save** button to save the details.
</>	Click this button to hide/display code in all the Paragraphs.
🗎	Click this button to hide/display results in all the Paragraphs.
▱	Click this button to clear results in all the Paragraphs. Warning: If you clear the results, you will have to re run all the Paragraphs to view the results again.

Table A–1 Manage Notebooks

Button/Icon	Action/Description
🗑	Click this button to remove a Notebook. Upon clicking this button, a confirmation message is displayed. Click the following button to confirm delete action: Delete
↻	Click this button to reset the session.
▶	Click this button to run all the paragraphs.
⧉	Click this button to create a copy (clone) of the current Notebook. All the Paragraphs in the current Notebook are replicated in the new Notebook. The name of the new Notebook is Copy of <current Notebook name>.
💾	Click this button to save the Notebook in your local machine. The details of the Notebook are saved in a file **<notebook name>.dsnb**.
⋖	Click this button to share the current Notebook with another User, User Group, or Role. You can select these from the *Share Notebook* window.
🔒	Click this button to enable/disable read-write protection in Notebook. Once write protected, the Notebook is protected from edit, clear result, delete, reset session, run paragraphs, and share.
↗	Click this button to open the Notebook in iFrame.
🖌	Click this button to select the template.
📓	Click this button to toggle between Zeppelin and Jupyter.
▦	Click this button to switch view between Default, Simple, and Report.

Manage Paragraphs

This section provides you the details of all the buttons that are used to manage Paragraphs.

Table A–2 Manage Paragraphs

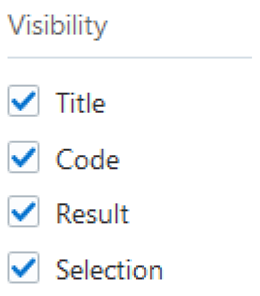
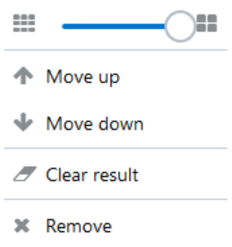
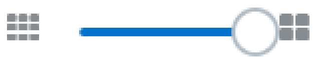
Button/Icon	Action/Description
▶	Click this button to execute the Paragraph.
⤢	Click this button to expand the Paragraph to the window size.
☰	Click this button to show/hide line numbers in code.
👁	Click this button to select the visibility settings. Upon clicking this button, the following window is displayed:  Visibility [✓] Title [✓] Code [✓] Result [✓] Selection select the check box(es) adjacent to the required options.

Table A–2 Manage Paragraphs

Button/Icon	Action/Description
⚙	Click this button to manage the widget size, placement, and so on. Upon clicking this button, the following window is displayed:  You can perform the following actions from this window: ■ Click and drag the following button to manage the Paragraph widget size:  ■ Click **Move Up** button to move the Paragraph one level to the top. ■ Click **Move Down** button to move the Paragraph one level to the top. ■ Click **Clear Result** button to clear the result area. ■ Click **Remove** button to remove the Paragraph.

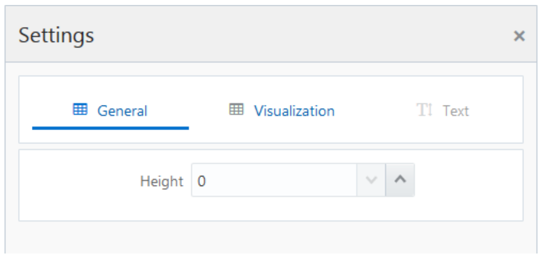
Result View Options

This section provides you the details of all the buttons that are used to manage various views in results.

Table A–3 Manage Notebooks

Button/Icon	Action/Description
▦	Click this button to view the results in tabular format.
📈	Click this button to view the results in area chart format.
📊	Click this button to view the results in bar chart format.

Table A–3 Manage Notebooks

Button/Icon	Action/Description
▼	Click this button to view the results in funnel chart format.
📈	Click this button to view the results in line chart format.
◕	Click this button to view the results in pie chart format.
▲	Click this button to view the results in pyramid chart format.
▦	Click this button to view the results in tree map format.
☼	Click this button to view the results in sunburst chart format.
☁	Click this button to view the results in tag cloud chart format.
⌶	Click this button to view the results in box plot chart format.
🗎	Click this button to view the raw results (unformatted) format.
☰	Click this button to update the settings. Upon clicking this button, the following window is displayed:  Settings: General, Visualization, Text; Height 0. In the *General* tab, enter height in **Height** field. In the *Visualization* tab, enter the number of items per page in the **Number of items per page** field.
⤓	Click this button to download the result. Select the required download option and select the location in you machine to save the file.