Oracle Financial Services
Enterprise Modeling
User Guide

Release 8.0.8.0.0
May 2019

ORACLE®

FINANCIAL SERVICES

ORACLE®

Oracle Financial Services
Enterprise Modeling
User Guide

Contents

Contents

APPENDIX A   *References* *119*

APPENDIX B   *User Groups* *125*

APPENDIX C   ....*EMF Object Migration 131*

List of Figures

List of Tables

List of Tables

Preface

The *Oracle Financial Services Enterprise Modeling User Guide* explains the concepts of Oracle Financial Services Enterprise Modeling (OFS EM) and provides step-by-step instructions for navigating through the application.

This chapter discusses the following topics:

- [Audience](#)
- [How this Guide is Organized](#)
- [Related Documents](#)
- [Conventions Used in This Guide](#)

Audience

The *Oracle Financial Services Enterprise Modeling User Guide* is designed for various users of Oracle Financial Services Enterprise Modeling. This guide is intended for the following users:

- **Business Analysts** (BA) who are instrumental in solution designing and creating statistical models using historical data.

- **System Administrators** (SA) who are instrumental in maintaining and executing batches, making the application secure and operational, and configuring the users and security of the application.

How this Guide is Organized

The *Oracle Financial Services Enterprise Modeling User Guide* includes the following topics:

- Chapter 1: About Oracle Financial Services Enterprise Modeling provides an overview of the Oracle Financial Services Enterprise Modeling, its features, the workflow, user roles, privileges, and actions.

- Chapter 2: General Features, explains the general features of Oracle Financial Services Enterprise Modeling. These are the common User Interface (UI) features that are found across all modules.

- Chapter 3: Getting Started explains how to access and login to the application.

- Chapter 4: Managing Sandbox gives detailed information on Sandbox module, user roles, creating Sandbox definitions, maintaining the Sandbox, and so on.

- Chapter 5: Managing Techniques describes techniques registration, creating, authorizing, editing, and deleting a technique.

- Chapter 6: Managing Variables explains about creating, editing, viewing, and deleting variables.

- Chapter 7: Managing Modeling gives detailed information about creating, editing, viewing, and deleting models. It also explains how to execute and deploy models.

- Chapter 8: Managing Stress Testing describes about creating, editing, viewing, and deleting stress definitions.

- Chapter 9: Creating and publishing models using the integration of Oracle Financial Services Enterprise Modeling and EMF Studio.

Related Documents

This section identifies additional documents related to OFS AAAI Application Pack. You can access Oracle documentation online from the [Oracle Help Center (OHC)Documentation Library](#).

- *Oracle Financial Services Advanced Analytical Applications Infrastructure (OFS AAAI) Application Pack 8.0.8.0.0 Installation and Configuration Guide*

- *Oracle Financial Services Analytical Applications Infrastructure Environment Check Utility Guide*

- *Oracle Financial Services Analytical Applications Infrastructure User Guide*

- *Oracle Financial Services Analytical Applications Infrastructure Administration Guide*

To find additional information about how Oracle Financial Services solves real business problems, see our Web site at www.oracle.com/financialservices.

Conventions Used

Table 1 lists the conventions used in this guide.

Table 1. Conventions Used in this Guide

Convention	Meaning
Italics	- Names of books, chapters, and sections as references - Emphasis
Bold	- Object of an action (menu names, field names, options, button names) in a step-by-step procedure - Commands typed at a prompt - User input
Monospace	- Directories and subdirectories - File names and extensions - Process names - Code sample, including keywords and variables within text and as separate paragraphs, and user-defined program elements within text
<Variable>	Substitute input value

CHAPTER 1 — *About Oracle Financial Services Enterprise Modeling*

This chapter provides an insight to Oracle Financial Services Enterprise Modeling. This chapter includes the following topics:

- Overview
- Concepts of Oracle Financial Services Enterprise Modeling
- Components of Oracle Financial Services Enterprise Modeling
- User Groups and Entitlements

Overview

The core of Financial Institutions Models is Risk, Marketing, Financial Crime and Enterprise Performance Analytical Applications. These models include traditional statistical techniques, modern machine learning methods, computational and simulation models. Oracle Financial Services Enterprise Modeling leverages popular statistical platforms such as the R platform and presents a framework for developing, deploying and managing models at the enterprise level, for financial institutions.

As an enterprise modeling toolkit, Oracle Financial Services Enterprise Modeling enables an institution's IT policies to be enforced while providing flexibility and freedom that Data Scientists and Statistical Modelers desire. Administrative users grant analysts and modelers access to sandboxes - particular analytical subject areas of interest along with a subset of production data - for model building. Validated and approved models may then be promoted from sandboxes to the enterprise model repository. Models in the repository may then be woven into in analytical application flows crafted by mixing data management tasks, model execution and deterministic business logic.

As the use of models proliferate and as modeling becomes a self-service idea within financial institutions, authorized modelers may publish techniques -- parameterized templates of models that serve as building blocks or standardized blueprints for models - so that the best ideas from experienced modelers are captured and reused within the firm. Oracle Financial Services Enterprise Modeling supports techniques developed using R language.

Unique to the needs of large and medium sized financial institutions is the need to project capital levels under a variety of macroeconomic conditions, in order to assess the institution's financial strength under different stress scenarios. The Stress Testing Framework within Oracle Financial Services Enterprise Modeling enables risk and finance officers to define various shocks and scenarios and to apply these conditions uniformly across different model execution runs.

Data lineage and traceability are central to a Financial Intuition's governance process. Oracle Financial Services Enterprise Modeling Application together with the pre-requisite Oracle Financial Services Analytical Applications Infrastructure Application provides a toolkit for developing complete end-to-end analytical applications with data lineage and traceability enabled at every step along the analytical workflow.

Key Features and Advantages

The key features and advantages of Oracle Financial Services Enterprise Modeling are:

Table 2. Features

Model Registry	Centrally Manage and Control all in the enterprise
Statistical Technique Library	Catalog and publish best modeling practices via reusable templates, which are to be used as building blocks for models
Track model usage across applications	Real-time view of dependencies such as which applications use which models; which variables are used a model, etc. Perform what-if impact analysis for changes to data sets, variables and models.
Modeling Sandboxes	Provision sandboxes with subsets of production data for modelers giving modelers complete freedom to build, test and calibrate models using production data, while keeping production environment locked-down for security and compliance needs. Readily deploy built models to locked down production via a built-in workflow.
Integrated Platform (With Oracle Financial Services Analytical Applications Infrastructure)	Statistical modeling, Data management and application deployment integrated in a unified platform so that models may be deployed for use within analytical applications quickly
Execute Models local to data	In-database (Oracle) execution of statistical models for high performance and scalability.
Foundation for Enterprise Stress testing	Enable baseline and stress executions of models defined within the platform

Enterprise-R support with Oracle Financial Services Enterprise Modeling

Oracle Financial Services Enterprise modeling application leverages the R statistical platform. R is an open source statistical programming language and environment for computing and graphics. For more information about R, see the R Project for Statistical Computing at http://www.r-project.org. Oracle R Enterprise (ORE) is an in-database implementation of the R platform. ORE is a component of the Oracle Advanced Analytics Option of Oracle Database Enterprise Edition. ORE enables running R models within the Oracle Database Environment. For more information on ORE, see the Oracle R Enterprise User Guide.

Oracle Financial Services Enterprise Modeling Application supports ORE based models and techniques, that is, models defined and registered within the platform may be executed on an ORE instance. R models may be scripted within the platform or may be imported into the platform from previously existing R scripts.

Concepts of Oracle Financial Services Enterprise Modeling

Oracle Financial Services Enterprise Modeling is built specifically to meet many of the needs of large Financial Institutions where external regulatory and internal governance policies.

Models may only be built and tested in a sandbox environment. A sandbox has to be provisioned and authorized for use (usually by an administrator) before it is visible to modelers. Any number of sandboxes may be provisioned, but generally, an enterprise may provision a sandbox for each department or analytical team. For e.g, there may be an LGD sandbox consisting of data needed to build and validate LGD models, and a separate one for PD modeling. Such segregation of modeling teamwork areas is desirable in practice, but it is not a requirement: here may be as few as a single sandbox for the entire organization

Sandboxes are provisioned along with data required for modeling. Tools are provided to aid administrators in provisioning sandboxes with subsets of production data. Datasets and variables abstract physical data sources from the modeler, and data in the sandbox is exposed to modelers using via datasets and variables. Models are built against datasets and variables not physical data tables and columns. i.e, the underlying data is exposed as a logical dataset within the application, and modelers need not write any database specific queries to obtain data for modeling. It is generally an administrative task to define datasets, and have a menu of datasets available for the modelers. When a sandbox is provisioned, one or more datasets can be associated with the sandbox.

Models in a sandbox can be changed (created/ edited) by anyone with access to the sandbox. Model versions are preserved in the sandbox along with execution and output histories. Once a model has been validated in the sandbox environment, the modeler may request that model to be promoted to the locked-down "production" environment, and once promoted, the promoted model cannot be altered.

Modelers may create new models by using a registered technique from the technique library, as a template. A technique is simply a parameterized and reusable script. An enterprise may publish a menu of techniques and require that modelers use those techniques as the foundation for models. The act of model building is then reduced to selecting an appropriate technique, binding the technique to the appropriate dataset and variables and providing runtime parameters to the script. Generally, a central data sciences team within an enterprise or a department is responsible for publishing techniques.

Not all models can be built using published techniques, and so an alternate way to build a model is to write an R script 'from scratch' and execute the script as a model. Regardless of how the model is built, the model must be bound to dataset and variables before it can access data.

 A business analyst may decide to include a published model (that is a promoted model) in an application flow. For e.g., a capital computation application flow may include many steps, some of which may be steps to execute statistical models for computing PD and LGD. As the Oracle Financial Services Enterprise Modeling application is fully integrated with the Oracle Financial Services Analytical Applications Infrastructure (AAI), models promoted from Oracle Financial Services Enterprise Modeling application are available as tasks in AAI, and so can be included in any orchestrated execution of tasks (application run). Note that within a sandbox, executable models are also made available as tasks private to the sandbox, and so can be included in sandbox specific orchestration of tasks.

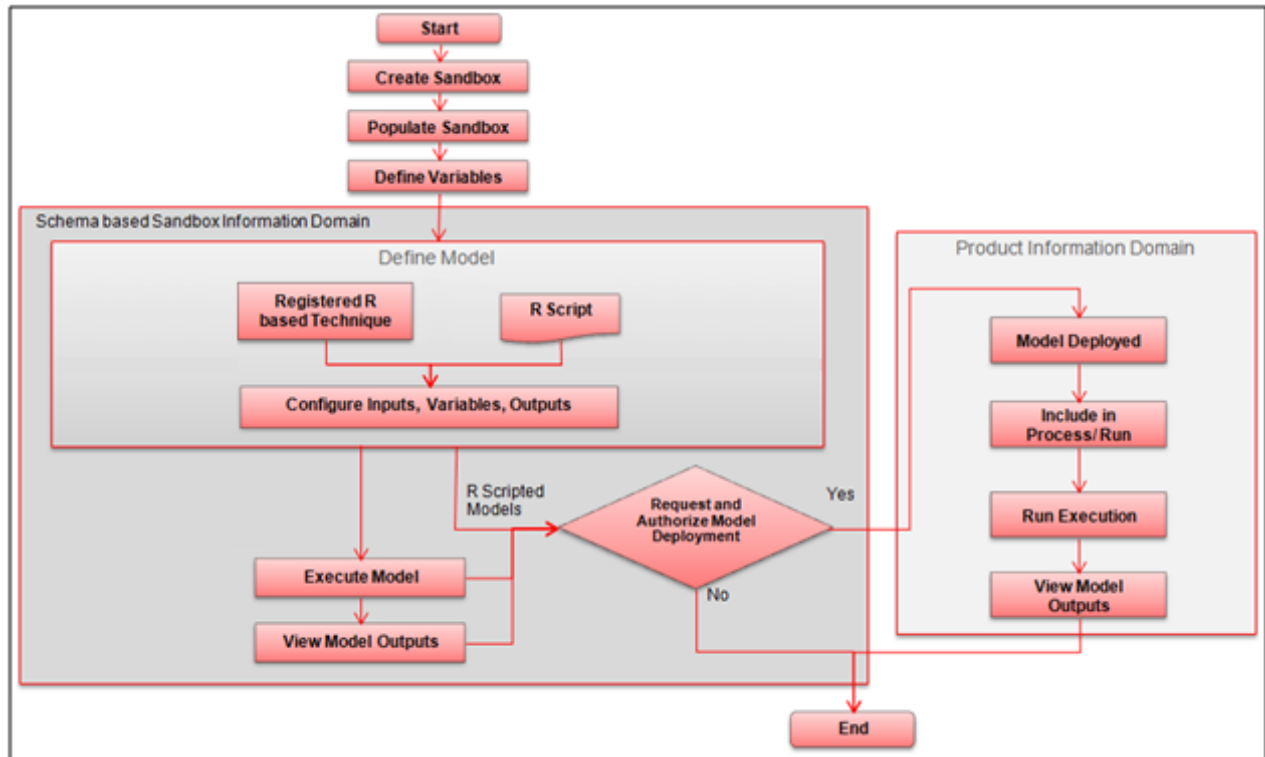## Oracle Financial Services Enterprise Modeling Workflow

Figure 1. Oracle Financial Services Enterprise Modeling Workflow

Components of Oracle Financial Services Enterprise Modeling

This section describes the components of Oracle Financial Services Enterprise Modeling.

The following are the components of Oracle Financial Services Enterprise Modeling application:

- Sandbox

- Sandbox Maintenance

- Techniques Registration

- Variable Definition

- Modeling

- You can create models in a Schema Based sandbox.

Sandbox

Sandbox in Oracle Financial Services Enterprise Modeling refers to a restricted modeling environment, where the Data model is uploaded. It is implemented as an information domain.

The following information domains are required to perform operations in Oracle Financial Services Enterprise Modeling.

- **Sandbox Information Domain**: This information domain is used to create and execute business models. Based on the execution status and the generated output, you can verify the results and deploy the model into production.

You can create the following type of Sandbox in the Oracle Financial Services Enterprise Modeling application.

  - **Physical Sandbox**: A Physical Sandbox is a restricted environment outside the Production information domain where there is actual movement of data from the production information domain to the physical sandbox. Creating a sandbox with multiple datasets eliminates the need for having a sandbox definition for each dataset. You can upload the data model while defining the sandbox, or upload it later using the Import Model option in Data Model Maintenance. When you save the sandbox definition, the required tables are created in the sandbox information domain. However, data present in the tables are copied only after authorizing sandbox population from the Sandbox Maintenance.

- **Production Information Domain**: This information domain allows you to request for model execution, and generate model outputs.

Note: Ensure the data model of the sandbox information domain should be a sub-set of the data model of the production information domain.

Sandbox Maintenance

Sandbox Maintenance helps you to populate the data to the tables in the Sandbox information domain, based on the dataset and the filters in the Sandbox definition. You have the option to do complete or incremental sandbox data population.

The Oracle Financial Services Enterprise Modeling application enables you to synchronize the different versions of a Data Model which exists in a Production and Sandbox Information Domain through Incremental Data Model

Upload. You can refresh the details and fetch the incremental data model changes from Production to Sandbox Information Domain.

Techniques Registration

Technique is a set of generalized statistical algorithms which can be used to build analytical models. Oracle Financial Services Enterprise Modeling application is based on techniques developed using R script and ORE functions.

Variable Definition

Variable refers to a logical set of attributes that are likely to change based on the selected parameters. In a modeling environment variable plays a vital role in filtering the model parameters and to derive an estimate based on historical data. Variables are defined in production information domain.

Modeling

Modeling refers to the process of designing a prototype based on a structured data model, considering all the variables for statistical analysis and to simulate real events and processes.

You can use the Modeling utility to measure and quantify risk. You can use the pre-defined models to predict business trends and validate the existing models. You can use R scripting (using R functions as well as ORE functions) or Open R to create business models. See *Enterprise-R support with Oracle Financial Services Enterprise Modeling* section for more information.

Oracle Financial Services Enterprise Modeling enables you to run and execute R functions as well as ORE functions in the database, thereby greatly increasing scalability and performance.

You can create models in a Schema Based sandbox.

User Groups and Entitlements

The following table gives the details about the User Groups in the Oracle Financial Services Enterprise Modeling Application.

Table 1. User Groups

User Group Name	User Group Description
Modeling Administrator	User mapped to this group will have access to all the menu items for Enterprise modeling and will be have authorization rights for sandbox population, model deployment and modeling technique authorization.
Modeler Group	User mapped to this group will have access to all the menu items for Enterprise modeling but will not have authorization rights for sandbox population, model deployment and modeling technique authorization.

CHAPTER 2 — *General Features*

This chapter explains the general features of Oracle Financial Services Enterprise Modeling application UI. These are the common features that are found across the modules of Oracle Financial Services Enterprise Modeling. It describes the organization of the user interface and provides step-by-step instructions for navigating through the application.

This chapter includes the following topics:

- Home Page Components
- Acronyms

Home Page Components

The Home Page contains the following sections.

- Header
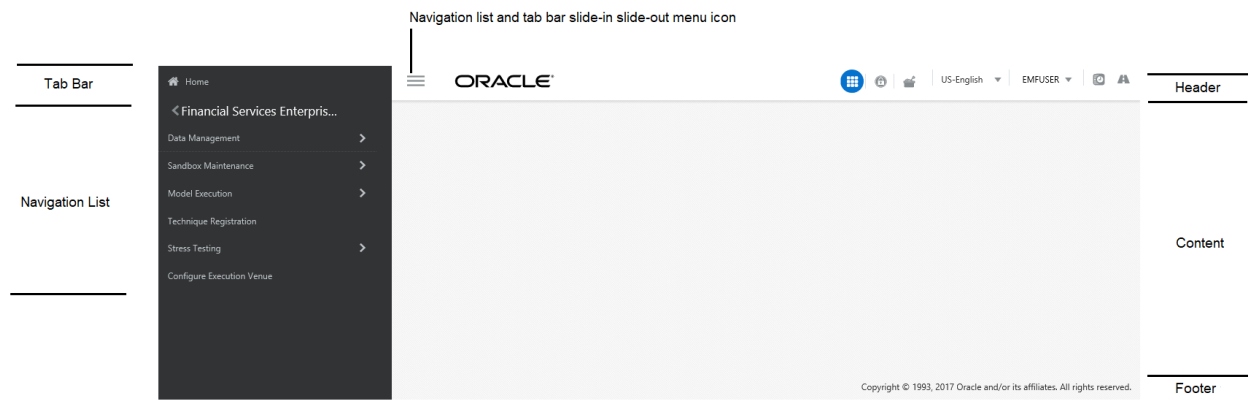- Navigation List
- Tab Bar
- Content
- Footer

Figure 2. Home Page

Header

The Header displays icons, buttons and text for generic information and access to the OFSAA application's features. The following user-interface elements are displayed for EMF:

- **Applications:** Click this icon to display applications in a Tiles menu on the content window.

- **Administration:** Click this icon to display administration tools in a Tiles menu on the content window.

- **Sandbox:** Click this icon to display Sandboxes in a Tiles menu on the content window.

- **Language:** Displays the selected language. Click to select from the options in the drop down.

- **User Name:** Displays the logged in user name. Click to select from the following options in the drop down:

  - **Preferences:** Select to set the Home page.

  - **About:** Select to view the copyrights and third-party information.

  - **Change Password:** Select to change the password in the Change Password window.

  - **Logout:** Select to log out of the application.

- **Last login date and time, and last failed login date and time**: Click to view the last login date and time, and the last failed login date and time.

- **Discover/Explore OFSAA Apps**: Click to view the OFSAA applications web page for more details.

Applications

The applications available are displayed in a Tiles menu on the content window. Click the Tiles to open the selected application. For EMF, the following applications are available:

- **Financial Services Enterprise Modeling**

  - Data Management
    For more information, see the *Oracle Financial Services Analytical Applications Infrastructure User Guide.*

  - Sandbox Maintenance

  - Model Execution

  - Technique Registration

  - Stress Testing

  - Configure Execution Venue

- **Common Tasks**
  For more information, see the *Oracle Financial Services Analytical Applications Infrastructure User Guide.*

  - Data Model Management

  - Unified Analytical Metadata

  - Operations

  - Data Entry Forms and Queries

  - Utilities

  - Object Administration

■   Metadata Browser

Administration

The administration feature displays on Tiles menu. Click the Tiles navigate further. For more information, see the *Oracle Financial Services Analytical Applications Infrastructure User Guide.* For EMF, the following applications are available:

- Translation Tools

- Processing Modeling Framework (link to PMF User Guide)

Sandbox

The Sandbox icon from the header displays options in Tiles menu. Click on the menu to display the following menu items:

- Modeling

- Model Creation

- Model Execution Status

- Model Output

- Model Deployment

Sandbox option is available only if EMF is enabled in your OFSAA application.

Navigation List

The Navigation list is a slide-in slide-out UI element that appears on clicking the Menu (three lines) icon. This element displays a list of links in a menu based on the application selected from the Applications menu and the access rights assigned to the logged in user. Click on the various links in the list to navigate further and open in the content window. For more details, seethe *Oracle Financial Services Analytical Application Infrastructure User Guide.*

Tab Bar

The Tab bar contains icons and text to navigate to specific OFSAA application modules. Click on the Home button in Tabs to navigate to the Applications page on the content window. The elements displayed in Tabs is dependent on the OFSAA application logged into and the access roles assigned to the user.

Acronyms

The following table describes the acronyms commonly used in this application.

Table 2. Acronyms

Acronyms	Description
BA	Business Analysts
Infodom	Information Domain
LHS menu	Left hand side menu
OFSAAI	Oracle Financial Services Analytical Application Infrastructure
OFSAA	Oracle Financial Services Analytical Applications
Production Infodom	Production Information Domain
RHS menu	Right hand side menu
Sandbox Infodom	Sandbox Information Domain
SA	System Administrator
URL	Uniform Resource Locator
UI	User Interface

CHAPTER 3 *Getting Started*

This chapter explains different steps to login to the Oracle Financial Services Enterprise Modeling application.

The following sections are included in this chapter:

- Logging into the Oracle Financial Services Enterprise Modeling Application
- Changing Password

Logging into the Oracle Financial Services Enterprise Modeling Application

Once the application is installed and configured, you can access Oracle Financial Services Enterprise Modeling Application.

To access Oracle Financial Services Enterprise Modeling, follow these steps:

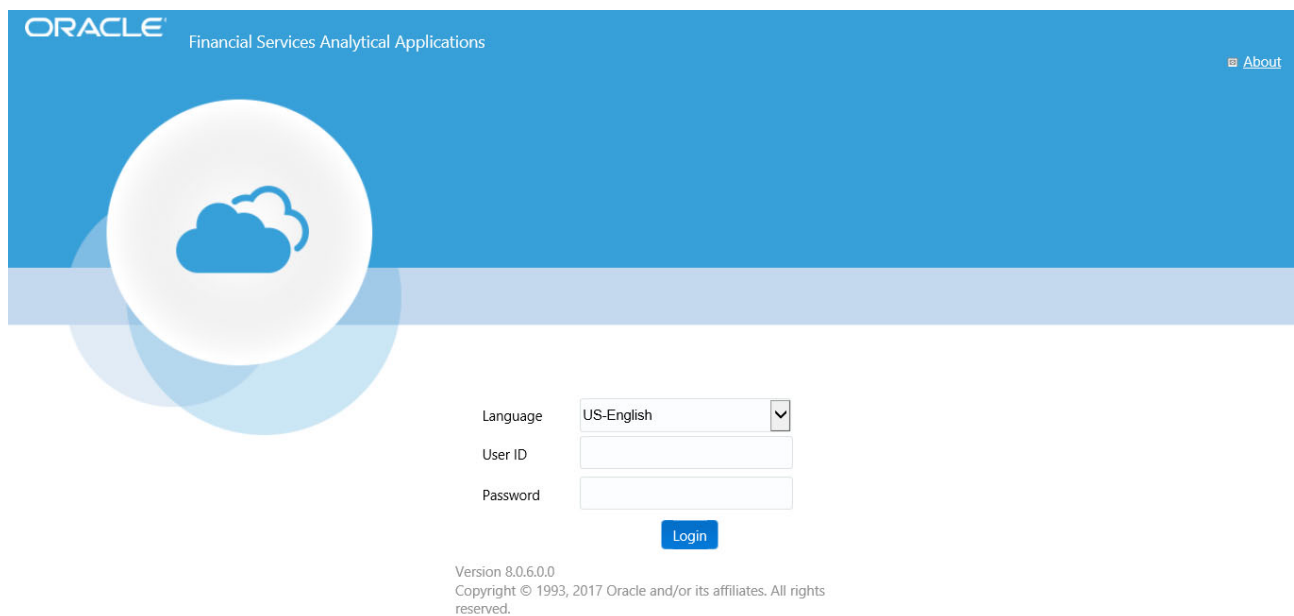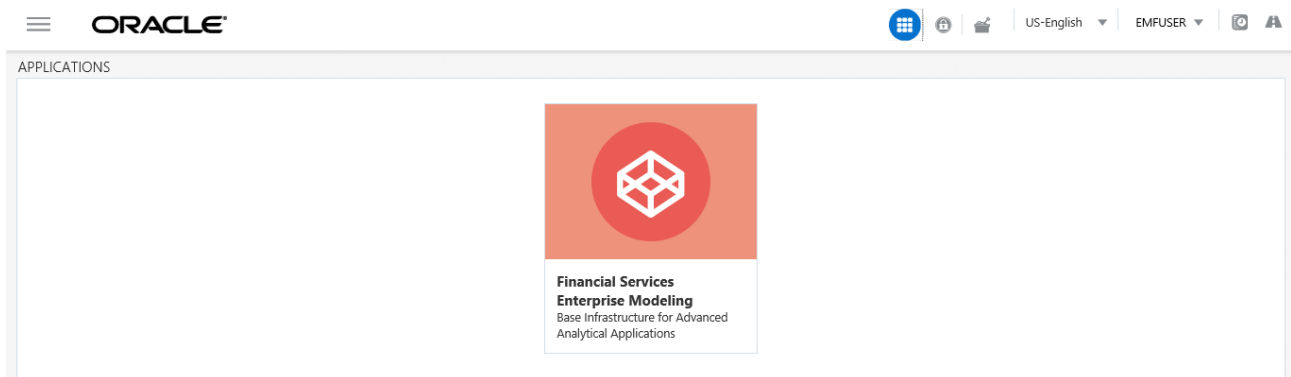1. Enter the OFSAAI URL in your browser. The Login window is displayed.

Figure 3. Oracle Financial Services Enterprise Modeling Application Login Page

2. Select the **Language**.

3. Enter your **User ID** and **Password**.

4. Click **Login**.

5. Click ⊞ from the header to display the applications in a Tiles menu.

6. Select **Financial Services Enterprise Modeling** from the Tiles menu. The Enterprise Modeling Application Home Page is displayed with the Navigation list to the left.

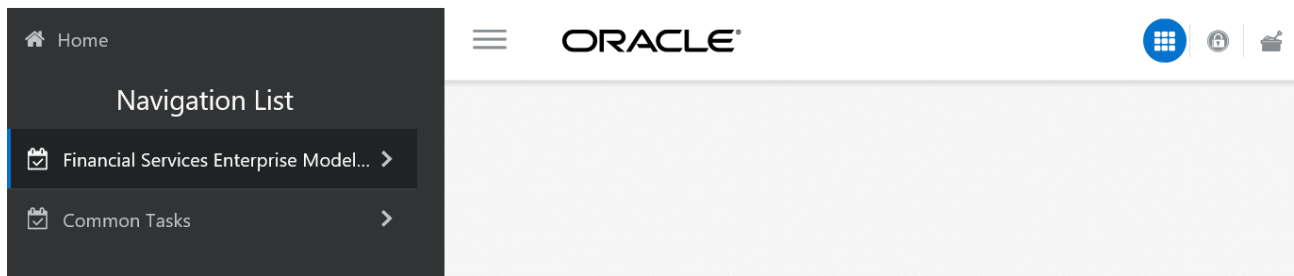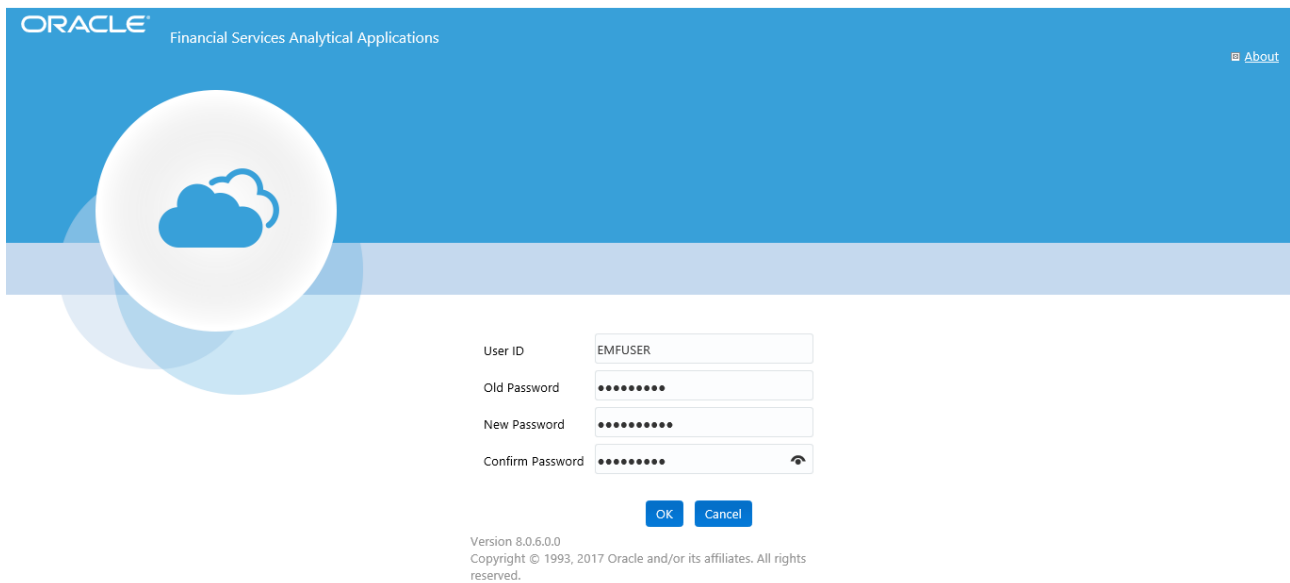**Figure 4. Home Page**

Changing Password

The first time you log in to the application, the *Change Password* window is displayed. Alternatively, you can also choose to change the password at any time.

To change password, follow these steps:

1. Click the **User Name** drop down and select **Change Password** from the header at the top to display the *Change Password* window.

Figure 5. Change Password Window

2. Enter the **User ID**.

3. Enter the **Old Password**.

4. Enter **New Password** and re-enter in the **Confirm Password** field.

5. Click **OK**.

Note:

- Passwords are displayed as asterisks while you enter. This is to ensure that the password is not revealed to other users.

- Ensure that the entered password is at least six characters long.

- The password must be alphanumeric with a combination of numbers and characters.

- The password should not contain spaces.

- Passwords are case sensitive and ensure that the Caps Lock is not turned ON.

- By default, the currently used password is checked for validity if password history is not set.

- The new password should be different from previously used passwords based on the password history, which can be configured. If you encounter any of the following problems, contact the System Administrator:

  - Your user ID and password are not recognized.

  - Your user ID is locked after three consecutive unsuccessful attempts.

  - Your user ID has been disabled.

  - Guest user cannot change the password.

CHAPTER 4 — *Managing Sandbox*

This chapter provides information about Sandbox in the Oracle Financial Services Enterprise Modeling application and step-by-step instructions to use this section.

This chapter includes the following topics:

- About Sandbox
- Workflow
- User Roles
- Managing Sandbox

About Sandbox

Sandbox is a restricted environment in which you can analyze the data on trial and error basis and come up with actual analysis that helps predict the risks and business opportunities for banking institutions. You can create a sandbox by selecting the required datasets and the information domain which you want to implement as the sandbox.

Workflow

This section explains the workflow of Sandbox. There is one type of Sandbox: Physical Sandbox

Physical Sandbox Workflow

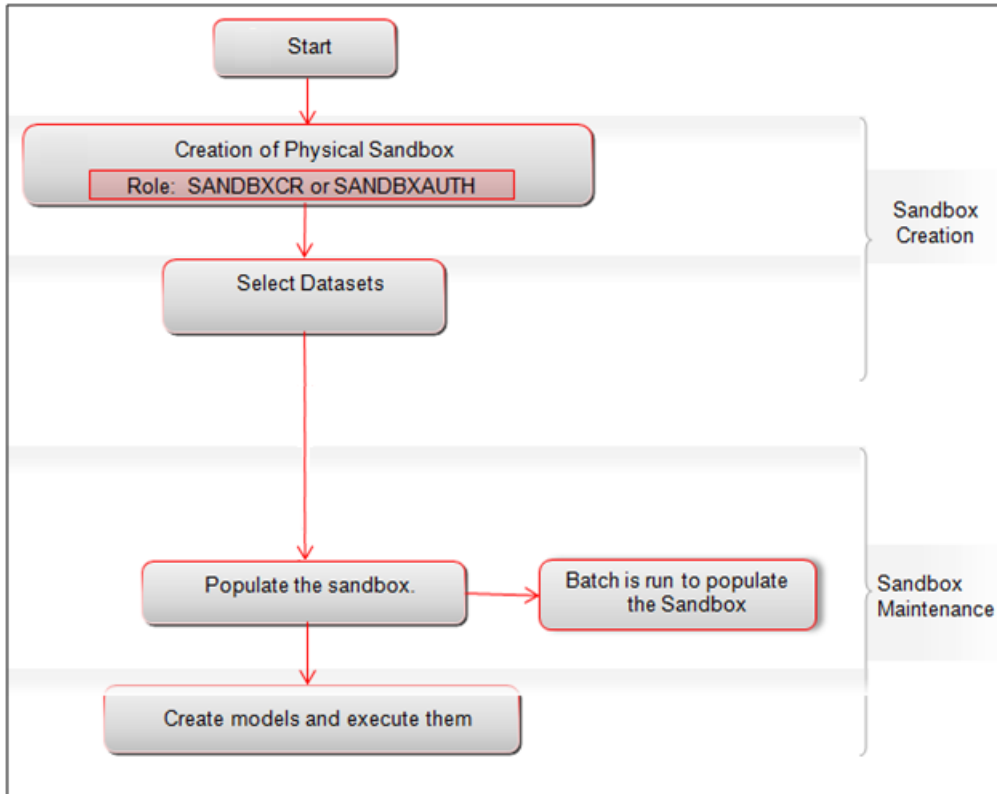The following image gives a detailed workflow of Physical Sandbox.

Figure 6. Physical Sandbox Workflow

User Roles

This section explains about the user roles required to define a Sandbox.

To create a sandbox, you must have the following use roles:

- SANDBXCR
- SANDBXAUTH

An authorizer must authorize the sandbox population and deletion.

To populate and delete a Sandbox, you must have the following user roles:

- SANDBXMOD
- SANDBXAUTH

Note: For more information on User Groups, see Appendix A, "References."

Managing Sandbox

This section explains how to manage Sandbox feature in Oracle Financial Services Enterprise Modeling application. This section includes the following topics:

- Accessing Sandbox Definition
- Creating Sandbox
- Viewing Sandbox Definition
- Modifying Sandbox Definition
- Searching Sandbox Definition
- Sandbox Maintenance

Accessing Sandbox Definition

You can access the Sandbox Definition page from the Oracle Financial Services Enterprise Modeling Home page. To access the Sandbox Definition page, follow these steps:

1. Login to Oracle Financial Services Enterprise Modeling.

2. Click ⊞ from the header to display the applications in a Tiles menu.

3. Select **Financial Services Enterprise Modeling** from the Tiles menu. The Enterprise Modeling Application Home Page is displayed with the Navigation list to the left.

4. Click **Financial Services Enterprise Modeling** from the Navigation list to expand view the menu.

5. Click **Sandbox Maintenance** from the expanded Navigation list to view the next level of links in the menu. Click **Sandbox Creation** to display the *Sandbox Definition* window.

Figure 7. Sandbox Definition Page

Creating Sandbox

This feature allows you to create a sandbox where you can create and execute models.

Note:

1. After creating a Sandbox you should refresh the screen, so that the newly created sandbox is displayed in the *Sandbox* tab.

2. If your OFSAA application is configured for Data Redaction (Advanced Security options), then your system administrator has to give certain Grants related to Data Redaction to the Sandbox schema for model execution to execute. See the section *Data Redaction Grants to Sandbox Schema* in the OFSAAI 8.0.8.0.0 Administrator Guide.

This section includes the following topics:

- Creating Schema-based Sandbox Definition

Creating Schema-based Sandbox Definition

A Schema-based Sandbox is a restricted environment outside the Production information domain, where there is actual movement of data from the production information domain to the physical sandbox. A schema-based sandbox is also called a physical sandbox. A sandbox can be created with multiple datasets, which eliminates the need for having a sandbox definition for each dataset.

To create a schema-based sandbox, follow these steps:

1. Select ➕ from the *Sandbox Definition* tool bar. The *Sandbox Definition* window is displayed.

Note: The Add button is disabled if you select a checkbox in the grid.

Figure 8. Schema-based Sandbox Definition Window

2. Enter the following details:

Table 3. Schema based Sandbox - Field Descriptions

| Field	Description
Fields marked in red asterisk (*) are mandatory.	
Sandbox Name	Enter a name for the sandbox. **Note:** Ensure that there are no special characters like `, {,},", ', ~, <,>, /, \, and multiple spaces. **Note:** Enter the alphabetical characters for the Sandbox Name in either uppercase or lowercase. If you enter the Sandbox Name in mixed case, then the EMF-R Execution fails with errors. For example, Sandbox Name can be **ofssdbox180** or **OFSSDBOX180**. If you have existing Sandbox Names in mixed case, then you must edit and update them from the **Database Details** window. For more information, see the **Database Details** section in the Oracle Financial Services Advanced Analytical Applications Infrastructure User Guide.
Sandbox Description	Enter a brief description for the sandbox.

Table 3. Schema based Sandbox - Field Descriptions

Field	Description
Source Dataset	Select the required datasets. The datasets you want to select should match the Data Model of the sandbox. 1. Click ⬚ and open the Source Dataset Browser. All the Authorized Datasets in the selected production Infodom are listed in the Hierarchical Browser. 2. Select the required dataset from the Datasets pane and click ⬚ . To deselect a dataset select it from the Selected Datasets pane and click ⬚ . 3. Click ⬚ to view the details of the selected datasets. 4. Click ➕ to create a dataset. For more details on creating a dataset, see *Creating Data Set* section in *Oracle Financial Services Analytical Applications Infrastructure User Guide*. **Note**: The Datasets based on Derived Entities are not supported.
Is data model upload required?	By default, the Yes option is selected. • Select **Yes** to upload the data model while saving the sandbox definition. • Select **No** if the data model is already uploaded.
Sandbox Location	This is the information domain you want to implement as sandbox. 1. Click ⬚ and open the Sandbox Location Browser. • If Yes is selected for Is data model upload required? option, the Sandbox Location pane displays the list of the Information Domains where data model is not uploaded and sandbox has not been defined. • If No is selected for Is data model upload required? option, the Sandbox Location pane displays the list of the Information Domains where data model is uploaded and sandbox has not been defined. 2. Select the required infodom and click ⬚ . To deselect the infodom select it from the Selected Information Domain pane and click ⬚ .
Non Time Hierarchy	1. Click ⬚ in the Non Time Hierarchy toolbar. The Hierarchy Browser displays all the Hierarchies/Filters from each selected Datasets. You can click "+" to expand the node to view the sub levels. 2. Select the required hierarchies and click ⬚ . To deselect hierarchy by select it from the Selected Members pane and click ⬚ . 3. Click **OK**. **Note:** Filter conditions are not considered while loading data in to the Hive based sandboxes. To load data with filters, you can use Data Mapping (T2T) to populate the sandbox. Otherwise, you can apply model level filters while modeling.

Table 3. Schema based Sandbox - Field Descriptions

Field	Description
Hierarchy member	Click [icon] in the Hierarchy Member toolbar and specify the members or nodes in the Hierarchy browser. You can click "+" to expand the node to view the sub levels. When you select a hierarchy member, the data belonging to that hierarchy will be copied to the sandbox.
Time Hierarchy	1. Click [icon] in the Time Hierarchy toolbar. The Hierarchy Browser displays the available Time Hierarchies which are common across all the selected Datasets. You can click "+" to expand the node to view the sub levels. 2. Select the required time hierarchy and click [icon] to move it to the Selected Hierarchies pane. To deselect time hierarchies select from the Selected Hierarchies pane and click [icon]. 3. Click **OK**. **Note:** Time hierarchy selection is optional. However, if you do not select a time hierarchy, the following will occur: • Time referencing will not be applicable for model definitions. • For the model definitions, the variable browser will not show any variable definitions that has got filters. • The shocking of transition matrix multi factor models will not work in sandboxes • For models defined using techniques transition matrix - EWMA, MCEM, MC simulations, the execution may fail
Hierarchy Member	Click [icon] in the Hierarchy Member toolbar and specify the members or nodes in the Hierarchy browser. You can click "+" to expand the node to view the sub levels. Time hierarchy is used to group data based on time (year or month). When you select time hierarchy member, the data belonging to that time hierarchy will be copied to the sandbox.

   3. Click **Save**. The tables of the selected dataset get populated in the sandbox information domain.

Viewing Sandbox Definition

You can view individual Sandbox Definition details at any given point. To view the existing Sandbox Definition details in the Sandbox Definition window, follow these steps:

   1. Select the checkbox adjacent to the Sandbox ID.

   2. Click [icon] in the Sandbox Definition tool bar.
      The Sandbox Definition View window is displayed with the details such as Sandbox Name, Sandbox Description, Source Dataset, Sandbox Locations, Time, and Non Time Hierarchy and Hierarchical members.

Modifying Sandbox Definition

You can update only the description and Time Hierarchy Member (if it is selected in the definition) of an existing schema based Sandbox Definition. To modify Sandbox Definition, follow these steps:

1. Select the checkbox adjacent to the Sandbox ID whose details are to be updated.

2. Click [image] in the Sandbox Definition tool bar. The Edit button is disabled if you have selected multiple Sandbox IDs.

3. Modify the Sandbox Definition details as required. For more information, see the *Creating Sandbox* section.

Note: You can edit Time and Non-Time Filters in Sandbox Definition Edit page.

Searching Sandbox Definition

The Sandbox Definition page allows you to search the sandboxes that you want to view.

To search for Sandbox, follow these steps:

1. Navigate to the Sandbox Definition page.

Note: By default, the Sandbox Definition page displays all Sandboxes.

2. Enter the name of the sandbox you want to search.

3. Click [image]. The filtered list is displayed.

Sandbox Maintenance

The Sandbox Maintenance window is used to map user groups to the sandbox definition which enables the users to create models in the sandbox.

This section includes the following topics:

- Accessing Sandbox Maintenance
- Searching Sandbox Maintenance
- Requesting and Authorizing to Populate Sandbox
- Requesting and Authorizing to Delete Sandbox
- Populate Data in the Sandbox
- Incremental Data Model Upload

Note: Once the Incremental changes are updated to the Data Model in the Sandbox Information Domain, you need to re-save the details in the Sandbox definition window. For more information, see the *Modifying Sandbox Definition* section.

Accessing Sandbox Maintenance

You can access the Sandbox Maintenance page from the Oracle Financial Services Enterprise Modeling Home page.

To access the Sandbox Maintenance, follow these steps:

1. Login to Oracle Financial Services Enterprise Modeling.

2. Click ▦ from the header to display the applications in a Tiles menu.

3. Select **Financial Services Enterprise Modeling** from the Tiles menu. The Enterprise Modeling Application Home Page is displayed with the Navigation list to the left.

4. Click **Financial Services Enterprise Modeling** from the Navigation list to expand view the menu.

5. Click **Sandbox Maintenance** from the expanded Navigation list to view the next level of links in the menu. Click **Sandbox Maintenance** to display the *Sandbox Maintenance* window.

Sandbox Maintenance						?
					Q Search ↺ Reset	
		Sandbox Name				
✎ Edit ↺ Incremental Data Model Upload						Records Per Page 1
⚬	Sandbox ID	Sandbox Name ▲	Population Request Action	Deletion Request Action	Population Is Authorized	Deletion Is Authorized	Incremental Upload Required
☐	1520494756379	SBQ1	N	N	N	N	Y

Figure 9. Sandbox Maintenance Page

Searching Sandbox Maintenance

The Sandbox Maintenance page allows you to search the sandboxes that you want to view.

To search for Sandbox Maintenance, follow these steps:

1. Navigate to the *Sandbox Maintenance* window.

2. Enter the name of the sandbox you want to search.

3. Click 🔍. The filtered list is displayed.

Requesting and Authorizing to Populate Sandbox

Note:
The data type **Timestamp** is not supported during sandbox population.

To request and authorize to populate sandbox in the Sandbox Maintenance window, follow these steps:

1. Select the sandbox which you want to populate and click ✎ in the *Sandbox Maintenance* toolbar. The **Edit** button is disabled if you have selected multiple checkboxes. The *Sandbox Maintenance (Edit Mode)* window is displayed.

Note: While populating a Sandbox both Time and Non-Time Filters can be edited in *Sandbox Maintenance* window.

2. In the *Request Action* tab, select **Complete** for **Populate Sandbox** to copy the required table data from the Production infodom to the Sandbox infodom based on the sandbox definition.

3. Click the **Authorize** tab, and select the **Populate Sandbox - Complete/ Incremental** checkbox to authorize sandbox population. This tab is enabled only if your user role is mapped to the function SANDBXAUTH.

4. Click **Save** to confirm changes.
   On authorization, a Sandbox-Populate batch is registered in the OFSAA Infrastructure *Operations* module. The batch will be available in the *Batch Scheduling* window with the Sandbox ID. This batch must be triggered from the *Batch Scheduling* window to complete the data population.

Note: You can populate data through the Batch Mode or Bulk Mode. For more information, see the Populate Data in the Sandbox section in this guide.

Requesting and Authorizing to Delete Sandbox

To request and authorize to delete sandbox in the Sandbox Maintenance window, follow these steps:

1. Select the sandbox which you want to delete and click ☑ in the Sandbox Maintenance tool bar. The Edit button is disabled if you have selected multiple checkboxes. The *Sandbox Maintenance (Edit Mode)* window is displayed.

2. In the *Request Action* tab, select the **Delete Sandbox** checkbox.

3. Click the **Authorize** tab, and select the **Delete Sandbox** checkbox to authorize the deletion. This tab will be enabled only if your user role is mapped to the function SANDBXAUTH.

4. Click **Save** to remove the Sandbox Definition.

Populate Data in the Sandbox

You can populate the Sandbox using the following modes of data population in the application:

- Batch Mode (default mode)
- Bulk Mode

Batch Mode of Data Population

The Batch Mode of data population is optimized for better memory utilization where in it moves the data in a batch size of 1000. This is the default mode to populate data in the Sandbox.

Note: The batch for data population is available in the Batch Scheduling window with the Sandbox ID. Trigger this batch to populate the data. For more information, see the **Batch Scheduler** section in the Oracle Financial Services Advanced Analytical Applications Infrastructure User Guide.

Bulk Mode of Data Population

The Bulk Mode of data population when enabled significantly improves the performance of the Sandbox Data Population.

Prerequisites

Before enabling the Bulk Mode, run the following SQL statement in the production schema and grant SELECT privileges on the production tables (database tables that are part of the dataset mapped to the Sandbox) to the Sandbox database (db) user. This is required because select privileges are not granted in OFS AAI at the time of the creation of the Sandbox or Infodom creation.

```
begin
for x in ( select tname from tab where tabtype='TABLE')
loop
execute immediate 'grant select on ' || x.tname || ' to <sandbox schema name>';
end loop;
end;
```

Note: In the preceding SQL statement, replace the `<sandbox schema name>` placeholder with the sandbox schema.

Enable Bulk Mode

To enable Bulk Mode, update the **SANDBOX_LOAD_MODE** entry in the **NEXTGENEMF_CONFIG** table in the Config Schema to **BULK**. Enabling Bulk Mode updates the queries to load data to the following SQL statement, which improves the data population performance:

```
insert into <sandbox table> select * from <production table>
```

Incremental Data Model Upload

The **Increment Upload Required** column in the *Sandbox Maintenance* window displays the status as "Y" or "N" depending on the need for Incremental Data Model Upload for the Sandbox definition. The Incremental Data Model Upload button is enabled only when the model versions between Production and Sandbox information Domain are different.

Incremental Data Model Upload facilitates you to synchronize the difference of a Data Model which exists in Production and Sandbox Information Domain. You can refresh the details and fetch the incremental data model changes from Production to Sandbox Information Domain.

Note the following points:

- The **Incremental Data Model Upload** button will not be enabled for the sandboxes in which the *No* option is selected for the **Is data model upload required?** field, even though there is a difference between the model versions in Production and Sandbox information domains.

- Incremental data model upload is supported only when the difference in sandbox and production is at table level, that is, the sandbox schema has at least one table less or more when compared to the production schema.

- Incremental data model upload is not supported for Hive based sandboxes.

To perform an incremental data model upload, follow these steps:

1. Select the checkbox adjacent to the Sandbox ID which has the **Increment Upload Required** status as "Y". The status will be Y only when there is a difference in the model versions between Production and Sandbox information Domain.

2. Click ↺ from the *Sandbox Maintenance* tool bar. The *Incremental Data Model Upload* window is displayed with the selected Sandbox definition details in View mode.

3. Click **Refresh**. The incremental data model changes are fetched and updated.

4. An information dialog is displayed confirming successful incremental data model upload.

5. Click **OK**.

Once the Incremental changes are updated to the Data Model in the Sandbox Information Domain, you need to re-save the details in the Sandbox definition window. For more information, see the *Modifying Sandbox Definition* section.

CHAPTER 5   *Managing Techniques*

This chapter provides information about Techniques in the Oracle Financial Services Enterprise Modeling application and step-by-step instructions to use this section.

This section includes the following topics:

- About Techniques
- User Roles and Functions
- Managing Techniques

About Techniques

Technique is a set of generalized statistical algorithms which can be used to build analytical models. The Technique Registration module in Enterprise Modeling displays the techniques developed using R script and ORE functions,.
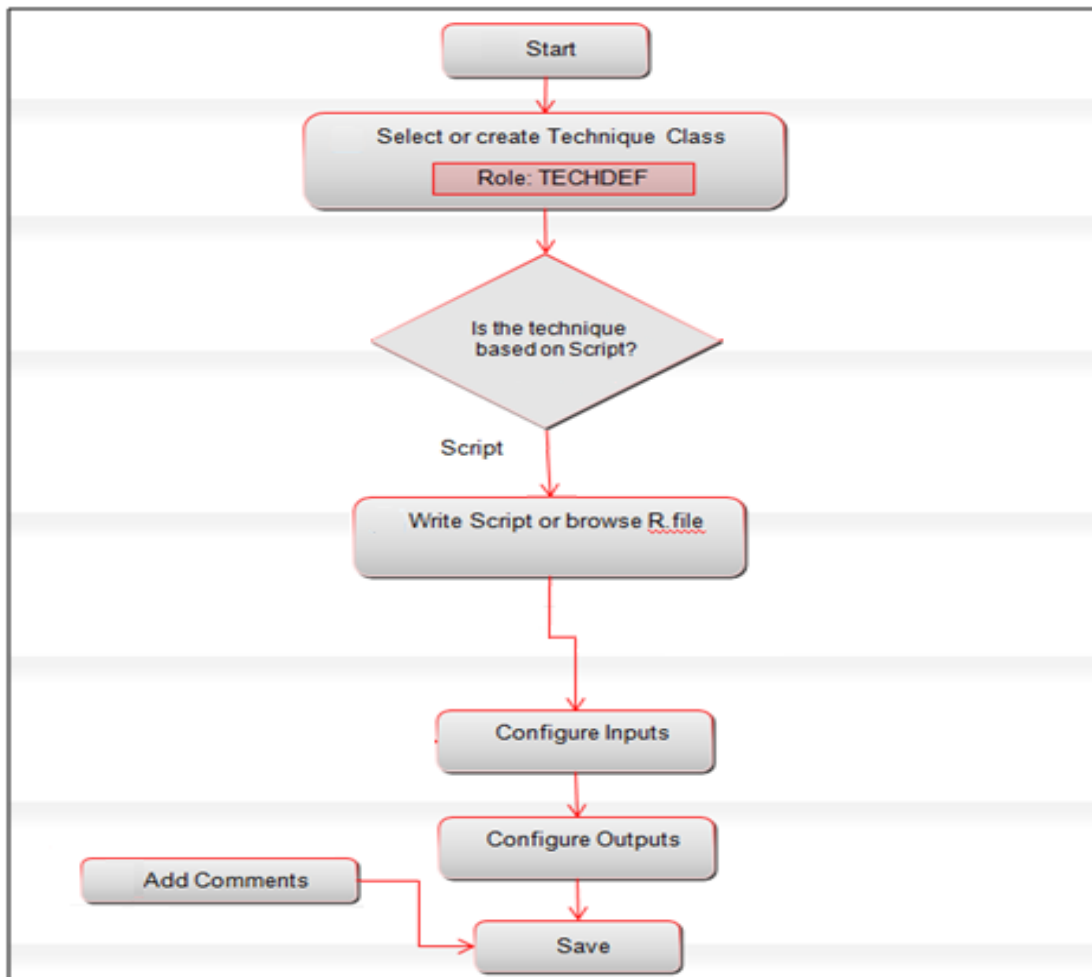
Technique Registration Workflow

Figure 10. Technique Registration Workflow

User Roles and Functions

The following functions should be mapped to the user to manage techniques:

- TECHDEF
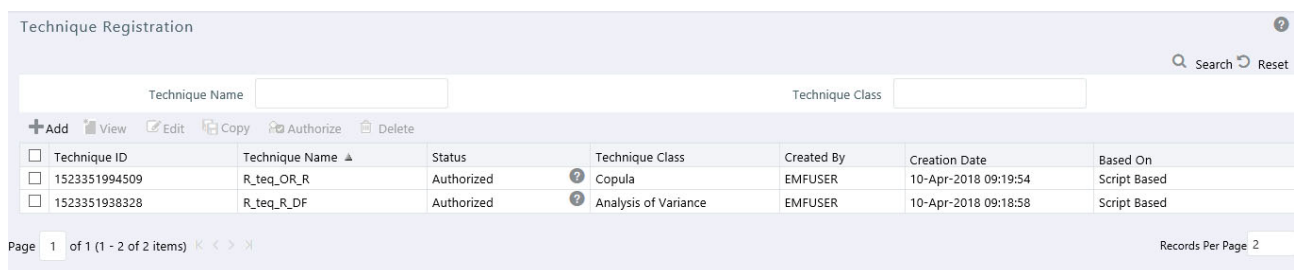- TECHAUTH

Managing Techniques

This section explains how to manage Techniques in Oracle Financial Services Enterprise Modeling application. The following topics are included in this chapter:

- Accessing Techniques
- Creating a Technique Based on Scripts
- Authorizing a Technique
- Viewing a Technique
- Modifying a Technique
- Copying a Technique
- Deleting a Technique
- Searching Techniques

Accessing Techniques

You can access the *Technique Registration* window from the *Oracle Financial Services Enterprise Modeling Home* window. To access the Technique Registration window, follow these steps:

1. Login to Oracle Financial Services Enterprise Modeling.

2. Click ⊞ from the header to display the applications in a Tiles menu.

3. Select **Financial Services Enterprise Modeling** from the Tiles menu. The Enterprise Modeling Application Home Page is displayed with the Navigation list to the left.

1. Click **Financial Services Enterprise Modeling** from the Navigation list to expand view the menu.

2. Click **Technique Registration**. The *Technique Registration* window is displayed.

Figure 11. Technique Registration

Creating a Technique Based on Scripts

Oracle Financial Services Enterprise Modeling application enables you to create techniques based on scripts. The supported scripting languages are R or ORE. After registering the techniques, they can be used in model definitions. In the *Model Script* tab, you can write the script in your preferred language. The Declaration Block in the script is used to declare the input and output parameters. The format of declaration block and the content should not be altered. The scripting should start only from the line after the declaration block.

There are three types of input parameters which can be used in the script, namely Variable, Single Value Parameter, and Model Input. The input parameters can be defined from the Declaration block in the *Script Console* pane, or by right-clicking the parameter and selecting the Input type. The value of single value parameter is defined from the *Inputs* window. However, the value of variables and model inputs are mapped during the model definition in which the technique is used.

You can configure the numeric output by declaring it in the Declaration block (## Output -->), or by right-clicking the parameter and selecting the Output type. It is not required to declare output parameters for plots or graphical outputs.

The script should be a valid script. You can use ORE functions instead of R functions for better performance in R script. All the object names or functions used in the script should be syntactically valid names. Validation of the script is not done while registering the technique. If you already have defined a technique using R script and saved it in a .R file, you can upload it and register it as a technique.

To create a technique using scripts, follow these steps:

1. Click **Technique Registration** from the Navigation list. The *Technique Registration* window is displayed.

2. Click ✚ from the *Technique Registration* toolbar. The *Technique Definition* window is displayed.

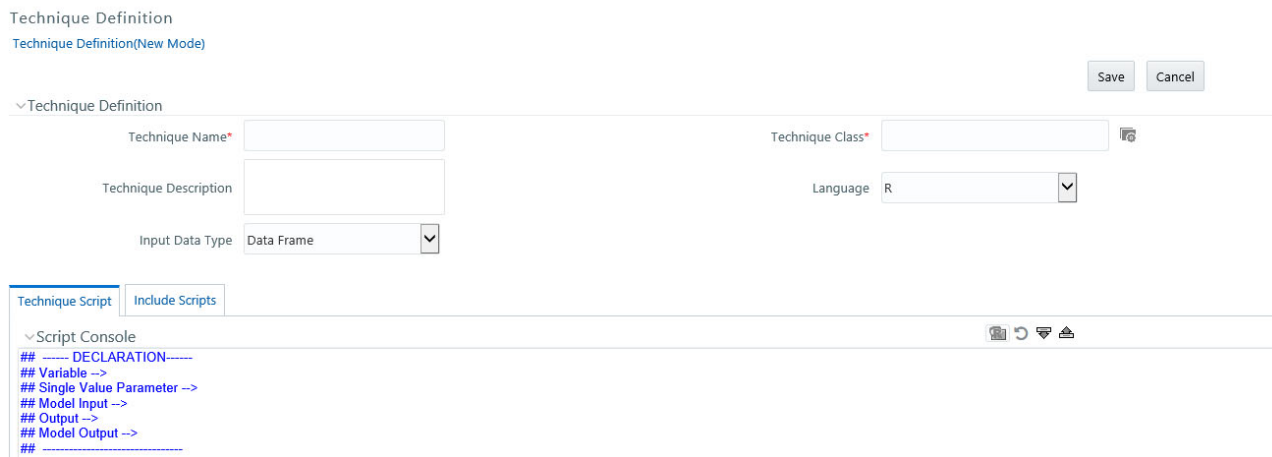**Figure 12. Technique Definition**

3. Enter the details as described in the following table:

Table 4. R script based Technique Definition - Field Description

Field	Description
Fields marked in red asterisk (*) are mandatory.	
Technique Name	Enter a name for the technique. Ensure that there are no special characters like \`, {,},", ', ~, <,>, /, \, and multiple spaces.

Table 4. R script based Technique Definition - Field Description

Field	Description
Technique Class	Technique Class is used for logically grouping the techniques. 1. Click and open the *Technique Class Selection* window. 2. Right-click **Technique Classes** or the technique class name under which you want to create a technique class, and click **Add Technique Class**. Enter the technique class name in the *Add Node* window. 3. Expand the **Technique Classes** tree, select the required technique class and click . 4. Remove a selected technique class by selecting the technique class and clicking .
Technique Description	Enter a brief description about the technique.
Based On	Select the **Script** option.
Language	Select the scripting language. The options are: ● R ● MATLAB
Input Data Type	Select the input data type. For Hive-based Infodoms, the input data types are **Data Frame** and **HDFS file**. For RDBMS based infodoms, the input data typesare **RDBMS Data Frame** and **ORE Frame**.

4. In the *Script Console* pane, enter the R script for the technique. For more information on how to use fitting and prediction, seethe *Fitting and Prediction Using R Script* section.

Figure 13. Script Console

5. By default, the *Script Console* is displayed as shown above. Enter the following details:

- **## Variable -->** Enter the variables you want to use in the R script separated by commas. You should map these variables to the corresponding variables which are available in the Variable Management window, during defining the model.

- **## Single Value Parameter -->** Enter the input parameters you want to use in the R script.

- ## **Model Input-->** Enter the scriptvariable names to which you want to load the output of another model in case of model chaining.

- ## **Output -->** Enter the output parameters you want to use in the R script.

- ## **Model Output-->** Enter the output object names you want to persist.

Note: The variables and the input parameters are case sensitive. The same name cannot be used for both variable and input parameters.

6. Click ![icon] from the *Script Console* toolbar. The *Inputs* window is displayed.

	Name	Type	Display Name	Value
☐	a	Single Value Parameter ▾	Pi	3.14
☐	b	Single Value Parameter ▾	exp	2.71

OK   Cancel

Figure 14. Inputs window

- Enter **Display Name** for the Variables or Model Inputs you defined. The mapping of variables is done during model definition.

- Enter **Display Name** and **Value** for single value parameters you defined.

- Select the parameter and click ![icon] to delete it.

7. Click ![icon] from the *Script Console* toolbar. The *Outputs* window is displayed.

	Output Name	Output Display Name	
☐	op		

OK   Cancel

Figure 15. Outputs window

- Enter **Display Name** for the output parameter you defined.

8. Click ![icon] to browse and upload **.R** file. You can use this option if you have already defined a technique and saved it in an **R** file. Once uploaded successfully, the script is displayed in the script console pane.

9. Select the **Include Scripts** tab if you want to source already defined algorithms to your script.

- Select the required algorithms/ scripts from the *Scripts List* tree and click ![icon] .

■ You can view the scripts by selecting the script and clicking ▦ .

■ After adding scripts, you can position the scripts as required by selecting it and clicking ⌃ and ⌄ .

10. Click the *Comments* tab and click ✚ to add additional information, if required.

11. Click **Save**. If you have authorization privilege, then the technique is auto authorized and it can be used in creating models.

Authorizing a Technique

An authorizer has to authorize a technique before it can be used for model definition. Users who have the function TECHAUTH mapped to their user role can authorize the technique. If the user who created the technique has authorization privilege, then the technique is auto authorized. An authorizer can reject a technique if required.

To authorize or reject a technique, follow these steps:

1. Click **Technique Registration** from the Navigation list. The *Technique Registration* window is displayed.

2. Select the technique and click 🔲 . The *Authorize Technique* window is displayed.

3. Enter a brief description in the **Comments** field. This is mandatory for rejecting the technique.

4. Click **Approve** or **Reject**.

Viewing a Technique

This option allows you to view the details of user defined techniques such as technique name, class, description and the script.

To view the technique details, follow these steps:

1. Click **Technique Registration** from the LHS menu. The *Technique Registration* window is displayed.

2. Select the checkbox adjacent to the Technique ID and click 📄 in the *Technique Registration* tool bar. The *Technique View* window is displayed with all the technique details.

Modifying a Technique

You can update the details of a user defined technique. If the technique is already mapped to a model, you cannot modify it. An alert message is displayed. You cannot modify the Technique Name and Technique Class. For techniques based on external library, you can modify External Library Name, function, and parameter details.

After modification, the technique should be authorized. If you have authorization rights, it will be auto authorized; else an authorizer has to approve it.

To modify a technique, follow these steps:

1. Click **Technique Registration** from the LHS menu. The *Technique Registration* window is displayed.

2. Select the technique you want to modify and click   . This button is disabled if you have selected multiple techniques.

3. Modify the technique details. For more information, refer to *Creating a Technique Based on Scripts* section.

Copying a Technique

You can create a copy of a user defined technique with the required variants. The copied technique should be authorized. An authorizer has to approve or reject the copying of the technique. If you have authorization rights, it will be auto authorized.

To copy a technique, follow these steps:

1. Click **Technique Registration** from LHS menu. The *Technique Registration* window is displayed.

2. Select the technique you want to copy and click   . This button is disabled if you have selected multiple techniques.

3. In the *Technique Definition Copy* window, you can:

- Create a new technique with the existing details. Specify a new Technique Name and click **Save**.

- Create a new technique by updating only the required details. Specify a new Technique Name and update the required details.

For more information, see section *Creating a Technique Based on Scripts*.

Deleting a Technique

You can delete a user defined technique which is no longer required. However, if the technique is already mapped to a model, you cannot delete it.

An authorizer has to approve or reject the deletion of the technique. If you have authorization rights, it will be auto authorized.

To delete a technique, follow these steps:

1. From the LHS menu, click **Technique Registration**. The *Technique Registration* window is displayed.

2. Select the technique you want to delete and click   .

3. Click **OK** in the information dialog to confirm deletion.

Searching Techniques

The *Techniques Registration* window allows you to search for the techniques that you want to view.

To search for Techniques, follow these steps:

1. Navigate to the *Technique Registration* window.

2. Enter the Technique Name and Technique Class you want to search.

3. Click . The filtered list is displayed.

CHAPTER 6   *Managing Variables*

This chapter provides information about variables in the Oracle Financial Services Enterprise Modeling application and step-by-step instructions to use this section.

This chapter includes the following topics:

- About Variable Definition
- Managing Variables

About Variable Definition

Variable refers to a set of attributes that are likely to change based on the selected parameters. In a modeling environment, variables play a vital role in filtering the model parameters and deriving an estimate based on historical data. Variables are defined in the production information domain.

Managing Variables

This section explains how to manage variables in the Oracle Financial Services Enterprise Modeling application. The following topics are included in this section:
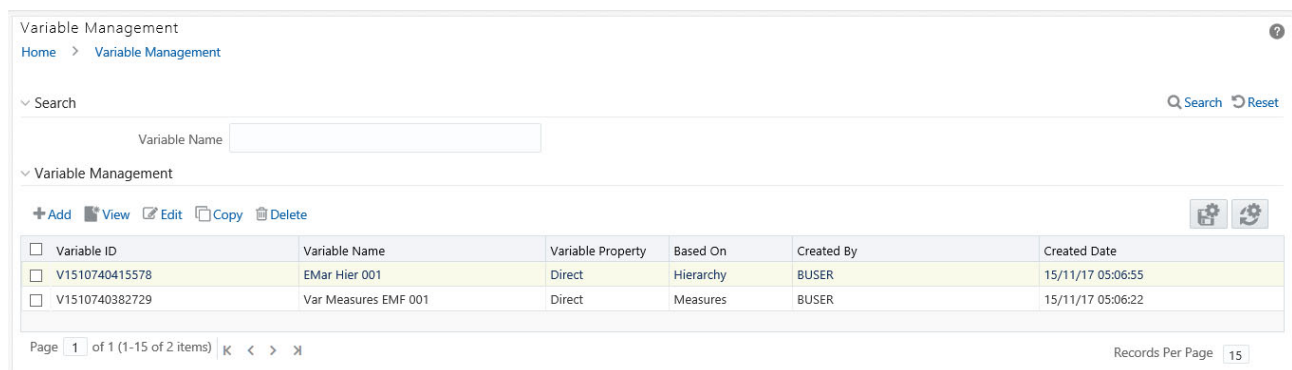
- Accessing Variable Management
- Adding Variable Definition
- Viewing Variable Definition
- Searching Variables
- Modifying Variable Definition
- Copying Variable Definition
- Deleting Variable Definition

Accessing Variable Management

You can access the *Variable Management* window from the Oracle Financial Services Enterprise Modeling Home page.

To access the *Variable Management* window, follow these steps:

1. Login to Oracle Financial Services Enterprise Modeling.

2. Click from the header to display the applications in a Tiles menu.

3. Select **Financial Services Enterprise Modeling** from the Tiles menu. The Enterprise Modeling Application Home Page is displayed with the Navigation list to the left.

4. Click **Financial Services Enterprise Modeling** from the Navigation list to expand view the menu.

5. Click **Data Management** from the expanded Navigation list to view the next level of links in the menu. Click **Datasets and Variables**.

6. Click **Variable Definition**. The *Variable Management* window is displayed.

Figure 16. Variable Management

Saving Column Preferences

You can move the default position of the columns to suit to your viewing preferences. The following is the procedure:

1. Drag and drop a column to a required position in the grid.

2. Click the **Save Preferences** button to save. Click the **Reset** button to revert to the default setting.

Adding Variable Definition

The Add Variable Definition feature enables you to add a variable by filtering through the required parameters. The options available to filter the data vary depending on the variable type selected.

Note: The variables created in the *Variable Definition* window are IDIOSYNCRATIC NUMERIC variables.

To add variable definition, follow these steps:

1. Select ➕ from the *Variable Management* tool bar.

2. The **Add** button is disabled if you have selected any checkbox in the grid. The *Variable Definition* window is displayed.

Figure 17. Variable Definition

3. Enter the Variable details as described in the following table:

Table 5. Variable Definition - Field Descriptions

Field	Description
Fields marked in red asterisk (*) are mandatory.	
Based On	Select any of the following options from the drop-down list, based on which the variable definitions are to be created. • Hierarchy • Measure • Business Processor • Entity
Folder	Select the folder in which you want to create the variable.
Variable Name	Enter a name for the variable. Ensure that there are no special characters like `, {,},", ', ~, <,>, /, \, and multiple spaces.
Variable Description	Enter the required description for the variable.

Table 5. Variable Definition - Field Descriptions

Field	Description
Hierarchy	The following steps are applicable if you are creating a variable based on Hierarchy: 1. By default, this field displays Direct and is read-only If it is enabled for in your application, select **Variable Property** as Direct or Dummy (variable in Binary form and used mainly in regression analysis). 2. Select the required Hierarchy from the **Selected Hierarchy** drop-down list. 3. Click **Save**. **Note:** Hierarchy based variables cannot be created on multi-level Business Intelligence hierarchies.
Measures	The following steps are applicable if you are creating a variable based on Measures: 1. Select the Measure from the **Selected Measures** drop-down list. 2. Create the Variable Transformation. See the description for Variable Transformation in this table for more details. 3. Click **Save**. **Note:** For more details on creating a Business Measure, see *Creating Business Measure* section in *Oracle Financial Services Analytical Applications Infrastructure User Guide.*
Business Processor	The following steps are applicable if you are creating a variable based on Business Processor: 1. Select the Business Processor from the **Selected BPs** drop-down list. 2. Create the Variable Transformation. See the description for Variable Transformation in this table for more details. 3. Click **Save**. **Note:** 1. Variables cannot be created based on Business Processor having placeholders. 2. For more details on creating a Business Processor, see *Adding a Business Processor* section in *Oracle Financial Services Analytical Applications Infrastructure User Guide.*

Table 5. Variable Definition - Field Descriptions

Field	Description
Entity	The following steps are applicable if you are creating a variable based on Entity: If the variable is based on Entity, follow these steps: 1. Click Select Entity to open the *Entity and Attribute* window. The available tables are displayed under the *Available Entities* grid. 2. Select the checkbox corresponding to the table that you want to select. The available columns in the selected table are displayed under the *Available Attributes* grid. 3. Select the required attribute for which you want to create a variable. You can select multiple attributes. 4. Click **Save**. The selected attributes are displayed in the *Selected Attributes* grid. 5. Click **Save** on the Variable Definition window.
Variable Transformation	This feature is available for variables based on Measures, Business Processor, and Entity. 1. Click Transformation to define a transformation for the Variable. The *Technique Browser for Transformation* window is displayed. 2. Select the transformation you want to apply to the variable and click ⟩ . The options are **Arithmetic**, **Exponential**, **Logarithm**, **Power**, and **Trigonometric**. ■ If **Arithmetic** is selected, select the Operator (Addition, Subtraction, Multiplication, and Division) from the drop-down list and enter the **Operator Value**. ■ If **Logarithm** is selected, select the **Base** from the drop-down list. The options are 10, 2, and e. ■ If **Power** is selected, enter the value of Power. ■ If **Trigonometric** is selected, select the **Function** from the drop-down list. The options are Sine, Cosine, and Tangent. ■ If **Exponential** is selected, the exponential value of the variable will be considered for transformation of the variable. Select an applied transformation and click ⟨ to remove its association to the variable. 3. Click **OK**.

4. Click **Save** to upload the new Variable Definition details.

Viewing Variable Definition

You can view individual Variable Definition details at any given point.

To view the existing Variable Definition details, follow these steps:

1. Select the checkbox adjacent to the Variable ID.

2. Click ![icon] in the *Variable Management* tool bar. The *Variable Definition View* window is displayed with all the variable details.

Searching Variables

The *Variable Management* window allows you to search for the variables that you want to view.

To search for Variables, follow these steps:

1. Navigate to the *Variable Management* window.

2. Enter the name of the Variable Name you want to search.

3. Click ![icon] .The filtered list is displayed.

Modifying Variable Definition

You can modify only the Variable Description details if the variable is mapped to a definition. If the variable is not mapped to any definition, then all the filtering parameters can be modified.

To modify an existing Variable Definition, follow these steps:

1. Select the checkbox adjacent to the Variable ID whose details are to be updated.

2. Click ![icon] in the *Variable Management* tool bar. The **Edit** button is disabled if you have selected multiple Variable IDs. The *Variable Definition Edit* window is displayed.

3. Edit the Variable Definition details as required. For more information, see section *Adding Variable Definition*.

4. Click **Save** to upload changes.

Copying Variable Definition

The Copy Variable Definition page enables you to quickly create a new Variable based on the existing variable parameters or by updating the required parameters.

To copy an existing Variable Definition, follow these steps:

1. Select the checkbox adjacent to the Variable ID whose details are to be duplicated.

2. Click ![icon] in the *Variable Management* tool bar. The **Copy** button is disabled if you have selected multiple Variable ID(s).

3. In the *Variable Definition Copy* window, you can:

- Create new variable with the existing details. Specify a new Variable Name and click **Save**.

- Create new variable by updating only the required details. Specify a new Variable Name and update the required details. For more information, see section *Adding Variable Definition*. Click **Save**. The new variable definition details are displayed in the *Variable Management* window.

Deleting Variable Definition

You can remove the variable definition(s) which are no longer required in the system, by deleting from the *Variable Management* window. You cannot delete a variable which is mapped to any definition.

To delete a Variable Definition, follow these steps:

1. Select the checkbox adjacent to a Variable ID.

2. Click 🗑 from the *Variable Management* tool bar. A confirmation dialog is displayed.

3. Click **OK** to confirm deletion.

CHAPTER 7   *Managing Modeling*

This chapter provides information about Modeling in the Oracle Financial Services Enterprise Modeling application and step-by-step instructions to use features in Modeling.

This chapter includes the following topics:

- About Modeling
- Managing Modeling
- Model Chaining
- Model Execution Status
- Model Outputs
- Model Deployment
- Model Execution

- Configure Execution Venue

About Modeling

Modeling refers to the process of designing a prototype based on a structured data model, considering all the variables for statistical analysis and to simulate real events and processes. Modeling module enables you to measure and quantify risk. You can use pre-defined models to predict the business trends and also to validate the existing models.

Models are built based on various techniques and related parameters based on the business purpose. Models thus built use various metadata objects such as Measures, Hierarchies, Filters, Datasets, and so on. The techniques can be inbuilt techniques and user defined script based techniques.

Oracle Financial Services Enterprise Modeling application supports developing statistical models using algorithms written in standard R. It also enables you to bring in already developed algorithms written in R for creating models based on them. Thus it provides a unified environment to build, host and execute models across risk categories and a common framework to manage stress scenarios.
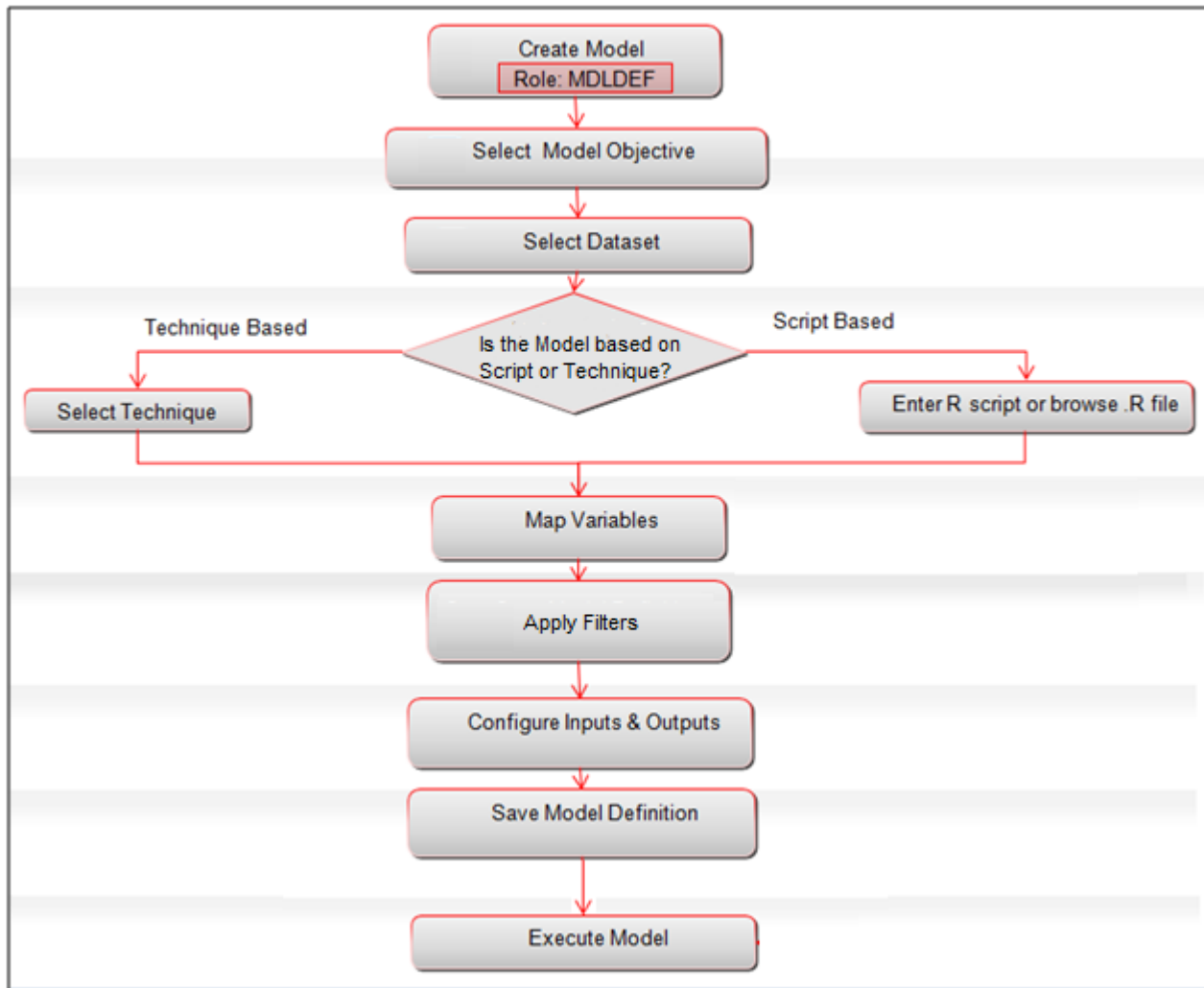
Modeling Workflow

Figure 18. Model Creation Workflow

User Roles and Functions

Functions required to be mapped to the user for managing models are as follows:

- MDLDEF
- MDLAUTH
- MDLEXEC
- MDLOUTPUT
- MDLDEPLOY
- MDLCALIB
- MDLCHAMP

Managing Modeling

This section explains how to manage modeling feature in Oracle Financial Services Enterprise Modeling application. The following topics are included in this section:

- Accessing Model Management
- Model Versioning
- Creating and Executing a Model
- Model Parameters
- Viewing Model Definition
- Modifying Model Definition
- Deleting Model Definition
- Searching Model Management

Accessing Model Management

You can access the *Model Management* window from the *Oracle Financial Services Enterprise Modeling Home* page. To access the *Model Management* window, follow these steps:

1. Login to Oracle Financial Services Enterprise Modeling.

2. Click ![icon] from the header to display the sandboxes in a Tiles menu.

3. Select the required **Sandbox** from the Tiles menu. The Navigation list to the left displays the link for Modeling. Click **Modeling** to expand the list.

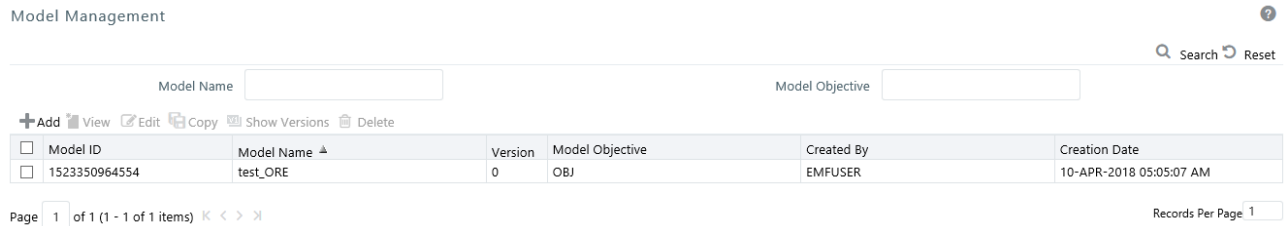4. Click **Model Creation**. The *Model Management* window is displayed.

Figure 19. Model Management

Note: The latest version of the model definition is displayed, when you select a model. You can click ▮ to view the Model Definitions and click ▱ to edit the Model Definition.

Model Versioning

The Model Definitions created in *Oracle Financial Services Enterprise Modeling* are versioned. The version of the Model Definition is displayed in the *Model Management* window. Select a model and click ▦ to view different versions of the selected Model Definition.

When a Model Definition is created, it is saved as version 0. During subsequent modifications, the Model Definition is saved as different versions. For example, the version of a Model Definition is 0, after it is created. The version is updated to 1, after the Model Definition is edited. After editing a Model Definition, you can choose to save a Model Definition as a newer version or overwrite the existing version.

Modeling using Open R with HDFS Option

Oracle R Advanced Analytics for Hadoop (ORAAH)/Oracle R Connector for Hadoop (ORCH) is the default approach for running Open-R on HDFS. For the configurations required for creating and executing models in Open R framework with HDFS option, see *Configurations for Open-R with HDFS* section in OFSAAI Administration Guide available at OHC Documentation Library.

Basic Functions

Use the following basic functions to get basic OFSAA Metadata details, basic Model details and basic Run details:

     ofs_get_log_handle()

     ofs_get_model_id()

     ofs_get_model_version()

     ofs_get_model_name()

     ofs_get_batch_run_id()

     ofs_get_task_id()

     ofs_get_infodom()

     ofs_get_dataset_codes()

     ofs_get_dataset_names()

     ofs_get_dataset_queries()

     ofs_get_output_names()

     ofs_get_meta_data_info()

      ofs_get_hive_schema_name()

Examples:

model_id   = OFSAAIRunnerHDFS::ofs_get_model_id();

batch_run_id = OFSAAIRunnerHDFS::ofs_get_batch_run_id();

meta_data_list = OFSAAIRunnerHDFS::ofs_get_meta_data_info();

      ofs_get_meta_data_info() is a special function for getting dataset and variable and its mapping info. This function returns a list for each of the dataset.

      Sample list for a single dataset looks like as given:

          variable.names - contains all the script variables mapped for the dataset.

          variable.index - contains the index of each selected variable in the dataset query.

          variable.entity.attributes - contains actual base entity and attributes for the selected mata-data variable.

          dataset.base.entities - contains all the base entities part of the dataset.

          dataset.query - contains dataset query for fetching data.

Note: All these attributes are available as vectors within the list.

      print(meta_data_list);

      $`Dataset_Name`

      $`Dataset_Name`$variable.names

      [1] "x" "y" "z"

      $`Dataset_Name`$variable.index

      [1] "1"   "2:3" "4"

      $`Dataset_Name`$variable.entity.attributes

      [1] "DATASET_TABLE_1.DATASET_COLUMN1" "DATASET_TABLE_2.DATASET_COLUMN2" "DATASET_TABLE_2.DATASET_COLUMN3" "DATASET_TABLE_3.DATASET_COLUMN4"

      $`Dataset_Name`$dataset.base.entities

      [1] "DATASET_TABLE_1" "DATASET_TABLE_2" "DATASET_TABLE_3"

      $`Dataset_Name`$dataset.query

      [1] "Dataset Query"

Note: For multiple datasets, preceding lists get appended.

ofs_get_log_handle()   : returns, handle to an existing log handle.

ofs_get_model_id()   : returns, ID of the Model, as character/string.

ofs_get_model_version()   : returns, Version of the Model, as numeric.

ofs_get_model_name()   : returns, Name of the Model, as character/string.

ofs_get_batch_run_id() : returns, Batch Run Id of the Current Batch, in which model is added for the execution, as character/string.

ofs_get_task_id() : returns, Task Id of the Model in current Batch, character/string

ofs_get_infodom()   : returns, OFSAA information Domain (Infodom) Name, as character/string.

ofs_get_dataset_codes() : returns, Dataset-Codes for all the selected datasets, as character vector.

ofs_get_dataset_names()   : returns, Dataset-Names for all the selected datasets, as character vector.

ofs_get_dataset_queries() : returns, Dataset-Queries for all the selected datasets, as character vector.

ofs_get_output_names()   : returns, selected Output names, as character vector.

ofs_get_hive_schema_name() : returns, Hive schema name, as character/string.

ORAAH Based Functions for Spark/ORE

Following framework functions will help in connecting/disconnecting Spark/ORE(Hive). Internally these functions will use ORAAH for Spark/ORE connections. These functions will return TRUE on success and FALSE in case of any failures with the proper error message in the logger.

> ofs_ORAAH_spark_connect()
>
> ofs_ORAAH_spark_disconnect()
>
> ofs_ORAAH_hive_connect()
>
> ofs_ORAAH_hive_disconnect()

All the 4 functions returns -

> TRUE : on Success
>
> FALSE : on Failure

Note: ofs_ORAAH_spark_connect/ofs_ORAAH_spark_disconnect should be called first/last , before using any ORCH-Spark related function, like orch.glm2().
ofs_ORAAH_hive_connect/ofs_ORAAH_hive_disconnect should called first/last, before using any ORCH-ORE related functions.

RDBMS Connect Functions

ofs_rdbms_connect(arg.ofsaa.schema.name = c("CONFIG", "ATOMIC"));

ofs_rdbms_disconnect(arg.ofsaa.schema.name = c("CONFIG", "ATOMIC", "BOTH"))

These functions should be used to get RDBMS connections/disconnections. OFSAA schema type should be passed as an argument. By default it connects to Configuration Schema.

ofs_rdbms_connect(arg.ofsaa.schema.name = c("CONFIG", "ATOMIC"));

returns -

> valid connection handle object in case of success.
>
> NULL in case of failure with proper error message in logger.

ofs_rdbms_disconnect(arg.ofsaa.schema.name = c("CONFIG", "ATOMIC", "BOTH"))

returns -

> TRUE : on Success
>
> FALSE : on Failure

Example:

atomic_conn_handle = OFSAAIRunnerHDFS::ofs_rdbms_connect(); #by default configuration schema.

atomic_conn_handle = OFSAAIRunnerHDFS::ofs_rdbms_connect(arg.ofsaa.schema.name = "ATOMIC");

disconnect_status = OFSAAIRunnerHDFS::ofs_rdbms_disconnect(arg.ofsaa.schema.name "BOTH");

Usage

All these functions are available as part of OFSAAIRunnerHDFS R package. This package is available under `$FIC_DB_HOME/lib` as part of installation. This package needs to be installed on a machine which is running Rserve or client R Engine. Package reference should be used in order to avoid conflict with similar function name.

Example: OFSAAIRunnerHDFS::ofs_get_model_id();

OFSAAIRunnerHDFS::ofs_ORAAH_spark_connect();

Creating and Executing a Model

The Modeling module enables you to create and execute data models. This section has the following topics:

- Creating a Model Using R Scripted Technique

- Creating a Model using Script

Note:

1. You can create models only in Sandbox Information Domain.

2. Modeling supports derived entities. However, materialized view is not created in the Sandbox schema during Sandbox creation and you have to do it manually. Get the materialized view definition details from the production schema and create the materialized view in the Sandbox.

Creating a Model Using R Scripted Technique

Note: You can create models using a technique which you have defined from the Technique Registration window. Execution Status Log is not displayed in *Model Definition* window for models created using Standard R Engine.

To create a model using R scripted technique, follow these steps:

1. Select ![+] from the *Model Management* tool bar. The *Model Definition* window is displayed. The **Add** button is disabled if you have selected any Model ID in the grid.

Figure 20. Model Definition page

2. Enter the Model Definition Details. The common fields are described in Table 6.

   The grid below the Model Details section displays the various tabs available for the selected technique. To update the required information, see the following sections:

   - Inputs
   - Filter
   - Outputs

Table 6. Model Definition - Field Descriptions

| Field	Description
Fields marked in red asterisk (*) are mandatory.	
Model Name	Specify a model name for the model definition. Model Name is case sensitive and does not allow duplication. For example, model name "Linear Regression" is not allowed if a model with the name "linear regression" exists. Ensure that there are no special characters like `` ` ``, {,},", ', ~, <,>, /, \, and multiple spaces.
Model Description	Enter a description for the model.
Do you like to script the model?	Select the checkbox to script the model in *Model Script* pane.
Model Objective	Select the Model Objective from the drop-down list. You can also click ➕ to create a Model Objective.
Technique	This field is disabled if you have selected to script the model. 1. Click 🗔 and open the *Technique Selection* window. The pre-packaged techniques, and user defined (registered and authorized) R techniques are listed in the *Techniques* pane. 2. Click ➕ and expand the technique heading groups. 3. Select the required technique and click ⟩ . 4. Click **OK**. The selected Technique details are displayed in the *Model Definition* window. If you have selected **R** technique, click ▥ to view the script.

Table 6. Model Definition - Field Descriptions

Field	Description

Dataset	By default, the dataset of the Sandbox is displayed. You can change the dataset if necessary.
	Dataset selection is mandatory:
	• For models based on R scripted techniques if variables are declared in the R script.
	
	1. Click [icon] to open the *Dataset Selection* window. The available datasets are listed in the *Datasets* pane.
	
	  ■ Select a dataset and click [icon] to view the details of the selected datasets.
	
	  ■ Click [icon] to create a new dataset. For more details on creating a dataset, refer *Creating Data Set* section in Oracle Financial Services Analytical Applications Infrastructure User Guide.
	   You can create dataset using any of the tables which is part of the production information domain. But if you create a dataset with a table which is not part of the Sanbox and create a model using that dataset, then deploy the model to production infodom and execute it there.
	
	2. Select the required Dataset based on which the model is to be created and click [icon]. Ensure the selected dataset is loaded with data, otherwise model execution will fail. You can select multiple datasets for models executed using Standard R Engine. If multiple datasets are used, you should use atleast a variable from each dataset.
	
	3. Click **OK**.
	
	Note: The Datasets based on Derived Entities are not supported.

Language	This field is not displayed for techniques based models. Select the scripting language from the drop-down list. The options are: • R • Input Data Type
Type	This field is not displayed for techniques based models. Select the type of engine from the drop-down list. The options are: • Standard R Engine • ORE Engine- This option is not displayed in Hive based Infodoms.
Input Data Type	This field is displayed only in Hive based infodoms for models based on R scripted techniques or if you select to script the model. Select the input data type. The options are **ORE Frame**, **Data Frame** and **HDFS File**.

4. Click **Save** to save the model definition details, after all the necessary details are updated.

5. Click **Preview Data** to view the data of the selected dataset. It displays the primary keys and the attributes/ columns of the tables in the selected dataset. Only those columns which are mapped to the variables in the script are displayed.

Note: In case of Hive based Sandbox information domain, previewing data takes long time and only 100 records are displayed.

6. Click **Execute**.

The Execution Status grid displays the model execution log dynamically.

Note: For R based models, the execution may fail if the dataset contains internal joins. Executing a model using standard R engine with the new cloudera jars is failing with model queries exceeding a certain limit. Workaround is to append UseNativeQuery=1 in the JDBC URL of the Hive schemas in which model definitions and executions happen. For example,
jdbc:hive2:10.123.456.78:1000/default;useNativeQuery=1

Creating a Model using Script

This feature allows you to create models based on user written scripts. The supported scripting languages are Standard R and ORE. The script should be a valid script.

In case of R script, all the object names used in the R-script should be syntactically valid names. You should select Language as ORE if you are using ORE functions in the script. ORE enables transparent access to data stored in the Database for scalability and high performance. For more information, see *Enterprise Modeling support for R Scripts and Oracle R Enterprise (ORE) Statistical Functions*.

Note: The models defined based on R scripting can be directly deployed to the production infodom without execution in sandbox.

To create a model using script, follow these steps:

1. Select ✚ from the *Model Management* tool bar. The *Model Definition* window is displayed.

Figure 21. Model Definition

2. Enter the model definition details. The common fields are described in Table 6.

3. For R scripted model, click to browse and upload **.R** file. You can use this option if you have already defined a technique and saved it in an .**R** file. Once uploaded successfully, the script is displayed in the *Model Script* pane.

Or

In the Model Script pane, enter the script for the model. For information on how to use fitting and prediction, see *Fitting and Prediction Using R Script*.
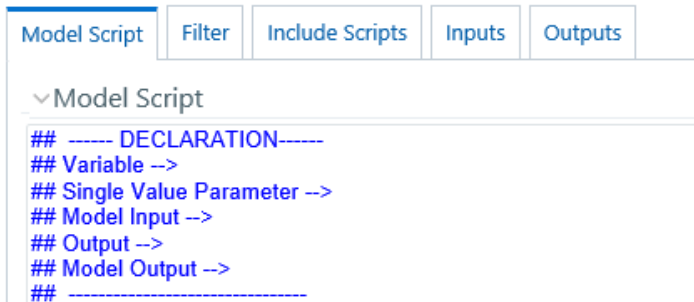

Figure 22. Scripting Console

> **Note:** The format of declaration and the content should not be altered. The scripting should start only from the line after the declaration block.

By default, the Model Script is displayed as shown above.

- ## **Variable -->** Enter the variables you want to use in the script separated by commas.

- ## **Single Value Parameter -->** Enter the input parameters you want to use in the script.

- ## **Model Input -->** Enter the script variable names to which you want to load the output of another model in case of model chaining.

- ## **Output -->** Enter the output parameters you want to use in the script.

- ## **Model Output-->** Enter the object names you want to persist.

> **Note:** The variables and the input parameters are case sensitive. Same name cannot be used for both variables and input parameters.
> You should explicitly define the outputs in the script that need to be updated back to tables. If the output is not defined, then the tables are not updated.
> For example :
> x = rbeta(100,1,2)
> y = rbeta(100,1,2)
> op = lm(y~x)
>
> If op\$residuals needs to be updated then declare the output with a name.
> For Example:
> Residuals = op\$residuals

The grid below the *Model Details* section displays the various tabs available. To update the required information, see the following sections:

- Inputs

- Filter

- Include Scripts

- Outputs

4. Enter valid script of the model in the selected scripting language. For more information on data usage and data handling when R script and ORE functions are used, see R *Scripts and Oracle R Enterprise (ORE) Statistical Functions*.

5. Click **Save** to save the model definition details, after all the necessary details are updated.

6. Click **Preview Data** to view the data of the selected dataset. It displays the primary keys and the attributes/ columns of the tables in the selected dataset. Only those columns which are mapped to the variables in the script are displayed.

Note: In case of Hive based Sandbox information domain, previewing data takes long time and only 100 records are displayed.

7. Click **Execute** to execute the model.

   You can view the status of the model execution dynamically in the Execution Status grid.

   Note the following points about Model execution:

   ■   For R based models, the execution may fail if the dataset contains internal joins.

   ■   Executing a model using standard R engine with the new cloudera jars is failing with model queries exceeding a certain limit. Workaround is to append UseNativeQuery=1 in the JDBC URL of the Hive schemas in which model definitions and executions happen. For example:

   ```
   jdbc:hive2:10.123.456.78:1000/default;useNativeQuery=1
   ```

   ■   Models executed using Standard R engine will fail if multiple primary nodes are set in the ModelingFramework.XML

   ■   Users who are using ORE1.4.1 and R 3.1.1 may have to explicitly set session timezone in scripts registered within OFS EM by using the 'Sys.env(TZ=<time zone>)' R function. Alternatively you may set the same call in 'R_HOME/etc/Rprofile.site' file on the database server, where R_HOME is the home directory of the R instance on which ORE server packages are installed.

8. Click **Output** to view the output. For more information, see the View Model Outputs section.

Model Parameters

The grid in the Model Details section displays the various parameters applicable when a technique is selected. It is mandatory to update the required information in the tabs displayed and the parameters or fields in the available tabs vary depending on the technique selected. Few of the common input parameter types are explained below.

Filter

In the *Filter* tab, you can add multiple non-time hierarchy members as filters.

1. In the *Model Definition* window, click ➕ from the *Filters* toolbar. The *Filter Browser* window is displayed.

2. Select Data Filter or Hierarchy to display the corresponding filters.

3. Select the required filter or hierarchy from the list and click ❯ . The selected filter is added to the *Selected Members* pane.

   Note: If a Hierarchy is created on a table with the same column holding hierarchy Code and Description, then the hierarchy nodes will not be displayed in the *Hierarchy Browser* window on Hive connections. In such scenario, use the concat functionality in the **Description** field like concat(<TableName.ColumnName>). For example, concat(credit.age,").

4. Click **OK**. The selected Filters are displayed in the grid.

5. Filters can be applied to both Production and Sandbox Information Domains. Select the required option **Apply in Production** or **Apply in Sandbox** to apply the filter in the required information domain.

Note: Data Element Filters created in the Filters screen is displayed when you select **Data Filter**.

Inputs

The following sections describe the various inputs for:

- For Script based Models and For R scripted Technique

For Script based Models and For R scripted Technique

The variables and model inputs you have declared in the *Model Script* pane are displayed in the *Inputs* tab.

1. From the *Variables* grid, map the configured script variables to the available variables in the system.

   - Select a variable from the *Available Variables* pane by expanding the required variable type and a variable in the *Configured Script Variables* pane, and click **Map**.

   - You can add a new variable and then use that variable for mapping.

     ◆ Click ➕ to add new variables. The *Variable Defintion* page is displayed. You can create a new variable from the *Model Definition* page, as per the requirement. For more information, see Adding Variable Definition section.

   The mapping details are displayed in the *Variable-Mapping* grid. To unmap, select the variable from the *Configured Script Variables* pane and click **Unmap**.

   Note:

■ You cannot select a variable in the *Available Variables* pane which is already mapped to another variable in the script. That is, an OFSAA variable can be mapped to only one input variable in the model script. If you want to use the same OFSAA variable for mapping to multiple model input variables, create a copy of the variable and assign it to required model input variables.

■ Mapping of single variable declared in the R script to multiple variables is not supported if ORE implementation is used.

2. From the *Model Inputs* grid, map the output of the model which you want to map to the declared Model Input.

■ Select a Model Output from the *Available Model Outputs* pane by expanding the required model and a Model Input from the *Model Inputs* pane, and click **Map**.

The mapping details are displayed in the *Configured Model Inputs* grid. To unmap, select the Model Input from the *Model Inputs* pane and click **Unmap**.

3. From the *Input Parameters* grid, select the **Evaluation Type** from the drop-down list. You can execute model on complete set of records in the dataset, on a group of records or on a set of rows.

● **Group**- Select this option if the model needs to be executed on a group, which is defined using a grouping variable. Select the Grouping Variable from the drop-down list.

● **Row**- Select this option if the model needs to be executed on a set of rows. Specify the Number of rows.

Note: For Evaluation Type as Row, graphical output (plot) is not supported and the number of records processed is five times the value given in the Number of Rows field. These are limitations.

● **NA**- Select this option if the model needs to be executed on complete set of records in the dataset.

4. All single value parameters declared in the technique are displayed. Displays the Value if it was given while defining the technique. Else enter the appropriate value.

Include Scripts

This tab is used to source already defined algorithms to your script.

1. Select the required algorithms/ scripts from the *Scripts List* tree and click ⟩ .

2. You can view the scripts by selecting the script and clicking ⊞ .

3. After adding scripts, you can position the scripts as required by selecting it and clicking ⌃ and ⌄ .

Outputs

The following sections describe the various Output options for:

● For R Scripted Technique and For Script Based Models

For R Scripted Technique and For Script Based Models

The *Outputs* tab allows you to store the output values computed during the model execution to a table in your atomic schema by mapping the output value to a variable, which you have defined through the Variables Definition window. This is supported only for record level outputs. Additionally, you can create a new table to store the output if it is a dataframe, vector or matrix.

Note: You can view the *Outputs* tab only after declaring Output variables (## Output -->) in the model script.
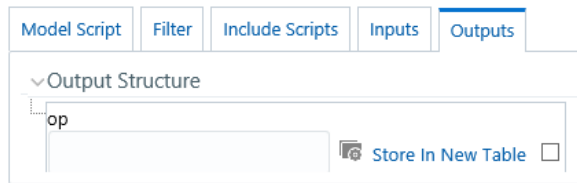

Figure 23. Outputs window

1. From the *Outputs* tab, click [icon] corresponding to the declared output variable whose value you want to map to a variable. The *Variable Browser* window is displayed.

2. Select the appropriate variable and click **Ok**.

3. Select the checkbox corresponding to **Store in New Table** if you want to create a new table to store the output.

   New table name is <<Outputname>>_<<ModelID>>_<<Version Number>>. The combined length should not be more than 30. So ensure the Output name does not have any special characters.

Note the following limitations:

- Creating new tables for storing the model outputs is supported only for scripted models, executed using ORE engine. New table creation may not work for Evaluation type selected as Group and Row.

Note: For models executed using ORE engine, you can map only row level outputs to variables.

Scenarios

Consider the following script where outputs are not configured. Then the framework output will be just the graphical output. That is, when the outputs are not chosen by the user, then the framework reports the last line's output as the only default output.

Note that the plots are always produced.

```
##   ---------DECLARATION------------
## Variable --> DepVar,IndepVar1,IndepVar2
## Single Value Parameter -->
## Output -->
##   ------------------------------
art.mod<-lm(DepVar ~ IndepVar1 + IndepVar2)

art.summ<-summary(art.mod)
coef( art.summ )
art.summ[[ "r.squared" ]]

#do some line plots
#plot()

#form new data.frame object
new.x.datafrme = data.frame(x=seq(from=range(IndependentVariable1)[ 1 ],to=range(
IndependentVariable1)[2],length=length(DependentVariable)))

#form new regression model
RegModel<-lm(DepVar ~ new.x.datafrme$x + IndepVar2)
```

```
#perform some operations on the new data
Plot(RegModel)
```

In this case, the output window will have a tree containing the components of 'RegModel' object with the same name as the root node.

Alternatively, you can also put the intended objects name at the last line of the script as:

```
##   ---------DECLARATION------------
## Variable -->
## Single Value Parameter -->
## Output --> RegModel
##   -------------------------------
art.mod<-lm(DepVar ~ IndepVar1 + IndepVar2)

art.summ<-summary(art.mod)
coef( art.summ )
art.summ[[ "r.squared" ]]

#do some line plots
#plot()

#form new data.frame object
new.x.datafrme = data.frame(x=seq(from=range(IndependentVariable1)[ 1 ],to=range(
IndependentVariable1)[2],length=length(DependentVariable)))

#form new regression model
RegModel<-lm(DepVar ~ new.x.datafrme$x +
IndepVar2)

#perform some operations on the new data
Plot(RegModel)
RegModel
```

Here the output window will show a tree containing the components of 'RegModel' object, but since the output name was not configured or specified, it will be held under a generic framework name 'OFSAAOutput'

One of the added advantages of configuring the outputs is that one can get more than one object as output as shown in the following script:

```
##   ---------DECLARATION------------
## Variable -->
## Single Value Parameter -->
## Output --> RegModel, art.mod
##   -------------------------------
art.mod<-lm(DepVar ~ IndepVar1 + IndepVar2)

art.summ<-summary(art.mod)
coef( art.summ )
art.summ[[ "r.squared" ]]
```

```
#do some line plots
#plot()

#form new data.frame object
new.x.datafrme = data.frame(x=seq(from=range(IndependentVariable1)[ 1 ],to=range(
IndependentVariable1)[2],length=length(DependentVariable)))

#form new regression model
RegModel<-lm(DepVar ~ new.x.datafrme$x +
IndepVar2)

#perform some operations on the new data
Plot(RegModel)
```

For the above script, the output window will contain trees for both 'RegModel' and 'art.mod'.

Here are the possible scenarios and the respective framework behavior:

- **Case A**: When no outputs are specified in the declaration block and outputs are not configured in the Outputs window:
The value of the last evaluated expression is returned with the name 'OFSAAOutput' for the model execution.

- **Case B:** When no outputs are specified in the declaration block and the Configure Outputs button is clicked:
All the objects from the script execution environment get listed as a tree with each object as separate nodes. Now you can choose the outputs to be reported. The selected ones will be placed in the declaration block. If you do not select any object as output, then the behavior will be the same as case A, for the model execution.

- **Case C:** Some objects are declared as outputs in the declaration block either by explicitly writing the correct names in the declaration block against the label Outputs, or by right-clicking a valid object name and selecting it as output:
Now, only those which are listed in the declaration block will come in the Configure Outputs window (as checked ones, since you have already selected them as the outputs explicitly).

Viewing Model Definition

You can view individual Model Definition details at any given point. To view the existing Model Definition details in the Model Definition window, follow these steps:

1. Select the checkbox adjacent to the Model ID.

2. Click ▉ in the Model Management toolbar.
An information dialog is displayed indicating that view mode is selected and any accidental changes will not be updated. Click **OK** to confirm and view the details.

3. In the Model Definition View window you can:

   Note: Select **Preview Data** to view the variable parameters corresponding to the technique selected.In case of Hive based Sandbox information domain, previewing data takes long time and only 100 records are displayed.

- Select **Execute** to process the model execution. The status of which can be verified in Model Execution Status option, accessing *Oracle Financial Services Enterprise Modeling > Modeling > Model > Model Execution Status*.

Modifying Model Definition

You can update the model definition details of an existing Model in the Model Definition window:

Note: Modifying ORE models is not supported. Additionally, applying hierarchy filters for R scripted models in edit mode is not supported.

To modify model definition

1. Select the checkbox adjacent to the Model ID whose details are to be updated.

2. Click [icon] in the Model Management tool bar. The Edit button is disabled if you have selected multiple Model IDs.

3. Edit the Model Definition details as required. Model Name, Technique, and Model Objective are not editable. You can update the Model Description, Dataset, and the variable parameters based on the technique selected. For more information, see *Creating and Executing a Model.*

4. Once you have updated all the necessary details in the Model Definition Edit window, you can:

- Click **Preview Data** to view the new Model Definition details before upload.

  Note: In case of Hive based Sandbox information domain, previewing data takes long time and only 100 records are displayed.

- Click **Save** to update the model definition details. You can choose save a Model Definition as a new version or overwrite the existing version. For more details, see *Model Versioning* section.

- Click **Save** and select **Execute** to process the model execution. The status of which can be verified in Model Execution Status option, accessing *Oracle Financial Services Enterprise Modeling > Modeling > Model > Model Execution Status.*

Deleting Model Definition

You can remove the model definitions which are no longer required by deleting from the Model Management window.

1. Select the checkbox adjacent to the Model ID(s) whose details are to be removed.

2. Click [icon] in the Model Management tool bar.

3. Click **OK** in the information dialog to confirm deletion.

Searching Model Management

The Model Management page allows you to search the models that you want to view.

To search for models, follow these steps:

1. Navigate to the Model Management page.

2. Enter the Model Name and Model Objective.

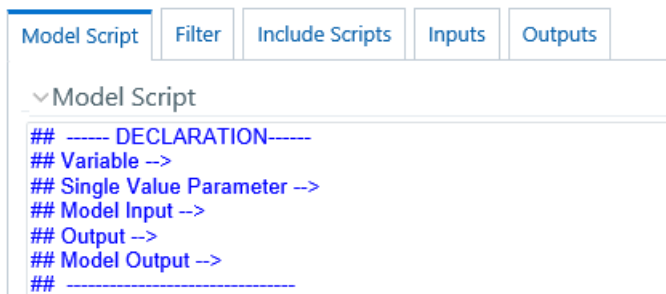3. Click [icon] . The filtered list is displayed.

Model Chaining

Model Chaining is used to input an object/objects from one model to another model. It can also be used for separation of calibration and prediction. The objects persisted from any model can be chained to any other model. In case of ORE intermediate objects are saved in ORE data-store whereas in Standard-R / Open-R, objects are saved or persisted as a file.

The following tags are available in the Model Script Console:

- **##Model Input-->** Enter the object names to which you want to map object from some other model.
- **##Model Output-->** Enter the object names you want to persist.

Note: Object persisted out of an ORE based model cannot be chained /inputted to Open-R / Standard-R based model and vice-versa.

Figure 24. Model Script Console

Model Execution Status

Model Execution Status in the Sandbox Information Domain enables you to verify the status of the executed model at any given point. The status can be Complete, Ongoing, or Failed based on the current processed stage of the system.

This section includes the following topics:

- Accessing Model Execution Status
- Searching Model Execution Status

Accessing Model Execution Status

To access the Model Execution Status page, follow these steps:

1. Click **Sandbox** tab.

2. Click **Model Execution Status**. The Model Execution Status page is displayed.

Note: This menu is displayed only if you are in sandbox infodom.

Figure 25. Model Execution Status

Searching Model Execution Status

The Model Execution Status page allows you to search the models that you want to view.

To search for Model Execution, follow these steps:

1. Navigate to the Model Execution Status page.

2. Enter the Model Name.

3. Click   . The filtered list is displayed.

Model Outputs

The Model Outputs page in Sandbox Information Domain displays the outputs of all models executed in the selected Sandbox Information Domain. The Model Outputs page in Production Information Domain displays the outputs for the models deployed in the production Information Domain and also those requested for execution. You can also view or download the derived output values as a .csv file and plots as .png files for reference.

Limitation: You cannot download the CSV file for ORE Model Output if data redaction is enabled.

This section includes the following topics:

- Accessing Model Outputs
- Searching Model Outputs
- View and Compare Model Outputs
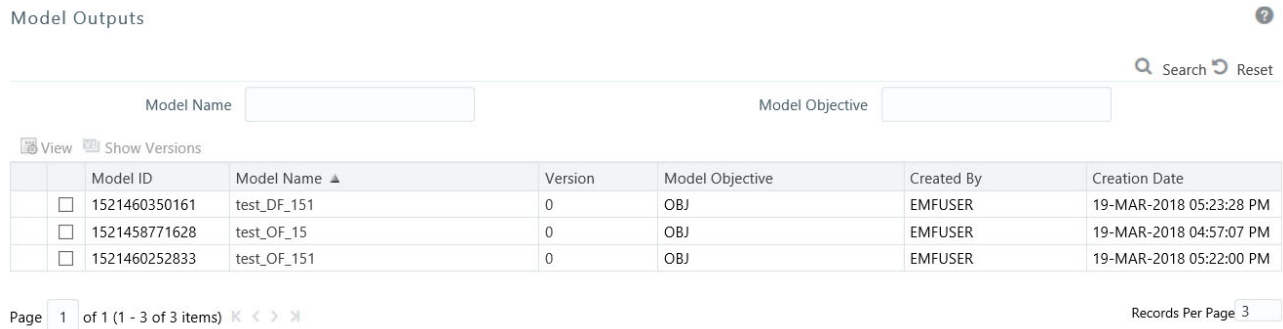
Accessing Model Outputs

You can access the Model Outputs page from the Sandbox.

To access the Model Outputs page from Sandbox, follow these steps:

1. Login to Oracle Financial Services Enterprise Modeling.

2. Click [image] from the header to display the sandboxes in a Tiles menu.

3. Select the required **Sandbox** from the Tiles menu. The Navigation list to the left displays the link for Modeling. Click **Modeling** to expand the list.

4. Click **Model Output**. The *Model Outputs* page is displayed.

Figure 26. Model Outputs

Searching Model Outputs

The Model Outputs page allows you to search the models that you want to view.
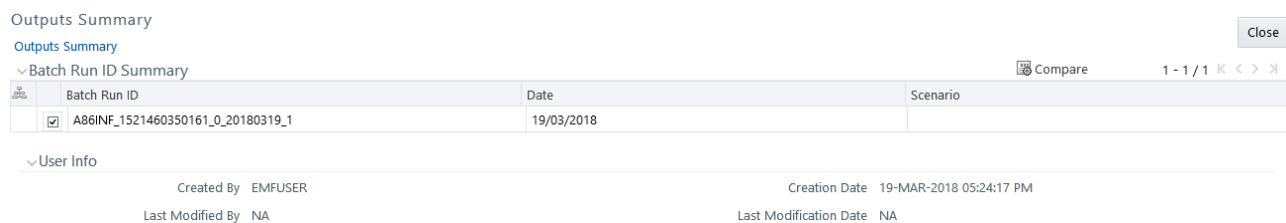
To search for Model Outputs, follow these steps:

1. Navigate to the Model Outputs page.

2. Enter the Model Name and Model Objective.

3. Click [image]. The filtered list is displayed.

View and Compare Model Outputs

To view and compare the model outputs, follow these steps:

1. Select the checkbox adjacent to the Model ID.

Click [image] from the *Model Outputs* toolbar. The *Output Summary* window is displayed.

Figure 27. Outputs Summary

2. Select the checkbox adjacent to a Batch Run ID to view the details or select multiple checkboxes to compare Batch Run IDs.

3. Click ![icon] to view details or compare Batch Run ID(s). The *View/Compare Sample* window is displayed with the comparison details of the selected models.

4. Click the links to view/download the derived values stored as .csv files or plots saved as image files.

Figure 28. Output window for R scripted models

5. Click the file name (.dat file) under the *Summary Output* pane to view the summary of outputs. This is displayed only for models based on Standard R Engine.

6. Click the links in the Outputs pane to download or view the derived values in .csv files.

Model Deployment

Model Deployment in the Sandbox Information Domain enables you to request for deployment of a model to the production information domain. Once you define a model and test it in the Sandbox, and if the testing (checking the results of model execution) results in concluding that the definition is sound and satisfactory, promote the model to the production infodom to execute the model on the production data.

Since models based on R scripting can be directly deployed without execution, those models will be displayed in the Model Deployment window.

This section includes the following topics:
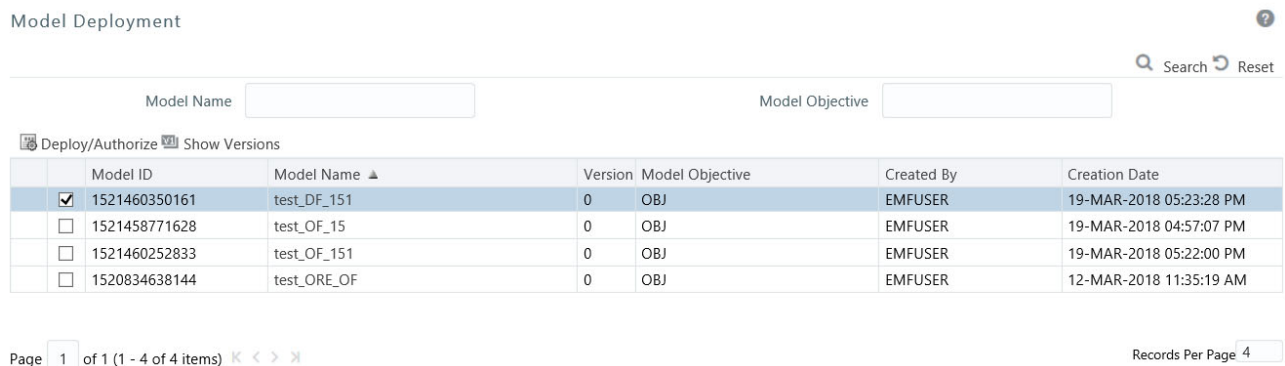
- Accessing Model Deployment
- Searching Model Deployment
- Requesting and Authorizing Model Deployment

Accessing Model Deployment

To access the Model Deployment page, follow these steps:

1. Click ![icon] from the header to display the sandboxes in a Tiles menu.

2. Select the required **Sandbox** from the Tiles menu. The Navigation list to the left displays the link for Modeling. Click **Modeling** to expand the list.

3. Click **Model Deployment** to display the *Model Deployment* window.

Note: This menu is displayed only if you are in sandbox infodom.

Figure 29. Model Deployment Page

Searching Model Deployment

The Model Deployment page allows you to search the models that you want to view.

To search for Model Deployment, follow these steps:

1. Navigate to the Model Deployment window.

2. Enter the Model Name and Model Objective.

3. Click ⌕ . The filtered list is displayed.

Requesting and Authorizing Model Deployment

You can request a model for deployment and view its status in the Processing Modeling Framework (PMF). You can view the current status of the model deployment life-cycle and click on the various symbols in the PMF diagram to view details for the specific state in the workflow.

Note: For more information on PMF, see the *OFSAAI User Guide Release* 8.0.8.0.0 on OHC.

To request and authorize a model for deployment, perform the following steps:

1. Select the checkbox adjacent to Model ID and click 🗐 to display the *Model Deployment Authorization* window.

2. Select the **Request for Deployment** checkbox in the Model Deployment grid.

   Note: Once you have requested for deployment, you can view the status for the model deployment in the PMF.

   Click 🔒 from the header to display the Administration tools in Tiles menu. Click **Processing Modeling Framework** from the Tiles menu to view a list of links. Click **Process Monitor** from the list and search for the Model that was requested for deployment. Click the link to display the **PMF** window. The window displays the state of the Model deployment.

3. Select **Authorize** from the **Authorize and Deploy** checkbox to authorize the deployment request. This radio button is enabled if you are mapped to the MDLAUTH function role. If you are not mapped to the role, an authorizer mapped to the role has to approve the deployment.

   Note: Click the **My Inbox** link and then click **My Task** from the Navigation list. This link is available on the menu if your administrator has configured the required privileges to your user profile in the system. You can view notifications for the approval or rejection of the Model deployment here.

   For rejected requests, you can click the 🖌 **Sandbox** icon and edit the model from the **Model Creation** window. For more information, see Creating and Executing a Model. Once you have edited the Model, you can request again for approval to deploy and the process is a repeat of steps 1 to 3.
   Authorizers approving or rejecting model deployment requests can access from **My Inbox > My Task** as described previously. Click **Task ID** in the notification to display the *Model Deployment Authorization* window and approve or reject the model deployment.

4. Click **Save** to process the model deployment.

On successful deployment, the model is available in the Run Rule Framework as a component. Run filter and process filter is supported for models stitched in a Run for execution.

Note:

To execute Open R models, pass the value for $HOST as a Run-Time parameter.

Note:

Once deployed, you can view the success or failure of the Model deployment in the **PMF** window.

Click 🔒 from the header to display the Administration tools in Tiles menu. Click **Processing Modeling Framework** from the Tiles menu to view a list of links. Click **Process Monitor** from the list and search for the

Model that was deployed. Click on the link to display the **PMF** window. The window displays the state of the Model deployment as either Successful or Failed. You can click on the symbols to view more details. The following screenshot of the Model deployment on the PMF window is an example that has been through a cycle of model rejection and approval, and deployment failure and success.

Model deployment states on PMF window

Model Execution

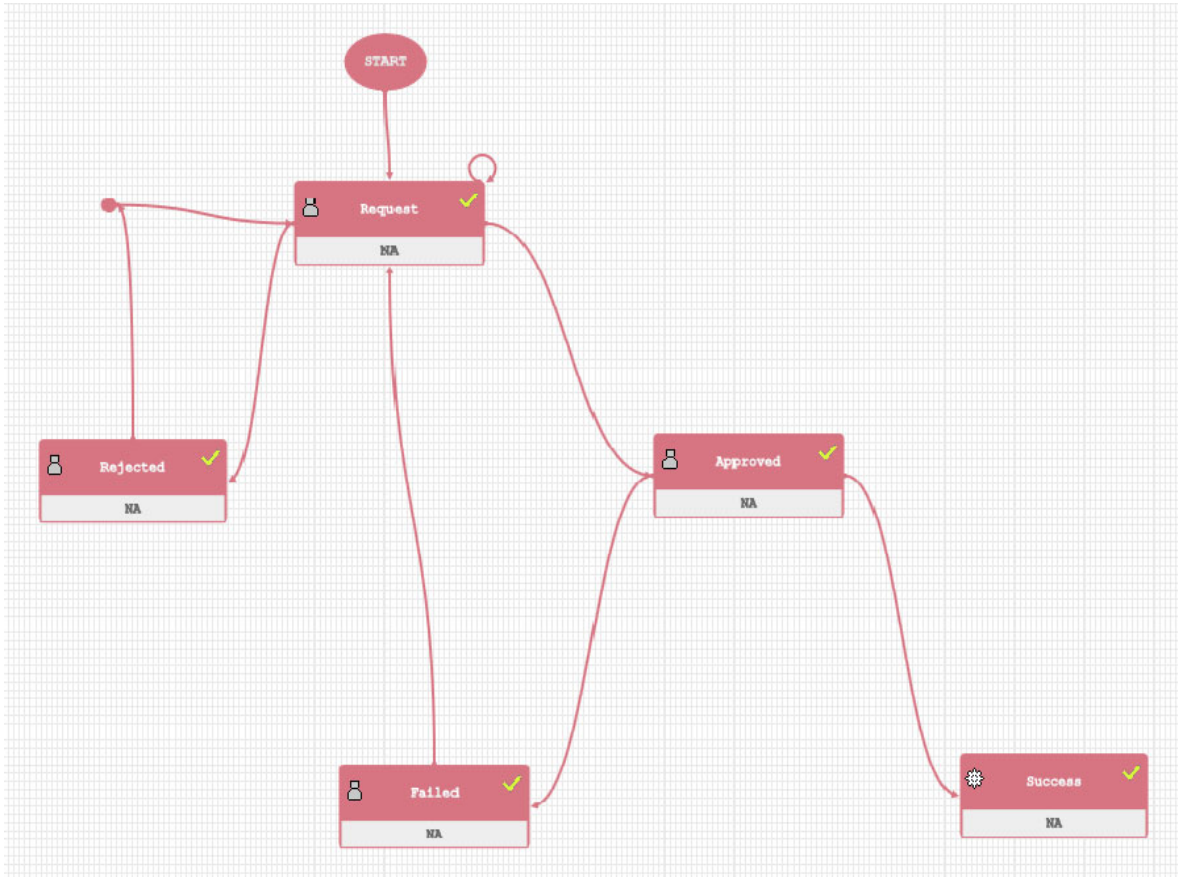The Model Execution module allows to you request for model execution and work with champion challenger.

This section includes the following topics:

- Request for Model Execution
- Champion Challenger

Request for Model Execution

Request for Model Execution in the Production Information Domain enables you to request for a batch execution for the selected model. A batch is scheduled in the Batch Processing and the model is executed in the production environment.

This section includes the following topics:

- Accessing Request for Model Execution

- Requesting for Model Execution

- Searching for Request for Model Execution

Accessing Request for Model Execution

To access the Request for Model Execution page, follow these steps:

1. Login to Oracle Financial Services Enterprise Modeling.

2. Click ⊞ from the header to display the applications in a Tiles menu.

3. Select **Financial Services Enterprise Modeling** from the Tiles menu. The Enterprise Modeling Application Home Page is displayed with the Navigation list to the left.

4. Click **Financial Services Enterprise Modeling** from the Navigation list to expand view the menu.

5. Click **Model Execution** and then click **Process Management** to view the menu.

6. Click **Model Management** and then click **Request for Model Execution**. The *Request for Model Execution* window is displayed.
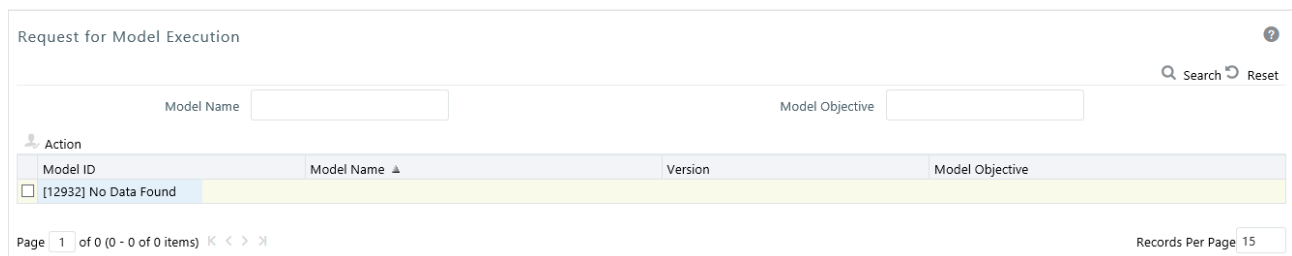
Request for Model Execution				?
				Q Search ⟲ Reset
Model Name		Model Objective		
⚊ Action				
Model ID	Model Name ▲	Version	Model Objective	
☐ [12932] No Data Found				
Page 1 of 0 (0 - 0 of 0 items) ⟦ ⟨ ⟩ ⟧			Records Per Page 15	

Figure 30. Request for Model Execution

Requesting for Model Execution

To request for model execution, follow these steps:

1. From the *Request for Model Execution* window, select the required model and click 🔲 . The Request for Model Execution window is displayed.

2. Select the **Register Batch** checkbox.

3. Click **Save** to process for model execution.

When there is a request for model execution, a batch is registered in the Operations framework which has to be executed from the Batch Scheduling window in the application. For more information, see *Operations* section in the *Oracle Financial Services Analytical Applications Infrastructure User Guide*.

Note: The Request for Model Execution page displays all deployed models in case of schema based sandboxes.

Note: Market Risk (MR) Models will not be displayed on the Request for Model Execution window. To create and execute a MR Model, go to the Run Definition window.

Searching for Request for Model Execution

The Request for Model Execution page allows you to search specific model execution request.

To search for Request for Model Execution, follow these steps:

1. Navigate to the Request for Model Execution page.

2. Enter the Model Name and Model Objective.

3. Click Q . The filtered list is displayed.

Champion Challenger

Champion refers to the first model deployed, after calibrating the parameters for a specific model objective. Challenger(s) refers to the subsequent models deployed for the same model objective but with different techniques used for predictions.

When several prediction models (challengers) are deployed and one of the challengers produces optimum output meeting all the prediction requirements, the same can be set as Champion.
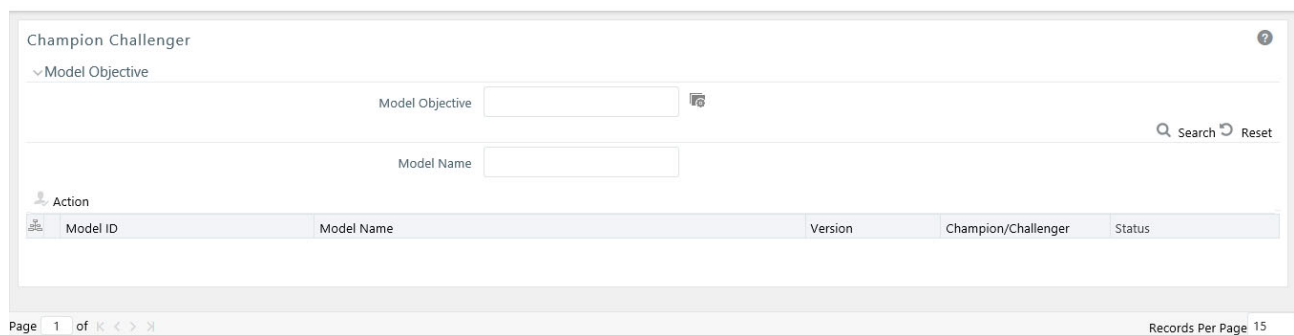
This section includes the following topics:

- Accessing Champion Challenger

- Searching for Champion Challenger

- Requesting New Champion

Accessing Champion Challenger

To access the Champion Challenger page, follow these steps:

1. Login to Oracle Financial Services Enterprise Modeling.

2. Click ⊞ from the header to display the applications in a Tiles menu.

3. Select **Financial Services Enterprise Modeling** from the Tiles menu. The Enterprise Modeling Application Home Page is displayed with the Navigation list to the left.

4. Click **Financial Services Enterprise Modeling** from the Navigation list to expand view the menu.

5. Click **Model Execution** and then click **Process Management** to view the menu.

6. Click **Model Management** and then click **Champion Challenger**. The *Champion Challenger* window is displayed.

Figure 31. Champion Challenger

Requesting New Champion

To request a new champion, follow these steps:

1. Select the Model Objective, click and open the Model Objective browser.

2. Select the required Model Objective in the Hierarchy list and click .

3. Click **OK**. The window is refreshed to display the available models.

4. Select the checkbox adjacent to the model whose status has to be set to champion.

5. Click in the Champion Challenger toolbar. The *Request New Champion* window is displayed.

6. Select the **Request New Champion** checkbox.
   If you have been mapped to MDLCHAMP function role, you can select **Authorize** checkbox to authorize new champion.

7. Click **Save** to update the changes and return to Champion Challenger window.
   In the Champion Challenger window, the status of the selected model is updated as Champion and Authorized.

Searching for Champion Challenger

The Champion Challenger page allows you to search for a specific champion challenger.

To search for Champion Challenger, follow these steps:

1. Navigate to the Champion Challenger page.

2. Enter the Model Name.

3. Click . The filtered list is displayed.

Configure Execution Venue

The machine details and compiler details required for model execution is configured on the *Model Execution Configuration* window.

This section includes the following topics:

- Prerequisites
- Accessing Configure Execution Venue
- Adding Model Execution Configuration
- Viewing Model Execution Configuration
- Modifying Model Execution Configuration
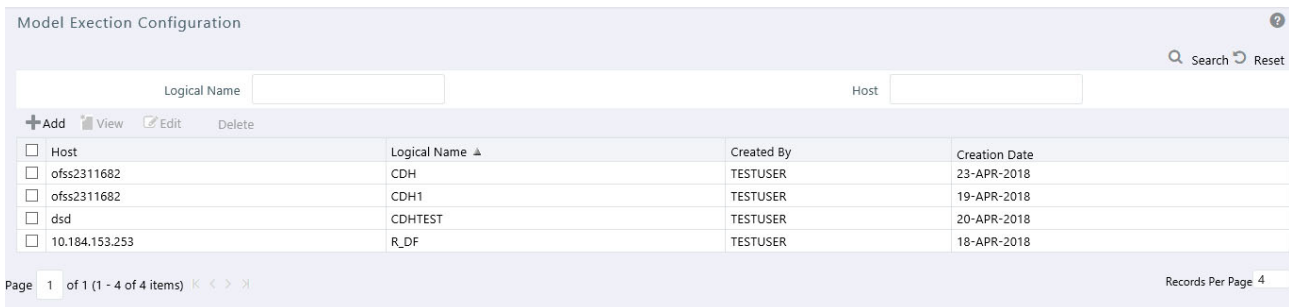- Deleting Model Execution Configuration

Prerequisites

Configure as instructed in the section *Configurations for OFSAAI Remote Invocation of Scripted Models Using Standard R Distributions* of the *OFSAAI Administration and Configuration Guide* on OHC.

Accessing Configure Execution Venue

To access the Configure Execution Venue menu, follow these steps:

1. Login to Oracle Financial Services Enterprise Modeling.

2. Click ⊞ from the header to display the applications in a Tiles menu.

3. Select **Financial Services Enterprise Modeling** from the Tiles menu. The Enterprise Modeling Application Home Page is displayed with the Navigation list to the left.

4. Click **Financial Services Enterprise Modeling** from the Navigation list to expand view the menu.

5. Click **Configure Execution Venue**. The *Model Execution Configuration* window is displayed.

Host	Logical Name ▲	Created By	Creation Date
ofss2311682	CDH	TESTUSER	23-APR-2018
ofss2311682	CDH1	TESTUSER	19-APR-2018
dsd	CDHTEST	TESTUSER	20-APR-2018
10.184.153.253	R_DF	TESTUSER	18-APR-2018

Page 1 of 1 (1 - 4 of 4 items)   Records Per Page 4

Figure 32. Model Execution Configuration

Adding Model Execution Configuration

You can create Model Execution Configuration from the *Model Execution Configuration* window.

To create a Model Execution Configuration, follow these steps:

1. Select ✚ button from the tool bar. The *Configure Execution Venue* window is displayed. The **Add** button is disabled if you select any existing Host in the grid.

2. Enter the Execution Venue Details. The fields are described in the following table:

Table 7. Execution Venue Details

Field	Description
Fields marked in red asterisk (*) are mandatory.	
Name	Enter name for the execution venue. This field is case-sensitive and does not allow duplication. For example, logical name "R_DF" is not allowed if a name "R_DF" exists. Ensure that there are no special characters such as `, {,},", ', ~, <,>, /, \, and multiple spaces.
Language	Select the language from the drop-down list. The options are: MATLAB and R.
Input Data Type	Select the input data type. The options are **ORE Frame**, **Data Frame** and **HDFS File**.
Machine Details	
User Name	Enter the user name with which Rserve is started on the remote machine.
Password	Enter the password for the preceding user.
Primary IP For Runtime Processes	Enter the machine's primary IP for runtime processes (IP address/host name of the remote machine where Rserve is running).
Shared Path	Enter the path of the Remote Machine or Server, which provides complete access rights to all users. The output of the executions will be created in the directory configured here. **Note:** Remote Machine or Server refers to the execution engine such as Rserve. The configured path should end with a directory separator at the end.
Output required in tables	Select **Yes** if you require the output in OFSAA tables.
Outputs required in CSV format	Select **Yes** if you require the output in CSV format.
Delete output files	Select **Yes** if you want to delete the output files. By default, this is **No**.
Class Name	Enter the Class Name of the execution implementation.
Is a primary node?	Select this option if the machine is a primary node.
SSH Port	Enter the SSH port number of OFSAA for file transfer between remote machine and OFSAA.
SFTP Port	Enter the SFTP port number of OFSAA for file transfer between remote machine and OFSAA.
MATLAB Specific Configurations	
Webservice URL	Enter the web service URL for executing script on MATLAB server.
Compiler Machine Details	
User name	Enter the user name for the compiler machine where the MATLAB compiler is running.
Password	Enter the password for the compiler machine.
Primary IP For Runtime Processes	Enter the machine's primary IP for runtime processes.
Compiler Path	Enter the compiler path where R code will be compiled.
SSH Port	Enter the SSH port number of OFSAA to transfer files to the MATLAB compiler machine.

Field	Description
SFTP Port	Enter the SSH port number of OFSAA to transfer files to the MATLAB compiler machine.
Script Configuration Note: This pane has two options: 1. Predefined scripts in the application and 2. Custom scripts that have to be uploaded.	
Predefined scripts	Select the required scripts and click ⟩ to move to selected scripts. Click ⌄ and ⌃ to change the order of the selected scripts.
Upload scripts	Click Browse button to upload custom scripts. Select the required scripts and click ⟩ to move to selected scripts. Click ⌄ and ⌃ to change the order of the selected scripts. Click ▮ to view the details of the selected scripts.

3. Once you have updated all the necessary details, click **Save** to save the configure execution venue details.

Viewing Model Execution Configuration

You can view the Model Execution Configuration from the *Model Execution Configuration* window.

To view the details for an existing Model Execution Configuration, follow these steps:

1. Select the checkbox adjacent to the **Host**.

2. Click ▮ button in the tool bar. The *Model Execution Configuration* window is displayed with details in view mode.

Modifying Model Execution Configuration

You can modify the Model Execution Configuration from the *Model Execution Configuration* window.

To modify an existing Model Execution Configuration, follow these steps:

1. Select the checkbox adjacent to the **Host** that you want to update.

2. Click 🖉 button in the tool bar. The *Model Execution Configuration* window is displayed in edit mode.

   Note: The **Edit** button is disabled if you select multiple Hosts.

3. Edit the Model Execution Configuration details as required. For more information on the fields, see Adding Model Execution Configuration.

4. Click **Save** to upload changes.

Deleting Model Execution Configuration

You can remove the Model Execution Configuration(s) that are no longer required in the system by deleting from the *Model Execution Configuration* window.

To delete a Model Execution Configuration, follow these steps:

1. Select the checkbox adjacent to a **Host**.

2. Click button from the tool bar. A confirmation dialog is displayed.

3. Click **OK** to confirm deletion.

CHAPTER 8   *Managing Stress Testing*

This chapter provides information about Modeling in the Oracle Financial Services Enterprise Stress Testing application and step-by-step instructions to use this section.

This chapter includes the following options:

- About Stress Testing
- Stress Definition

About Stress Testing

Stress testing is an integral part of a bank's risk measurement system and plays an important role in estimating the effects of potential financial crises on a bank's operations. Stress Testing or risk estimation technique, refers to the

process of examining the stability of a system or entity in adverse conditions. It involves testing beyond normal operational capacity, often to a breaking point, in order to observe the results. It also helps banks conduct analysis to estimate the impact of movements in the variables on specific measures such as profitability and capital adequacy.

The Stress Testing utility supports the stress testing requirements across the entire suite of OFSAA products. It allows banks to define shocks and assess the impact of such shocks across multiple business areas.

The two commonly accepted forms of Stress Testing are:

- **Sensitivity Analysis**: Shocks are applied on a single variable.
- **Scenario Analysis**: A scenario is defined as a shock to a single variable or a collection of shocks on multiple variables. Scenario analysis involves applying simultaneous shocks on multiple variables to assess the impact of scenario on a measure or a set of measures. Scenarios are further classified into the following categories:

    - **Historical Scenarios:** These scenarios replicate the past events docket. They are defined by specifying shocks to variables such that they replicate the movement seen during historical events.
    For example, the user may define a scenario that replicates the movement in stock market indices as observed during the catastrophic event of October 19, 1987. This scenario can then be applied to the current trading book portfolio of the bank to estimate the loss that might be incurred if a catastrophic event occurs. However, the historical scenarios may not cover the entire range of potential adverse conditions

    - **Hypothetical Scenarios**: These scenarios are based on user judgment and addresses the other possible adverse movements in the variables.
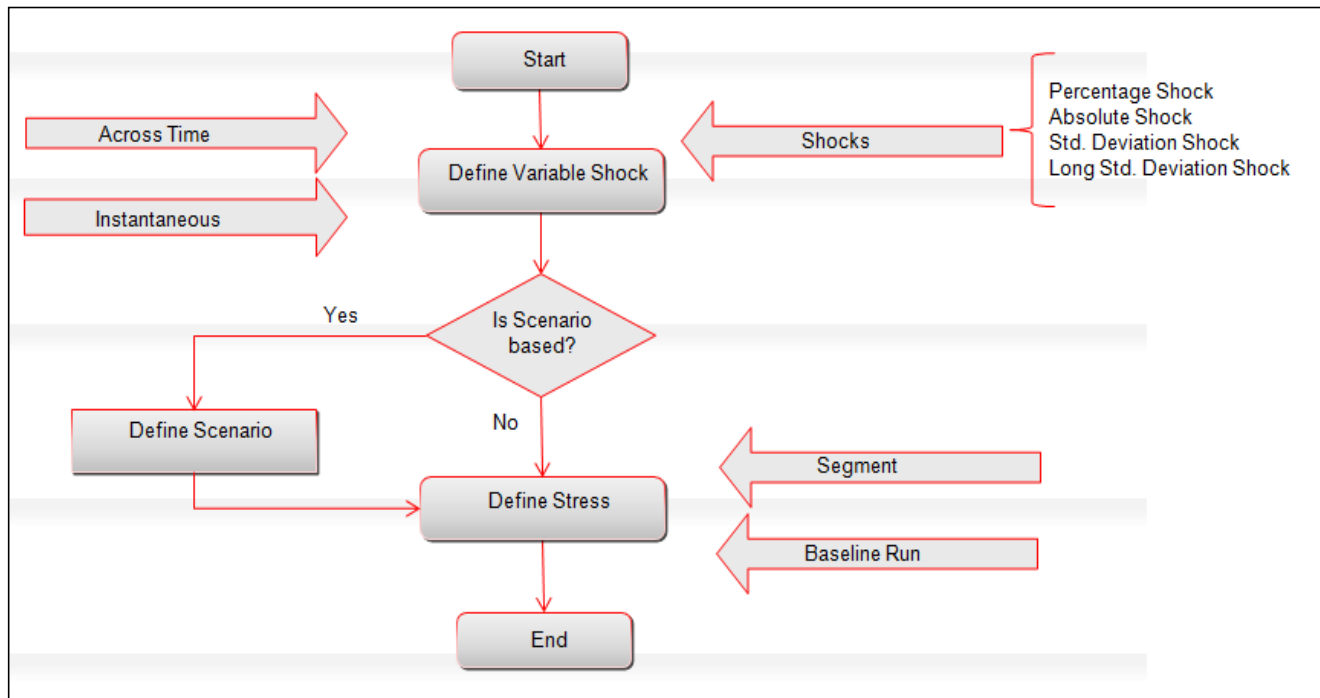
Figure 33. Stress Testing Workflow

Stress Definition

Stress refers to modeling a series of scenarios to analyze the impact of extreme market conditions and to measure the risk. This module enables you to create a stress definition. You can either specify a stand-alone execution of a Scenario, or map a Scenario(s) to a Baseline Run. The integrated process helps you to assess the impact of a Scenario across multiple areas. You can also view, modify, and delete stress definitions.

This section includes the following topics:

- Accessing Stress Definition
- Adding Stress Definition
- Executing Stress Definition
- Viewing Stress Definition
- Modifying Stress Definition
- Deleting Stress Definition
- Searching Stress Definition

Accessing Stress Definition

To access the Stress Definition page, follow these steps:

1. Login to Oracle Financial Services Enterprise Modeling.

2. Click ⊞ from the header to display the applications in a Tiles menu.

3. Select **Financial Services Enterprise Modeling** from the Tiles menu. The Enterprise Modeling Application Home Page is displayed with the Navigation list to the left.

4. Click **Financial Services Enterprise Modeling** from the Navigation list to expand view the menu.

5. Click **Stress Testing** and then click **Stress Definition**. The *Stress Definition* window is displayed.

		Stress ID	Stress Name ▲	Baseline Run	Created By	Creation Date
	☐	1522212900233	Stress for Fraud Minor	Stress Test F2T US	EMFUSER	28-MAR-2018 12:55:00 AM
	☐	1522212842558	Stress Test F2T US	test_model	EMFUSER	28-MAR-2018 12:54:03 AM

Figure 34. Stress Definition Page

Adding Stress Definition

Stress definition can be based on stress scenarios or business assumptions. A business assumption is essentially a process comprising of multiple rules which facilitate the computation of the impact of the assumption on base cash flows. Business assumptions are non-variable based.

Scenario based stress testing involves applying simultaneous shocks on multiple variables to assess the impact of scenario on a measure or a set of measures.

To add stress definition, follow these steps:

1. Select ![+] from the *Stress Definition List* tool bar. The *Stress Definition Add* window is displayed. The **Add** button is disabled if you have selected any checkbox in the grid.


Figure 35. Stress Definition Page

2. Enter the Stress Definition details as described in the following table:

Table 8. Stress Definition - Field Descriptions

Field	Description
Fields marked in red asterisk (*) are mandatory.	
Stress Name	Enter a name for the stress definition. Ensure that there are no special characters like `, {,},", ', ~, <,>, /, \, and multiple spaces.
Stress Definition Description	Enter the required description for the stress definition.
Segment	Select the Segment from the drop-down list. The list displays all segments mapped to your user group.
Base Line Run	1. Click ![icon] and open the Baseline Run browser. The available Baseline Runsin the selected Segment are listed in the Hierarchical pane. **Note**: 2. Select the required Baseline Run based on which stress definition is to be created and click ![>] . 3. Click **OK**. 4. Click ![icon] and view the selected baseline Run details.
	5.

6. In the *Run Details* grid, all the processes and tasks associated with the selected baseline run is displayed.

- You can insert a new process to the selected Baseline run.

- You can replace a process in the selected baseline run with another process. The process replacement can happen only for the processes which are directly placed under the run.

  - Select the checkbox adjacent to the process and click ![icon] to replace it with another process. The *Tasks Browser* window displays all the processes which are defined in the selected segment and infodom.

- Select the required process from the list and click [>]. You cannot select a process which is already present in the baseline run.

- You can modify or replace a task within a process with another task.

  - Select the checkbox adjacent to the task and click [✎]. The Tasks Browser window displays all the tasks which are defined in the selected segment and infodom.

  - Select the required process from the and click [>].

- Select a task and click [🗑] to delete it.

- Select a process and click [🗑] to delete it.

- Set the precedence in which the tasks need to be executed.

  - Click [🗂] adjacent to the Precedence field of a task. The available scenarios/ tasks are listed in the Precedence Browser window.

  - Select the tasks in the order in which they need to be executed and click [>]. You can change the order by clicking [^] and [⌄].

- Provide the Parameters for the task in the **Parameters** field.

  Note: If the configuration is set to RRF, then the Parameters field value must be specified within double quotes.

7. Click **Save**. The stress definition details are saved and displayed in the *Stress Definition* window.

Executing Stress Definition

After creating stress definition, you need to create Manage Run Definition. Based on the Run Type (Single Request or Multiple Request), you need to execute the stress definition from Batch Execution window or Batch Group Execution window respectively. You can also monitor the status of stress definition execution.

To execute stress definition, follow these steps:

1. Login to Oracle Financial Services Enterprise Modeling.

2. Click [⊞] from the header to display the applications in a Tiles menu.

3. Select **Financial Services Enterprise Modeling** from the Tiles menu. The Enterprise Modeling Application Home Page is displayed with the Navigation list to the left.

4. Click **Financial Services Enterprise Modeling** from the Navigation list to expand view the menu.

5. Click **Model Execution** and then click **Process Management** to view the menu.

6. Click **Execution** and then click **Manage Run Execution**. The *Manage Run Execution* window is displayed.

7. Click [➕] from the List toolbar. The Manage Run Definition window is displayed.

8. Click in the Run field and select the Stress Definition that you want to execute from the Run Selector window. For more information, see the *Manage Run Definition* section in the *Oracle Financial Services Analytical Applications Infrastructure User Guide.*

9. Click **Save**.

10. Execute the stress definition:

- If you have selected Type as Single Request in the Manage Run Execution window:

   a. Go to **Execution > Batch Execution**. The Batch Execution window is displayed. For more information on how to execute a Batch, see the *Execute Batch* section in the *Oracle Financial Services Analytical Applications Infrastructure User Guide.*

   b. Click **Execute Batch**.

- If you have selected Type as Multiple Request in the Manage Run Execution window:

   a. Go to **Execution > Batch Group Execution**. The Batch Execution window is displayed. For more information on how to execute a Batch Group, see the *Batch Group Execution* section in the *Oracle Financial Services Analytical Applications Infrastructure User Guide.*

   b. Click **Start**.

To monitor the stress execution:

- For Batch Execution:

   a. Go to **Execution > Batch Monitor**. The Batch Monitor window is displayed.

   b. Select the Stress Definition whose execution you want to monitor and select the Batch Run details. For more information, refer to the *Monitor Batch* section in the *Oracle Financial Services Analytical Applications Infrastructure User Guide.*

   c. Click to start monitoring.

- To view log files:

   a. Login to Oracle Financial Services Enterprise Modeling.

   b. Click from the header to display the applications in a Tiles menu.

   c. Select **Financial Services Enterprise Modeling** from the Tiles menu. The Enterprise Modeling Application Home Page is displayed with the Navigation list to the left.

   d. Click **Common Tasks** from the Navigation list to expand view the menu.

   e. Click **Operations** to view the menu.

   f. Click **Processing Report.** The *Batch Processing Report* window is displayed.

   g. Select the Information Date and Batch Status from the drop-down lists. For more information, see the *Batch Processing Reports* section in the *Oracle Financial Services Analytical Applications Infrastructure User Guide.*

Viewing Stress Definition

You can view individual Stress Definition details at any given point.

To view the existing Stress Definition details, follow these steps:

1. Select the checkbox adjacent to the Stress ID.

2. Click ![icon] in the Stress Definition tool bar. The Stress Definition View window is displayed with the stress definition details along with the mapped scenarios.

Modifying Stress Definition

You can modify only the Stress Description and Scenario for any selected Stress Definition.

To modify an existing Stress Definition, follow these steps:

1. Select the checkbox adjacent to the Stress ID whose details are to be updated.

2. Click ![icon] in the Stress Definition List tool bar. The Edit button is disabled if you have selected multiple Stress IDs. The Stress Definition Edit window is displayed.

3. Edit the Stress Definition details as required. For more information, refer to *Add Stress Definition*.

   Note: While editing Stress Definition, you cannot unmap a scenario that is mapped to the stress definition. You can only change the version of the scenario.

4. Click **Save** to save changes.

Deleting Stress Definition

You can remove the stress definitions which are no longer required in the system, by deleting from the Stress Definition Summary window.

Note: Stress definitions, which use PR2 Base Line Run and has batch registered for processing, cannot be deleted. However, Stress definitions, which use RRF Base Line Run, can be deleted irrespective of whether it has batch registered for processing.

To delete a Stress Definition, follow these steps:

1. Select the checkbox adjacent to the Stress ID whose details are to be removed.

2. Click ![icon] in the Stress Definition List tool bar.

3. Click **OK** in the information dialog to confirm deletion.

Searching Stress Definition

The Stress Definition page allows you to search specific model execution request.

To search for Stress Definition, follow these steps:

1. Navigate to the Stress Definition page.

2. Enter the Stress Definition Name.

3. Click ![icon]. The filtered list is displayed.

CHAPTER 9 — *Creating and publishing models using integration of OFS EM with EMF Studio*

The Oracle Financial Services Enterprise Modeling (OFS EM) application allows creating and publishing Models from external applications such as Enterprise Modeling Framework (EMF) Studio. The advantage is that Model Creation, which is done through the OFS EM application, is also possible through the integration of OFS EM with applications such as EMF Studio.

EMF Studio is browser-based workbench for creating Models and publishing to OFSAA. It provides a system for rapid creation and deployment of Models. RScript queries on Workspaces derives data from existing scenarios and you can test the scenarios with what-if analysis. What-if analysis helps you visualize data and lets you create Models

that are fit to be published. You can access the metadata from OFSAA and use it to create Models within Notebooks, which you can publish back on OFSAA.
This chapter includes the following topics:

- Prerequisites
- Logging into EMF Studio
- Creating Notebooks
- Creating Paragraphs
- Publishing Models
- Using python to create notebooks and paragraphs, and publishing models

Prerequisites

The prerequisites to use this functionality on OFS EM are in the following list:

1. Users must have the requisite permissions to access EMF Studio.

2. Users must know how to use EMF Studio for features such as Creating Notebooks and Paragraphs.

3. EMF Studio servers must be up and running. See the OFS AAAI Applications Pack Installation Guide for more information.

4. OFS EM studio service must be up and running. See the OFS AAAI Applications Pack Installation Guide for more information.

5. Users must have Sandboxes, Datasets and Variables created in the OFS EM application to create Models. See previous chapters in this guide for more information.

Logging into EMF Studio

Login to EMF Studio and create Notebooks. The following is the procedure to login to EMF Studio:

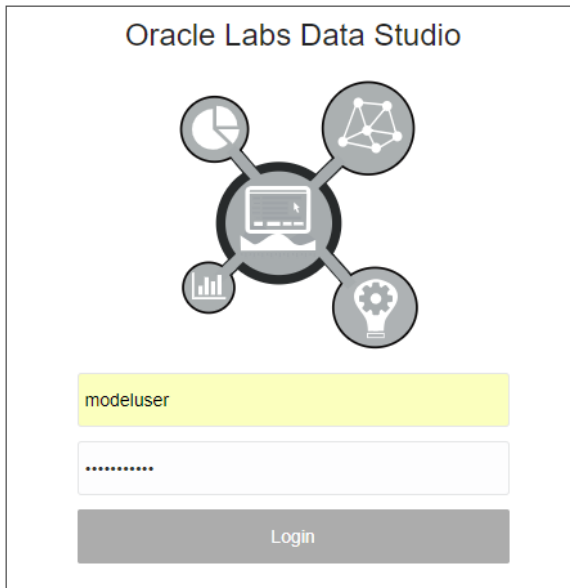1. Enter the URL for EMF Studio in a web browser to display the *Login* window.

Figure 36. EMF Studio Login Window

2. Enter the details in the **Username** and **Password** fields, and click **Login** to display the *EMF Studio Home* window.

   Note: The user name and password for login is your OFSAA login ID and password.

Creating Notebooks

Notebooks contain a collection of Paragraphs, which is used to code or query the database. The EMF Studio Home window displays the Notebooks that are mapped to your role. It displays details for the Notebook's Name, Author and Tags.

The following is the procedure to create a Notebook:

1. Click **Create** from the *EMF Studio Home* window to view the *Create Notebook* popup.
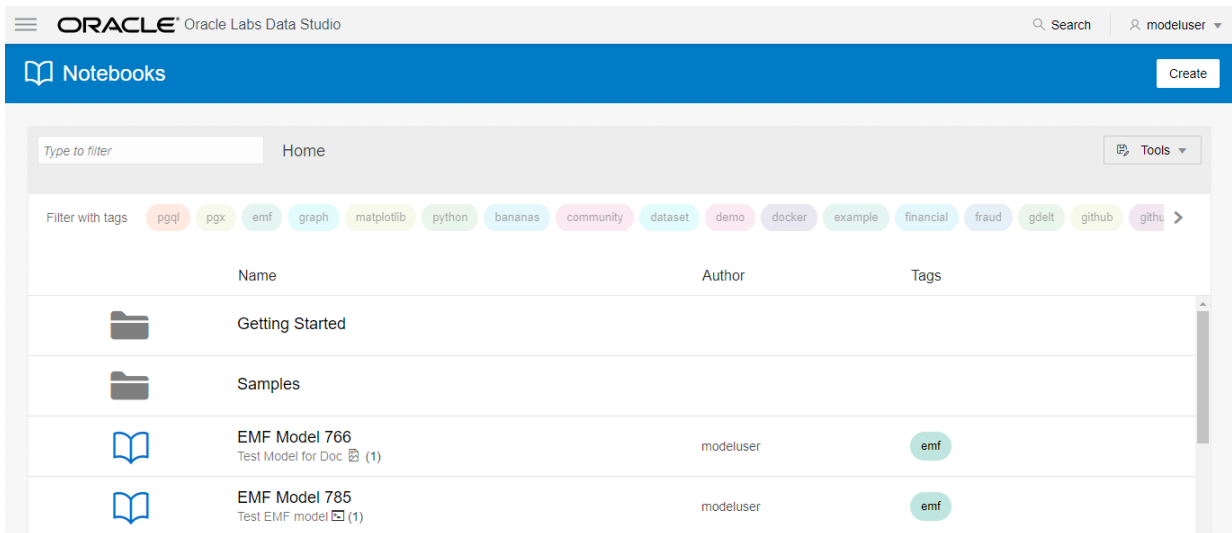

Figure 37. EMF Studio Home Window

2. On the popup, enter details for **Name** and **Description**. Enter tag details in **Tag**, if required.
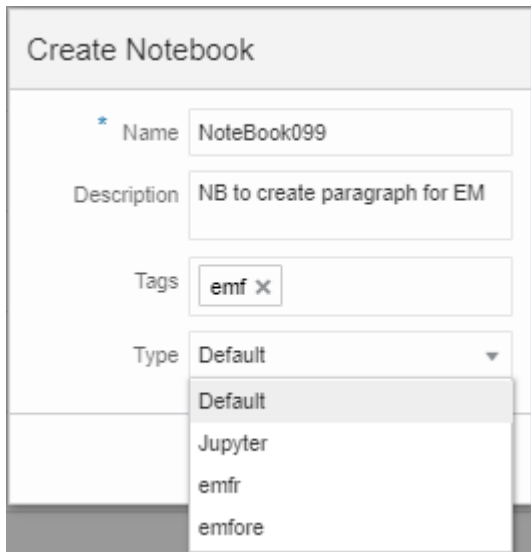
Figure 38. Create Notebook Popup
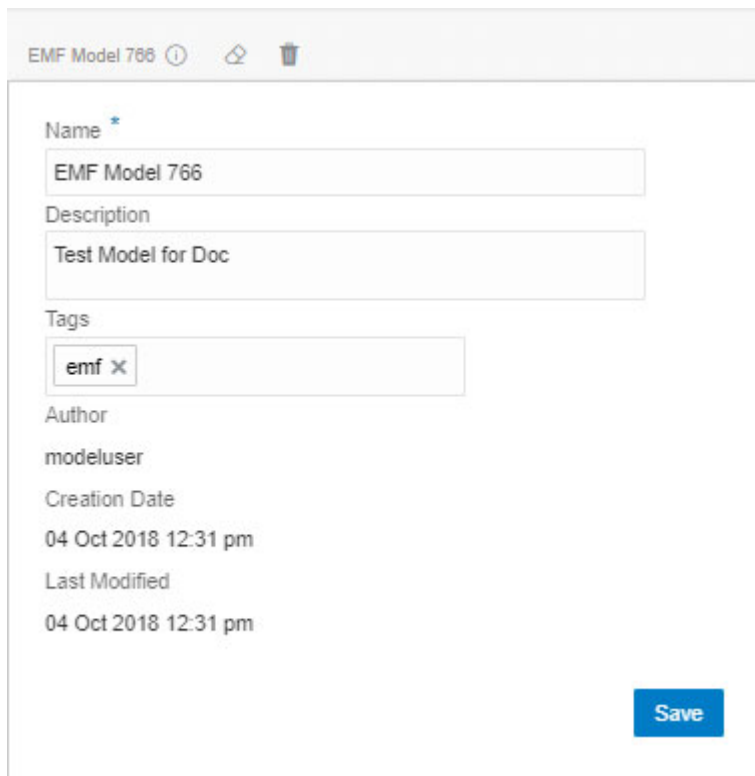
3. Select from the relevant option for the type of Notebook from **Type**. For OFS EM, select from the options **emfr** and **emfore** for R Language, or select **Default** for Python.

4. Click **Create** to create the Notebook and view the *Paragrap*h window.

Editing a Notebook

The following is the procedure to edit Notebooks:

1. Click a Notebook from the list of Notebooks from the *EMF Studio Home* window to view the *Paragraph* window.

2. Click ⓘ to view the *Notebook Details* popup. Modify the details if required.

Figure 39. Notebook Details Popup

3. Enter details for **Name** and **Description**. Enter tag details in **Tag**, if required or click **X** to remove existing tags.

4. Select the relevant option for the type of Notebook from **Type**.

5. Click **Save**.

Creating Paragraphs

Create Paragraphs to run your code or query using the various interpreter-friendly scripting languages supported in EMF Studio. You can use Paragraphs to query the database and extract datasets and variables information. You can then use this information to apply formulas and construct Models, which you can publish to OFSAA.

1. Click a Notebook from the list of Notebooks from the *EMF Home* window to view the *Create Paragraph* window. This window is displayed if you are creating a Paragraph for the first time in the Notebook. The following is an example:
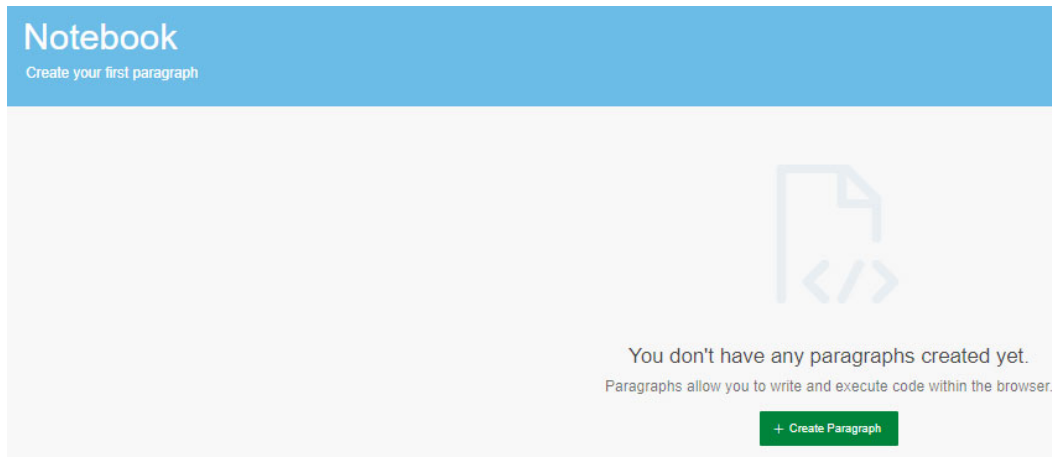
Figure 40. Create Paragraph Window

> **Note:** If you have created a Paragraph, the *Paragraph* window is displayed.

2. Click **Create Paragraph** to view the *Paragraph* window.
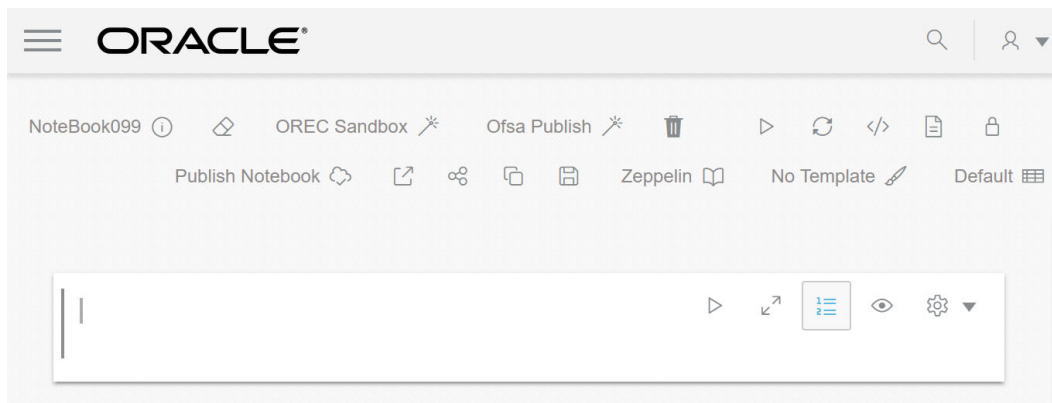
Figure 41. Paragraph Window

3. Click **Workspaces** ⚒ to select the required Workspace. If there's only one Workspace, then by default, the Workspace is displayed in the drop-down field.

4. Enter a Dataset query using **RScript** in the *Paragraph* form and click **Run** ▷ to view the result in the form area following the query. Click **HTML** 📄 to view the result in *HTML View* format. The following illustration is an example:

Figure 42. Dataset query result example in EMF Studio

5. Hover the mouse pointer over the top or bottom of the form to display the set of buttons as shown in the following illustration and click **Add Paragraph** ➕ button to add a new paragraph.

Figure 43. Hover mouse pointer to view Add Paragraph button

6. Enter a RScript query, in the new Paragraph form, for the Variables selected from the list shown for the Dataset.

7. Click **Run** ▷ to view the result in the form area following the query. Click **HTML** to view the result in HTML view format. This is similar to the query operation described previously for Datasets.

```
va=ofs.listVariables("Operational  Loss")
va
```

```
[1] "Scaled_Loss_Severity"   "Unscaled_Loss_Severity"
```

115 ms @ 11:43:12
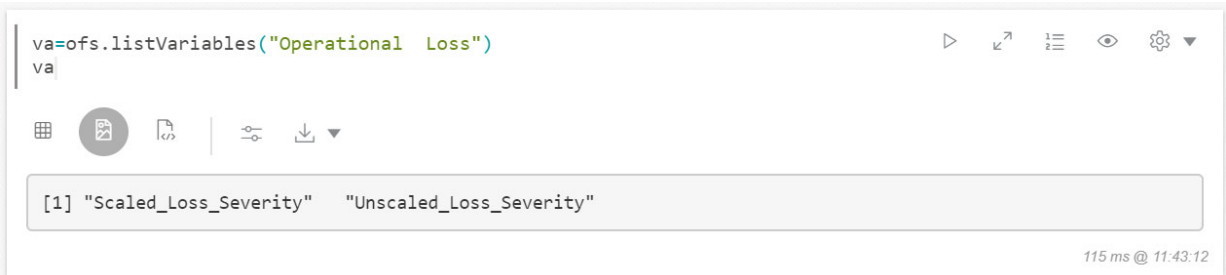
Figure 44. Variables query result example in EMF Studio

8. Hover the mouse pointer over the top or bottom of the form to display the set of buttons and click **Add Paragraph** ⊕ button to add a new paragraph.

9. Enter a RScript query, in the new paragraph form, for the Dataframes selected from the list shown for the Variables. Click **Run** ▷ to view the result in the form area following the query. Click **Table** to view the result in Table view format.

```
x=ofs.as.dataframe("Operational  Loss",c("Scaled_Loss_Severity","Unscaled_Loss_Severity"))
head(x)
```

SCALED_LOSS_SEVERITY	UNSCALED_LOSS_SEVERITY	PK_1	PK_2	PK_3	PK_4	PK_5	PK_6	PK_7	PK_8	PK_9	PK_
NA	394.563	2009-12-31	1190566	2	NA	2009-12-31	NA	5	107	108	NA
NA	394.563	2009-12-31	1190573	2	NA	2009-12-31	NA	5	102	103	NA
NA	394.563	2009-12-31	1190578	2	NA	2009-12-31	NA	4	104	105	NA
NA	394.563	2009-12-31	1190587	2	NA	2009-12-31	NA	4	103	101	NA
NA	394.563	2009-12-31	1190591	2	NA	2009-12-31	NA	5	107	106	NA
NA	394.563	2009-12-31	1190593	2	NA	2009-12-31	NA	4	101	105	NA

Page 1 of 1 (1-6 of 6 items) |◁ ◁ 1 ▷ ▷|
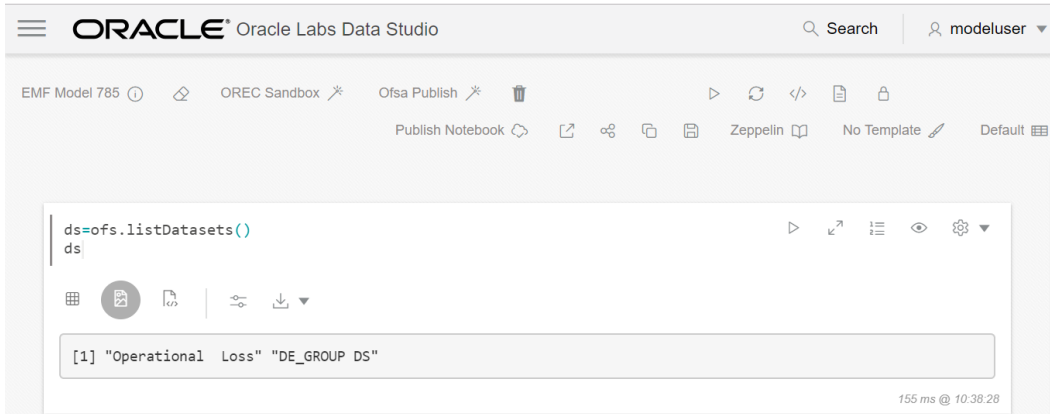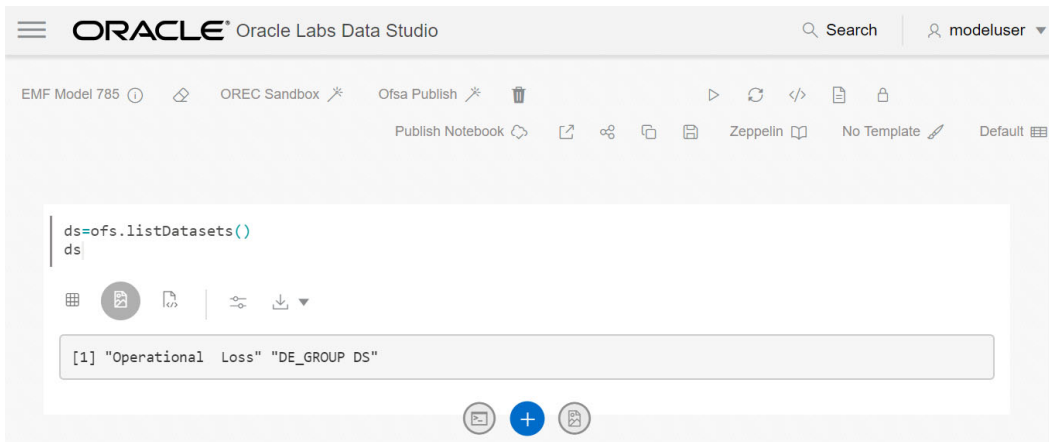
623 ms @ 11:57:22
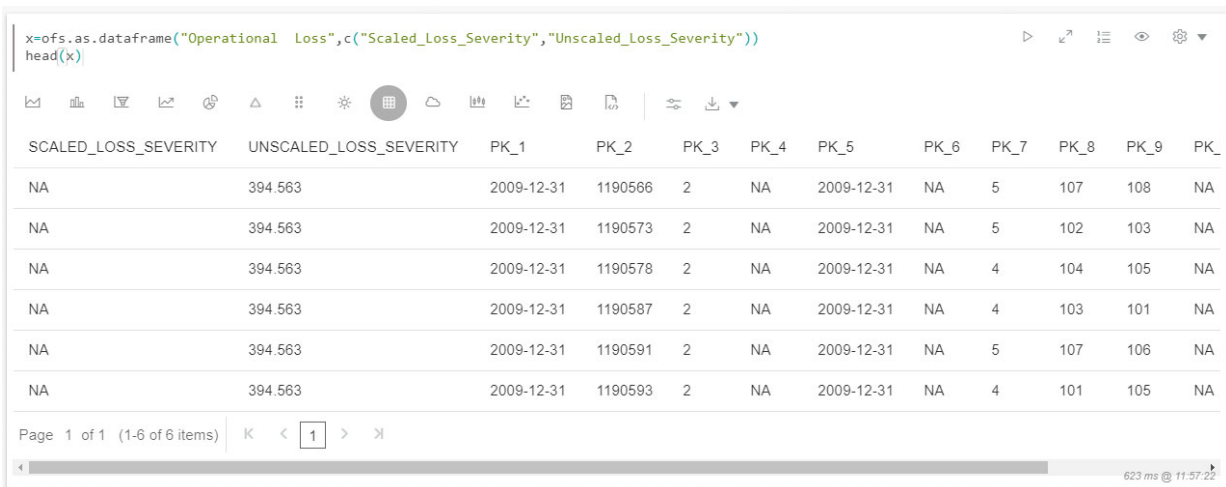
Figure 45. Dataframes query result example in EMF Studio

10. Hover the mouse pointer over the top or bottom of the form to display the set of buttons and click **Add Paragraph** ⊕ button to add a new paragraph.

11. Enter RScript formula queries that can give visual representation of modified data. You can use this to extend on the available dataframes data and create views of Models in the new Paragraph form.

12. Click **Run** to view the result in the form area following the query. Click **HTML** to view the result in HTML view format.


```
op=lm(x[[1]]~x[[2]])
summary(op)


Call:
lm(formula = x[[1]] ~ x[[2]])

Residuals:
    Min      1Q  Median      3Q     Max
-143948    -122    -119    -103  351575

Coefficients:
             Estimate Std. Error  t value Pr(>|t|)
(Intercept) 1.233e+02  1.684e+01    7.326 2.41e-13 ***
x[[2]]      9.976e-01  2.603e-04 3832.901  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3592 on 46952 degrees of freedom
  (3046 observations deleted due to missingness)
Multiple R-squared:  0.9968,    Adjusted R-squared:  0.9968
F-statistic: 1.469e+07 on 1 and 46952 DF,  p-value: < 2.2e-16
```
268 ms @ 11:58:02

Figure 46. Example of RScript formula application on dataframes result in EMF Studio

You can query using various combinations of formulas and derive Models that fit your requirement. You can then publish these Models to OFSAA.

Publishing Models

After creating Models from the Paragraph window, you can publish it to OFSAA. The following is the procedure to publish:

1. Click [Publish Notebook] to view the *Publish Notebook* popup.
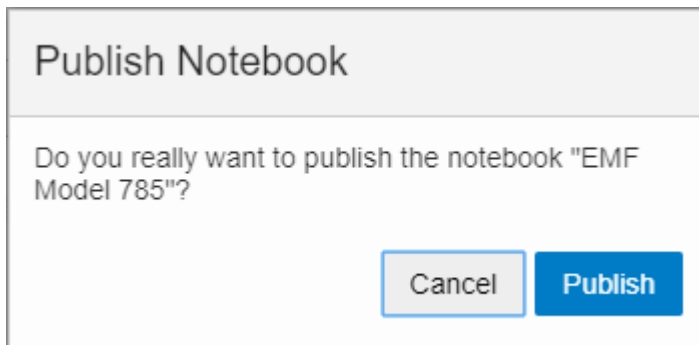
Figure 47. Publish Notebook Popup

2. Click **Publish**. The Notebook is published to OFSAA as a Model. Similarly, you can republish Notebooks to overwrite with changes to the existing Notebooks. Click [Publish Changes] to view the Re*publish Notebook* popup. Click **Publish**.
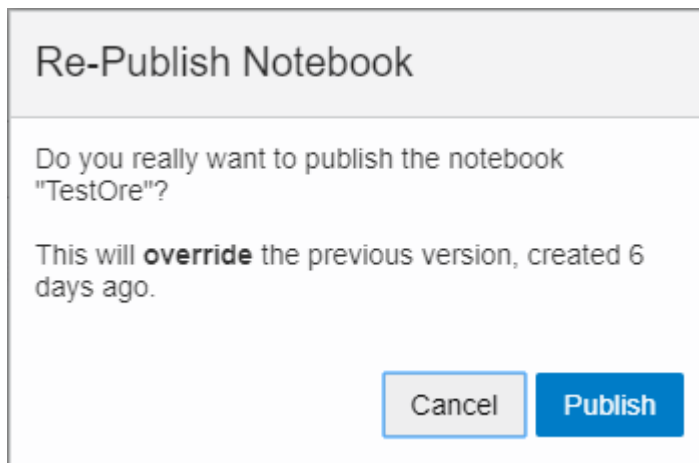
Figure 48. Republish Notebook Popup

Using python to create notebooks and paragraphs, and publishing models

The previous sections describe creating Notebooks and Paragraphs, and publishing Models for R Language. For most purposes, the procedure is similar for Python, except that there are a few minor differences, which are described in this section. The procedure for Python is in the following:

1. Create Notebooks. See Creating Notebooks for detailed information.

   Note: For Python, select **Default** for the type of Notebook from **Type**.

2. Create Paragraphs using Python scripting. See Creating Paragraphs for detailed information.

   Note: For Python, **Workspaces** and **Publish** is not available on the *Paragraph* window. This requires that you enter instructions in Python format in the Paragraph forms to create Paragraphs and Publish the Models to OFSAA.

3. Enter a query using **Python** scripting, in the *Paragraph* form, to derive a list of workspaces. Click **Run** ▷ to view the result in the form area following the query. Click **HTML** 🗎 to view the result in *HTML View* format. The following illustration is an example:


Figure 49. List of workspaces for python result example

   Note: Queries or scripts for Python, entered in the Paragraphs for Data Studio, must begin with **%emfpy**.

4. Hover the mouse pointer over the top or bottom of the form to display the set of buttons and click **Add Paragraph** ⊕ button to add a new paragraph.

5. Enter a query using **Python** scripting, in the new *Paragraph* form, to select the workspace from the list derived in the previous steps.

6. Click **Run** ▷ to view the result in the form area following the query. Click **HTML** 📄 to view the result in HTML view format. This is similar to the query operation described previously.

```
%emfpy
ofs.attachWorkSpace('OREC Sandbox')
```

True

Figure 50. Attach workspace python query result in EMF Studio

7. Hover the mouse pointer over the top or bottom of the form to display the set of buttons and click **Add Paragraph** ⊕ button to add a new paragraph.

8. Enter the list of Dataset query using **Python** in the *Paragraph* form and click **Run** ▷ to view the result in the form area following the query. Click **HTML** 📄 to view the result in *HTML View* format. The following illustration is an example:

```
%emfpy
ds=ofs.listDatasets()
ds
```

[1] "Operational Loss" "DE_GROUP DS"

Figure 51. Dataset python query result example in EMF Studio

9. Hover the mouse pointer over the top or bottom of the form to display the set of buttons and click **Add Paragraph** ⊕ button to add a new paragraph.

10. Enter the Python query, in the new Paragraph form, for the Variables selected from the list shown for the Dataset.

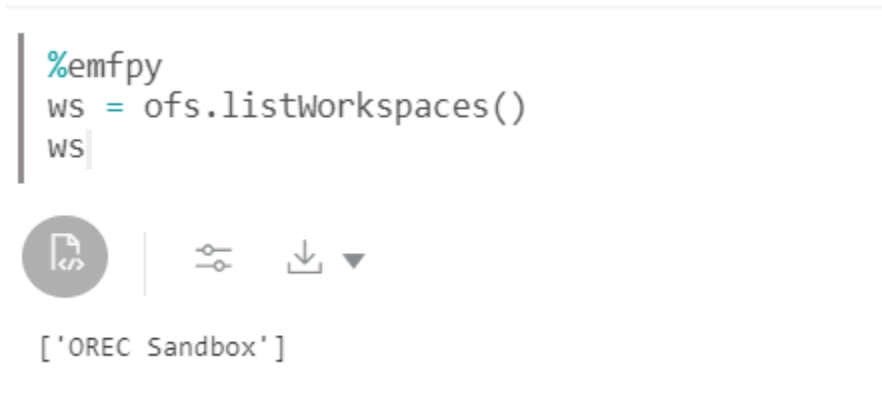11. Click **Run** ▷ to view the result in the form area following the query. Click **HTML** 📄 to view the result in HTML view format. This is similar to the query operation described previously for Datasets.

```
%emfpy
va=ofs.listVariables("Operational  Loss")
va
```

[1] "Scaled_Loss_Severity" "Unscaled_Loss_Severity"
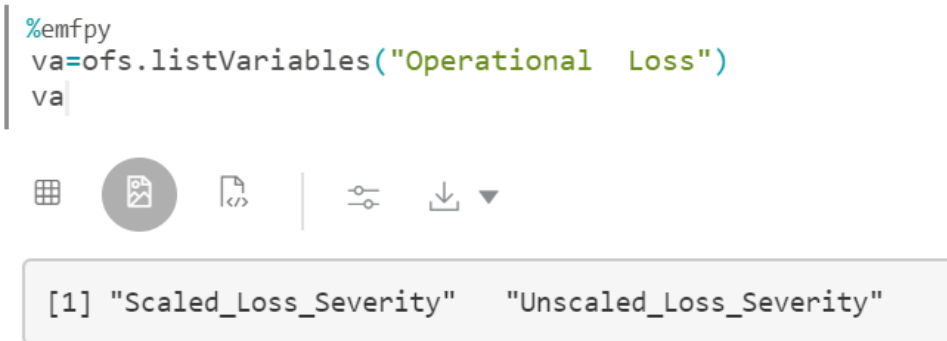
Figure 52. Variables python query result example in EMF Studio

12. Hover the mouse pointer over the top or bottom of the form to display the set of buttons and click **Add Paragraph** ⊕ button to add a new paragraph.

13. Enter the Python query, in the new paragraph form, for the Dataframes selected from the list shown for the Variables. Click **Run** ▷ to view the result in the form area following the query. Click **Table** ⊞ to view the result in Table view format.

```
%emfpy
x=ofs.as.dataframe("Operational  Loss",c("Scaled_Loss_Severity","Unscaled_Loss_Severity"))
head(x)
```

SCALED_LOSS_SEVERITY	UNSCALED_LOSS_SEVERITY	PK_1	PK_2	PK_3	PK_4	PK_5	PK_6	PK_7	PK_8	PK_9	PK_
NA	394.563	2009-12-31	1190566	2	NA	2009-12-31	NA	5	107	108	NA
NA	394.563	2009-12-31	1190573	2	NA	2009-12-31	NA	5	102	103	NA
NA	394.563	2009-12-31	1190578	2	NA	2009-12-31	NA	4	104	105	NA
NA	394.563	2009-12-31	1190587	2	NA	2009-12-31	NA	4	103	101	NA
NA	394.563	2009-12-31	1190591	2	NA	2009-12-31	NA	5	107	106	NA
NA	394.563	2009-12-31	1190593	2	NA	2009-12-31	NA	4	101	105	NA

Page 1 of 1 (1-6 of 6 items) |< < 1 > >|

623 ms @ 11:57:22
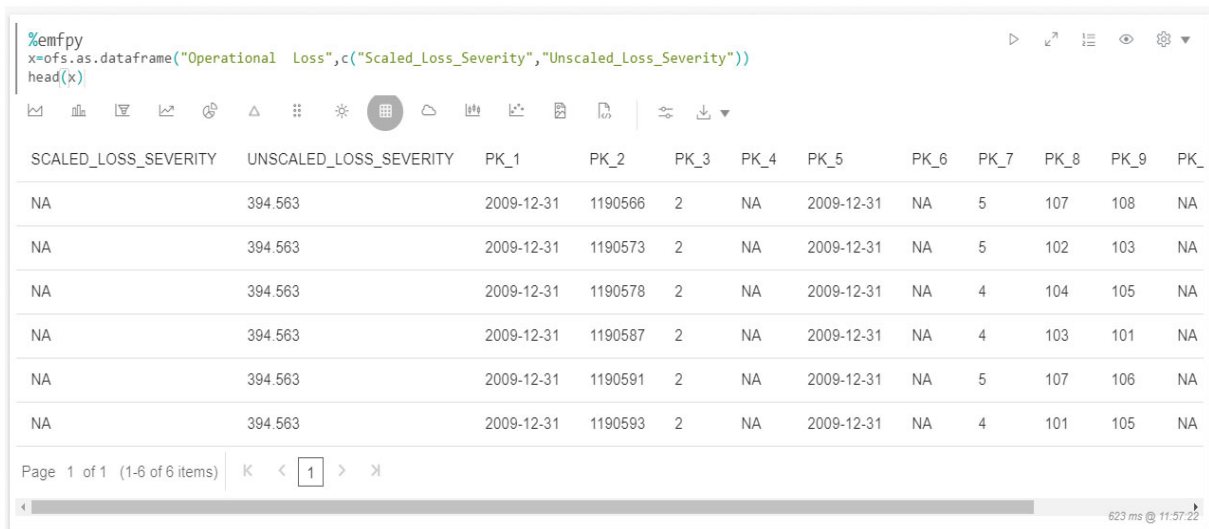
Figure 53. Dataframes python query result example in EMF Studio

14. Hover the mouse pointer over the top or bottom of the form to display the set of buttons and click **Add Paragraph** ⊕ button to add a new paragraph.

15. Enter the Python formula queries that can give visual representation of modified data. You can use this to extend on the available dataframes data and create views of Models in the new Paragraph form.

16. Click **Run** ▷ to view the result in the form area following the query. Click **HTML** to view the result in HTML view format.

Figure 54. Example of python formula application on dataframes result in EMF Studio

17. You can query using various combinations of formulas and derive Models that fit your requirement. You can then publish these Models to OFSAA.

   Note: The publishing procedure for Python varies from the publishing procedure for RScript. For RScript publish, see Publishing Models.

18. Enter the Python instruction to register and publish the model to OFSAA. Click **Run** ▷ to publish and view the result in the form area following the query. Click **HTML** to view the result in HTML view format.


```
%emfpy
ofs.register("name the note book",True)
```

Figure 55. Example of python instruction to publish to OFSAA

The model is published to OFSAA.

APPENDIX A   *References*

This section of the document consists of information related to intermediate actions that needs to be performed while completing a task. The procedures are common to all the sections and are referenced where ever required. You can refer to the following sections based on your need.

R Scripts and Oracle R Enterprise (ORE) Statistical Functions

Data usage when R script is used entirely

While defining a model scripted in R, user can select **Dataset** and **Variables** to assign data (table columns) to the R/ ORE objects used within the script. Oracle Financial Services Enterprise Modeling application prepares data from the dataset, variables and other attributes like filters chosen for the model and will make the same available as user specified R objects/ frames.

No specific treatment is required in the script for using the data. User can just have an R data.frame object say 'GDP' and a variable (table.column) assigned to it and use 'GDP' as is in the script.

Data usage when ORE native implementation is employed - Oracle Financial Services Enterprise Modeling application - Specifics to be considered

The data which is prepared from the dataset and variables as explained in the previous section, will be available in this case as a named ORE ore.frame object ('OFSDATASET') when ORE native implementations are used (that is, when the **Is ORE implementation used?** checkbox is selected). Hence, the R objects(data) must always be accessed through the ore.frame object 'OFSDATASET'.

For instance, if two objects say 'x' and 'y' are used and dataset/ variables are chosen for 'x' and 'y' in the model definition, then the objects 'x' and 'y' should be accessed in the script as 'OFSDATASET$x' and 'OFSDATASET$y' respectively.

An illustration to explain the R and ORE cases is given in the following section:-

For a simple regression model which is entirely scripted in R, that uses a dataset and three variables (DependentVariable, IndependentVariable1 and IndependentVariable2), the user should not select the **Is ORE implementation used?** checkbox. In this case the R script is as follows:

```
art.mod<-lm(DependentVariable ~ IndependentVariable1 + IndependentVariable2)
art.summ<-summary(art.mod)
coef( art.summ )
art.summ[[ "r.squared" ]]


#do some line plots
new.x.datafrme = data.frame(x=seq(from=range(IndependentVariable1)[ 1 ],to=range(
IndependentVariable1)[2],length=length(DependentVariable)))
```

Here the variables chosen for the model are accessed directly in the script as IndependentVariable1, IndependentVariable2, and DependentVariable.

Whereas for a parallel ORE implementation of the same (a dataset and three variables DependentVariable, IndependentVariable1 and IndependentVariable2) using ORE statistical functionalities, user must check the **Is ORE implementation used?** checkbox and the script is as follows:

```
art.mod<-ore.lm(DependentVariable ~ IndependentVariable1 + IndependentVariable2,
data=OFSDATASET)
art.summ<-summary(art.mod)
coef( art.summ )
art.summ[[ "r.squared" ]]
```

```
#do some line plots
new.x.orefrme = data.frame(x=seq(from=range(OFSDATASET$IndependentVariable1 )[ 1
],to=range(OFSDATASET$IndependentVariable1)[2],length=
length(OFSDATASET$DependentVariable)))
#perform some operations on the new data
```

Here the variables chosen for the model are accessed from OFSDATASET as
OFSDATASET$IndependentVariable1, OFSDATASET$IndependentVariable2,
OFSDATASET$DependentVariable.

Data Handling

It is highly recommended that data required from the database should be pulled through the framework provided
mechanism, that is, using dataset and variables and not with any explicit DB connections. This ensures proper
security, authenticity, and auditing.

Auditing is enabled in the definition windows by introducing audit trials that captures and displays the user details
and the date of creation/ modification, along with comments.

For instance, here is a sample script where data is fetched from the DB directly (not through framework). This way
of accessing the database resident data in a model is not recommended.

```
con<-dbConnect(Oracle(),"userName","password")
qry<-"select EventLoss as Y, CardType as X1, AccBalance as X2, CustSalary as X3 from
CustTable where  Default= 'Y' "
res<- dbSendQuery(con, qry)
OperationalData<-fetch(res)
dbDisconnect(con)

#Model Logic
NewRegModel<-lm(Y~X1+X2+X3, data= OperationalData)
Plot(NewRegModel)
NewRegModel
```

Here is how a script for the same purpose can be created for accessing data via framework dataset and variables.
Define 'EventLoss', 'CardType', 'AccBalance', and 'CustSalary' from the table 'CustTable' as variables in the Oracle
Financial Services Enterprise Modeling application. while defining the model, select these variables and assign them
to R objects (the R names used within the script) say, 'IndepVariable1', 'IndepVariable2', 'IndepVariable3' and
'DepVar' from the Configure Inputs window.

```
#Assuming EventLoss, CardType, AccBalance, CustSalary from #CustTable are defined as
variables in a data set, and that #dataset is selected for variable assignments to the
respective R #variable names: Y, X1, X2, and X3.

NewRegModel<-lm(DepVar ~ IndepVariable1+ IndepVariable2+ IndepVariable3)

#The variables are made directly available to the R
#environment by the framework

Plot(NewRegModel)
NewRegModel
```

Logging

R processing log that captures script processing information, any warnings, errors or exceptions from the script, gets generated in the database server at '$ORACLE_HOME/dbs', since the R executable runs completely on the database server. Purging of the files is recommended at a regular basis.

Fitting and Prediction Using R Script

In order to support fitting and prediction using R script, the output objects can be stored in datastore using ore.save() during fitting and retrieve it using ore.load() during prediction.

ore.save() saves an R object or a list of R objects to the specified datastore in the current user's schema of the connected database.

ore.load() loads all of the R objects stored in a specified datastore from the current user's schema of the connected database.

For more information, see the *Oracle® R Enterprise User's Guide* (E36761-08).

The approach to use ore.save() and ore.load() is explained with the following example:

R scripting

Fitting:

```
fit<-lm(y~x)
ore.save(fit, name='regression_fit', overwrite=TRUE)
```

`ore.save()` will store the R object 'fit' in the datastore with the name 'regression_fit'. overwrite is a logical value specifying whether to overwrite the datastore if it already exists; the default is FALSE.

The overwrite flag is set to true to make sure that ore.save() does not fail. This is one way of doing it; otherwise the user's script should check for the existence of the datastore object.

Prediction:

```
ore.load('regression_fit')
predict<-predict(fit)
```

`ore.load()` is going to load all the R objects in the datastore 'regression_fit'.

From the datastore user should know the R object name to be used in the predict call.

ORE scripting

The approach is same as explained above, but the **Is ORE implementation used?** checkbox to be checked.

Fitting:

```
ore_fit<-ore.lm(y~x, data=OFSDATASET)
ore.save(ore_fit, name='ore_regression_fit', overwrite=TRUE)
```

Prediction:

```
ore.load('ore_regression_fit')
predict<-ore.predict(ore_fit,  newdata=OFSDATASET)
```

APPENDIX B   *User Groups*

User Groups

Table 9. User Groups

User Group Name	User Group Description
Modeling Administrator	User mapped to this group will have access to all the menu items for Enterprise modeling and will be have authorization rights for sandbox population, model deployment and modeling technique authorization.
Modeler Group	User mapped to this group will have access to all the menu items for Enterprise modeling but will not have authorization rights for sandbox population, model deployment and modeling technique authorization.

User Group - Role Mapping

Table 10. User Group to Role Mapping

Group Name	Role Name
Modeler	Alias Access
Modeler	Alias Read Only
Modeler	Alias Write
Modeler	Atomic excel upload write
Modeler	BMM Hierarchy Access
Modeler	BMM Hierarchy Read Only
Modeler	BMM Hierarchy Write
Modeler	BMM Processor Access
Modeler	BMM Processor Read Only
Modeler	BMM Processor Write
Modeler	Batch Access
Modeler	Batch Read Only
Modeler	Batch Write
Modeler	Config excel advanced
Modeler	DEFQ Manager
Modeler	DEFQ write
Modeler	DI Write
Modeler	DMM Write
Modeler	DQ Access
Modeler	DQ Advanced
Modeler	DQ Read
Modeler	DQ Write
Modeler	DT Write
Modeler	Dataset Access
Modeler	Dataset Read Only
Modeler	Dataset Write
Modeler	Derived Entity Access
Modeler	Derived Entity Read Only
Modeler	Derived Entity Write
Modeler	Dimension Access
Modeler	Dimension Read Only
Modeler	Dimension Write
Modeler	ETL Analyst
Modeler	Essbase Cube Access
Modeler	Essbase Cube Read Only
Modeler	Essbase Cube Write
Modeler	MDB Write
Modeler	MF Access

Table 10. User Group to Role Mapping

Group Name	Role Name
Modeler	MF Advanced
Modeler	MF Read
Modeler	MF Write
Modeler	Manage Run Access
Modeler	Manage Run Read Only
Modeler	Manage Run Write
Modeler	Measure Access
Modeler	Measure Read Only
Modeler	Measure Write
Modeler	Obj Migration Access
Modeler	Obj Migration Read
Modeler	ObjectAdmin advanced
Modeler	Oracle Cube Access
Modeler	Oracle Cube Read Only
Modeler	Oracle Cube Write
Modeler	Process Access
Modeler	Process Read Only
Modeler	Process Write
Modeler	Rule Access
Modeler	Rule Read Only
Modeler	Rule Write
Modeler	Run Access
Modeler	Run Read Only
Modeler	Run Write
Modeler	STF Access
Modeler	STF Advanced
Modeler	STF Read
Modeler	STF Write
Modeling Administrator	Alias Access
Modeling Administrator	Alias Authorize
Modeling Administrator	Alias Read Only
Modeling Administrator	Alias Write
Modeling Administrator	Atomic excel advanced
Modeling Administrator	Atomic excel upload write
Modeling Administrator	BMM Hierarchy Access
Modeling Administrator	BMM Hierarchy Authorize
Modeling Administrator	BMM Hierarchy Read Only
Modeling Administrator	BMM Hierarchy Write
Modeling Administrator	BMM Processor Access
Modeling Administrator	BMM Processor Authorize

Table 10. User Group to Role Mapping

Group Name	Role Name
Modeling Administrator	BMM Processor Read Only
Modeling Administrator	BMM Processor Write
Modeling Administrator	Batch Access
Modeling Administrator	Batch Advanced
Modeling Administrator	Batch Authorize
Modeling Administrator	Batch Phantom
Modeling Administrator	Batch Read Only
Modeling Administrator	Batch Write
Modeling Administrator	Config excel advanced
Modeling Administrator	DEFQ Manager
Modeling Administrator	DEFQ access
Modeling Administrator	DEFQ advanced
Modeling Administrator	DEFQ authorize
Modeling Administrator	DI Write
Modeling Administrator	DMM Write
Modeling Administrator	DQ Access
Modeling Administrator	DQ Advanced
Modeling Administrator	DQ Authorize
Modeling Administrator	DQ Read
Modeling Administrator	DQ Write
Modeling Administrator	DT Write
Modeling Administrator	Dataset Access
Modeling Administrator	Dataset Authorize
Modeling Administrator	Dataset Read Only
Modeling Administrator	Dataset Write
Modeling Administrator	Derived Entity Access
Modeling Administrator	Derived Entity Authorize
Modeling Administrator	Derived Entity Read Only
Modeling Administrator	Derived Entity Write
Modeling Administrator	Dimension Access
Modeling Administrator	Dimension Authorize
Modeling Administrator	Dimension Read Only
Modeling Administrator	Dimension Write
Modeling Administrator	ETL Analyst
Modeling Administrator	Essbase Cube Access
Modeling Administrator	Essbase Cube Authorize
Modeling Administrator	Essbase Cube Read Only
Modeling Administrator	Essbase Cube Write
Modeling Administrator	MDB Write
Modeling Administrator	MF Access

Table 10. User Group to Role Mapping

Group Name	Role Name
Modeling Administrator	MF Advanced
Modeling Administrator	MF Authorize
Modeling Administrator	MF Read
Modeling Administrator	MF Write
Modeling Administrator	Manage Run Access
Modeling Administrator	Manage Run Read Only
Modeling Administrator	Manage Run Write
Modeling Administrator	Measure Access
Modeling Administrator	Measure Authorize
Modeling Administrator	Measure Read Only
Modeling Administrator	Measure Write
Modeling Administrator	Obj Migration Access
Modeling Administrator	Obj Migration Advanced
Modeling Administrator	Obj Migration Read
Modeling Administrator	Obj Migration Write
Modeling Administrator	ObjectAdmin advanced
Modeling Administrator	Oracle Cube Access
Modeling Administrator	Oracle Cube Authorize
Modeling Administrator	Oracle Cube Read Only
Modeling Administrator	Oracle Cube Write
Modeling Administrator	PR2 Administrator
Modeling Administrator	Process Access
Modeling Administrator	Process Advanced
Modeling Administrator	Process Authorize
Modeling Administrator	Process Read Only
Modeling Administrator	Process Write
Modeling Administrator	Publish Metadata
Modeling Administrator	Rule Access
Modeling Administrator	Rule Advanced
Modeling Administrator	Rule Authorize
Modeling Administrator	Rule Read Only
Modeling Administrator	Rule Write
Modeling Administrator	Run Access
Modeling Administrator	Run Advanced
Modeling Administrator	Run Authorize
Modeling Administrator	Run Read Only
Modeling Administrator	Run Write
Modeling Administrator	STF Access
Modeling Administrator	STF Advanced

Table 10. User Group to Role Mapping

Group Name	Role Name
Modeling Administrator	STF Read
Modeling Administrator	STF Write

APPENDIX C   *EMF Object Migration*

This section presents information related to EMF Object Migration prerequisites. For detailed information on how to migrate objects, see the *Command Line Utility* section in the Oracle Financial Services Analytical Infrastructure Applications User Guide.

Prerequisites

For Modeling Framework and Stress Testing objects:

- Migrate the Sandbox before you migrate your Models. This is important since you can migrate Models to the same Sandbox across environments.

- The availability of the application with same APP_ID in source and target environment is a prerequisite for Sandbox Migration.

- If the "Data model upload required" checkbox is selected for the sandbox definition, the infodom code provided in SANDBOXINFODOM attribute should be a clean infodom where no model upload is done.

- Do not perform Technique migration between infodoms, it is not allowed. However, you can perform Technique migration between environments.

- Create Data Source for Sandbox Infodom to migrate R or ORE based models.

Note:
Dependency migration of models is not supported. Since the dependent metadata for a model can be in the production infodom, a user who is granted rights to the sandbox may not have rights to the production infodom to create these dependent metadata. This would create a conflict and hence dependency migration of models cannot be done.

Appendix C

Oracle Financial Services Enterprise Modeling application
8.0.8.0.0 User Guide

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
www.oracle.com/us/industries/financial-services/

ORACLE®