Oracle® Banking Party Management

Administrator Guide

Release 2.7.0.0.0

F11757-01

March 2019

ORACLE®

Oracle Banking Party Management Administrator Guide, Release 2.7.0.0.0

F11757-01

Contents

List of Figures

List of Tables

Preface

This guide describes how to administer the Oracle Banking Party Management applications environment, including user administration, batch execution, application monitoring, and bank and branch setup.

Oracle recommends that you review its contents before installing, or working with the product.

This preface contains the following topics:

Audience

This guide is intended for the administrators of Oracle Banking Party Management.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at

http://www.oracle.com/us/corporate/accessibility/index.html

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/us/corporate/accessibility/support/index.html#info or visit http://www.oracle.com/us/corporate/accessibility/support/index.html#trs if you are hearing impaired.

Organization of the Guide

This document contains:

Chapter 1 Users Administration

This chapter describes all user management related activities to be performed by an administrator for Oracle Banking Party Management.

Chapter 2 Approvals Management

This chapter describes worklist authorization related activities to be performed as an administrator.

Chapter 3 Defining Task Configuration Rules

This chapter describes various configurations that can be done for human tasks.

Chapter 4 Data Management

This chapter describes data related activities to be performed as an administrator.

Chapter 5 Setting Up The Bank And Branch

This chapter provides the process of setting up the bank and the branch commonly referred to as the Day 0 setups.

Chapter 6 Application Monitoring Using Administration Application

This chapter provides an overview on the various monitoring operations performed as an administrator using application screens.

Chapter 7 Application Monitoring Using EM Plugin

This chapter provides an overview on the various monitoring operations performed as an administrator, using Enterprise Manger (EM) Plugin.

Chapter 8 Configuration Export-Import Operations

This chapter gives an insight to the Configuration Export-Import operations.

Chapter 9 Batch Shells in OBPM

This chapter describes the batch shells used in Oracle Banking Party Management and their execution sequence.

Chapter 10 Information Lifecycle Management (ILM)

This chapter describes the configuration, installation, and policy setup of Information Lifecycle Management (ILM).

Chapter 11 Transparent Data Encryption (TDE)

This chapter describes the configuration, installation, and policy setup of Transparent Data Encryption (TDE).

Chapter 12 Masking Customer Private Data

This chapter describes the configuration, installation, and policy setup to mask customer private data categories as sensitive or Personally Identifiable Information (PII).

Chapter 13 Configure ODI for Inbound Document Upload

This chapter provides the steps to configure ODI for Inbound Document Upload

Chapter 14 Additional Recommendations

This chapter provides specific recommendations to be considered for implementation:

Related Documents

For more information, see the following documentation:

- For installation and configuration information, see the Oracle Banking Party Management Installation Guide - Silent Installation.
- For a comprehensive overview of security, see the Oracle Banking Party Management Security Guide.
- For the complete list of Oracle Banking licensed products and the Third Party licenses included with the license, see the Oracle Banking Party Management Licensing Guide.
- For information related to customization and extension, see the Oracle Banking Party Management Extensibility Guides for Host, SOA, and UI.
- For information on the functionality and features, see the respective Oracle Banking Party Management Functional Overview document.
- For recommendations of secure usage of extensible components, see the Oracle Banking Party Management Secure Development Guide.

Conventions

The following text conventions are used in this document:

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

1 Users Administration

This chapter describes all user management related activities to be performed by an administrator for the application.

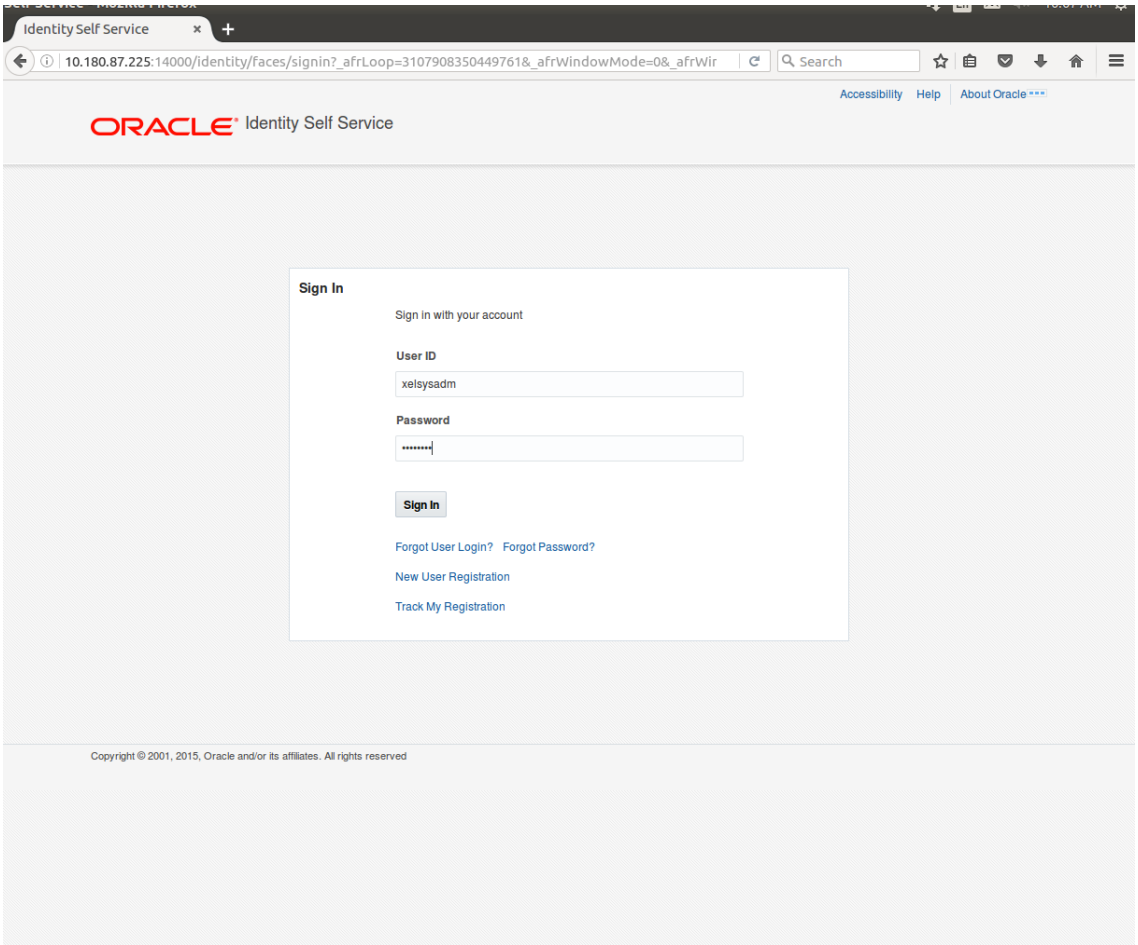
1.1 Creating Users in Oracle Identity Manager (OIM)

This section explains the procedure to create users in Oracle Identity Manager (OIM).

To create users in OIM:

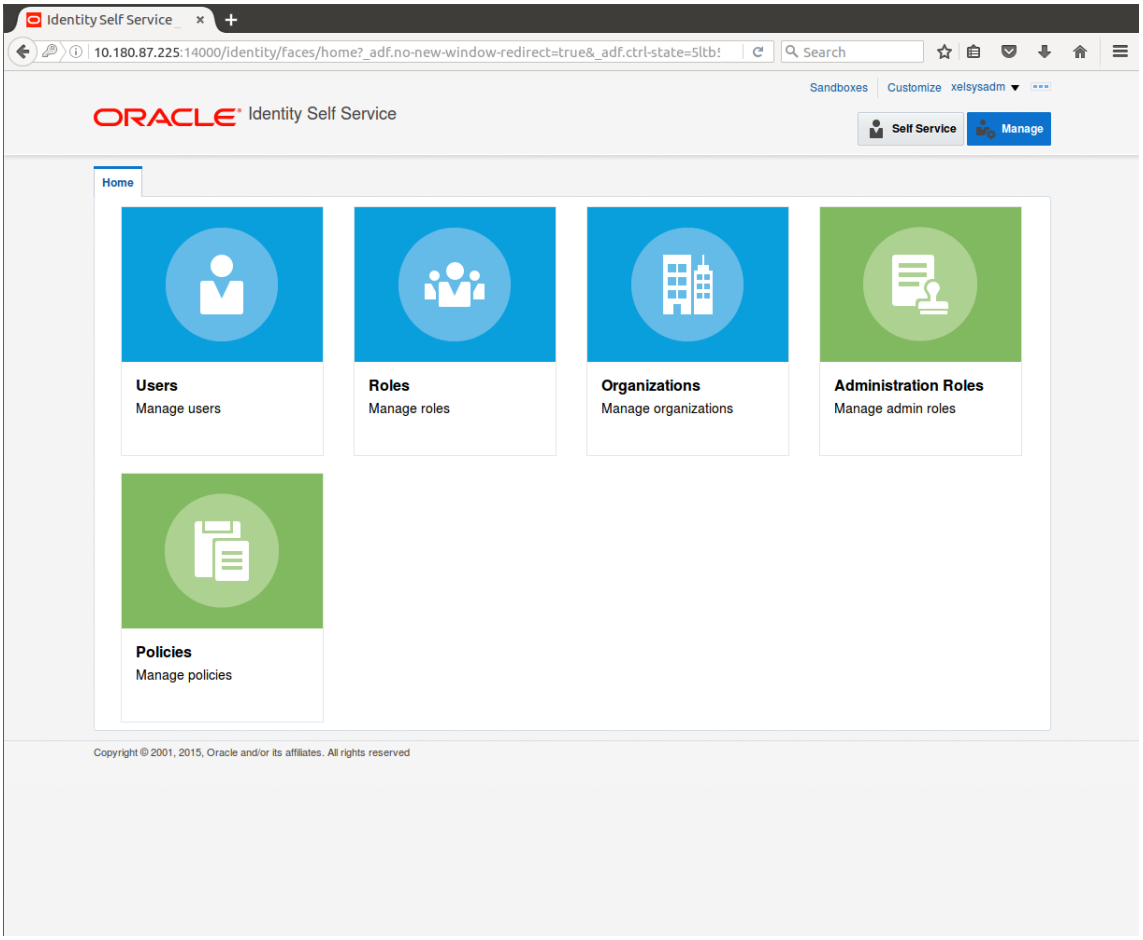
1. Log in to OIM with the User ID as **xelsysadm** and the relevant <Password>.

Figure 1–1 Creating Users in OIM - Log in



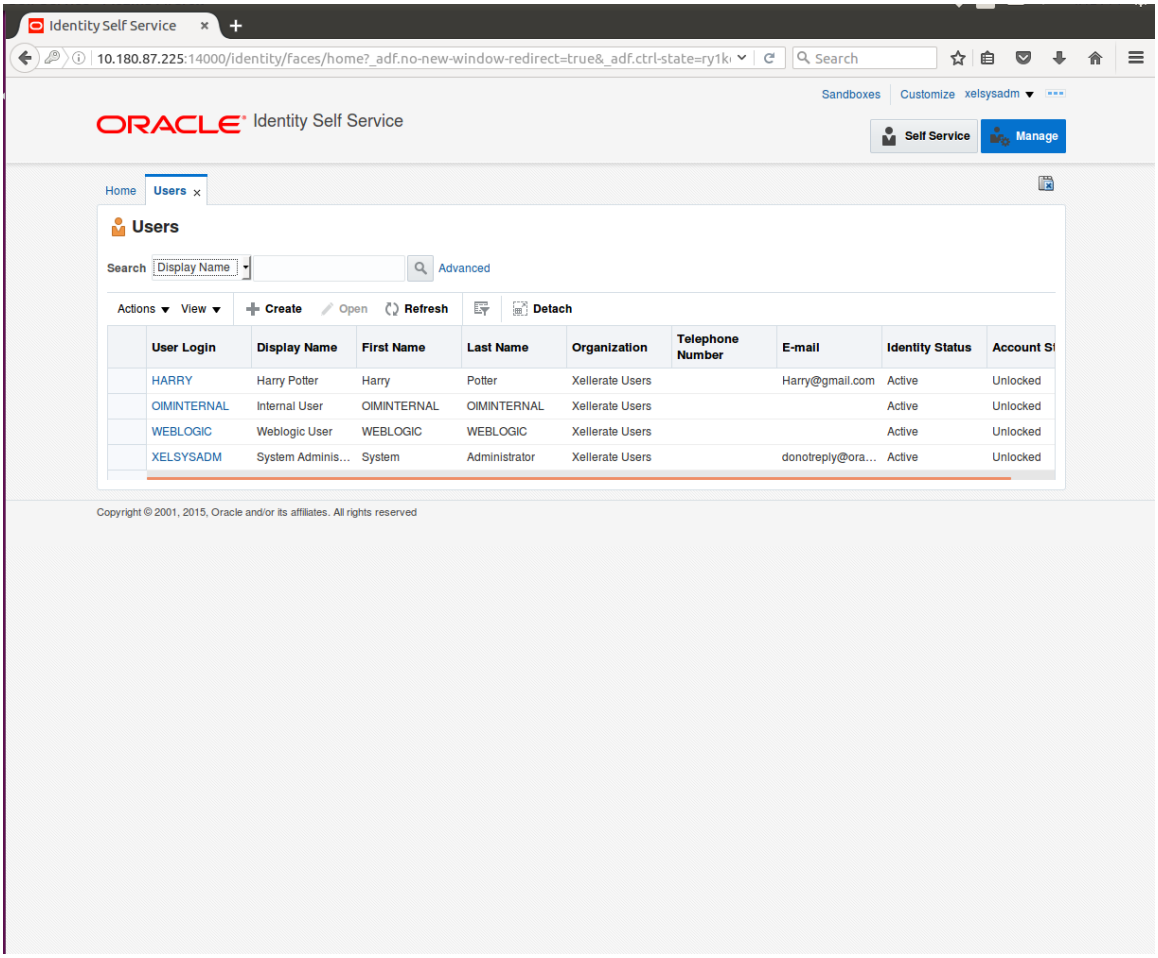
2. Click **Users** under the Manage section.

Figure 1–2 Creating Users in OIM - Manage Section



3. In the **Search Users** page, search for existing users. The Search Results appear.
4. Click **Create** in the Search Results section to create a new user.

Figure 1–3 Creating Users in OIM - Click Create



5. In the **Create User** page, enter the required user details.

Figure 1–4 Creating Users in OIM - Enter User Details

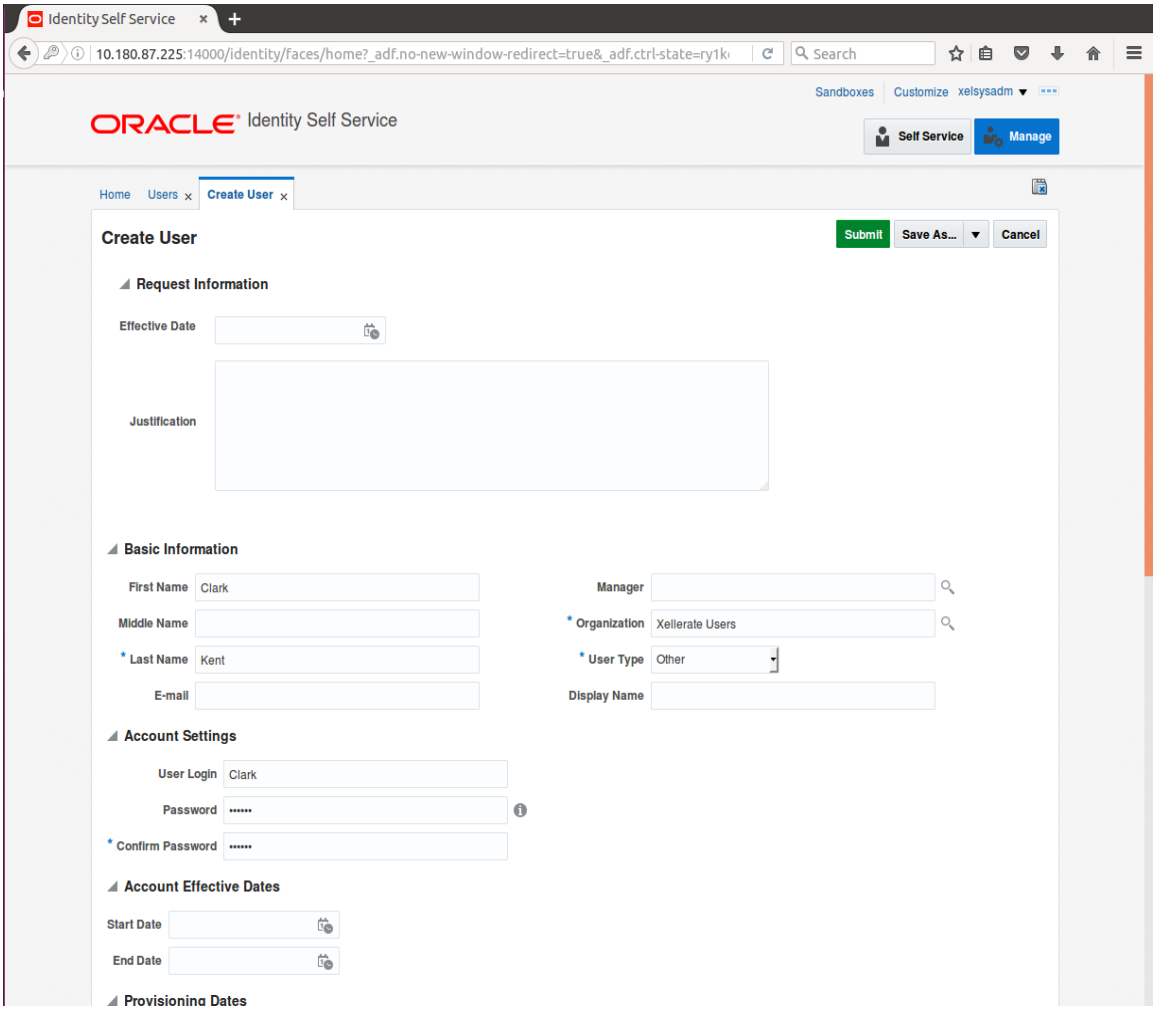
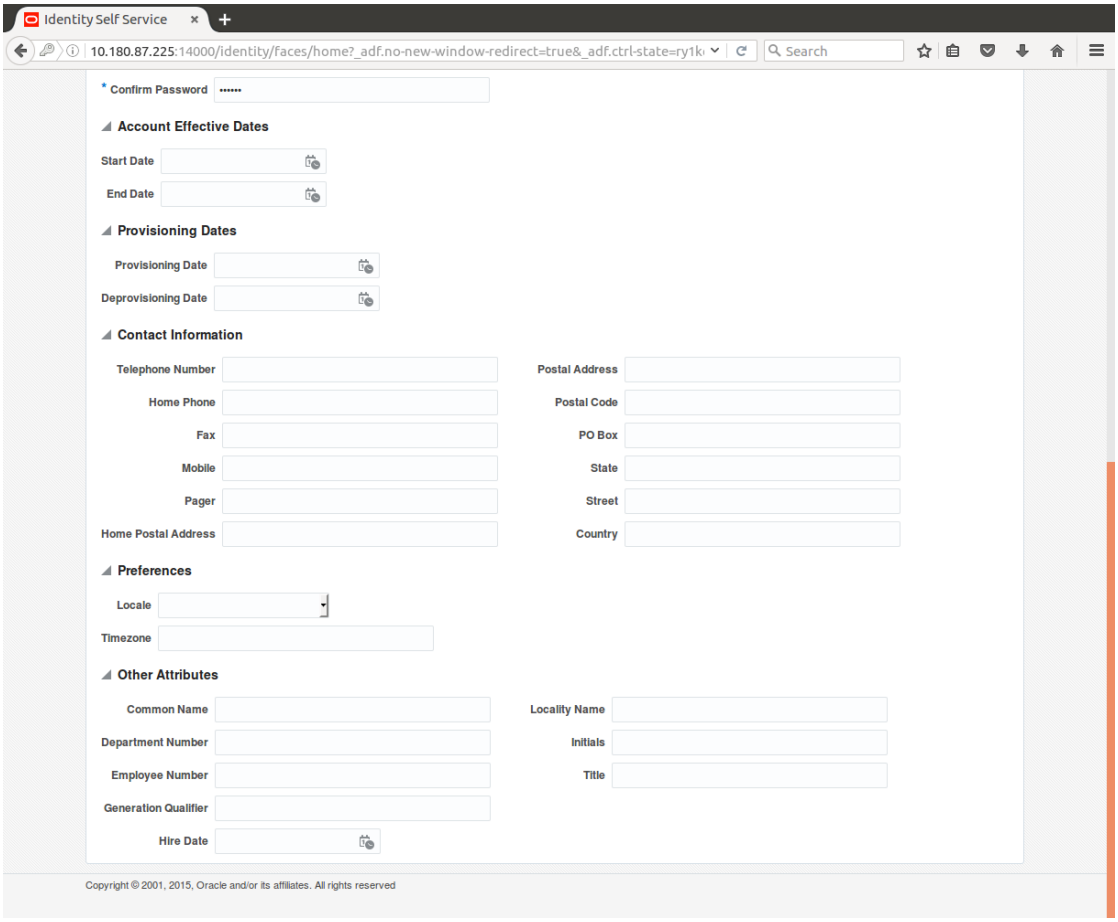


Figure 1–5 Enter User Details (Continued)



6. Click **Submit.**

   On completion of this procedure the user gets created in OIM, and gets synced in OID.

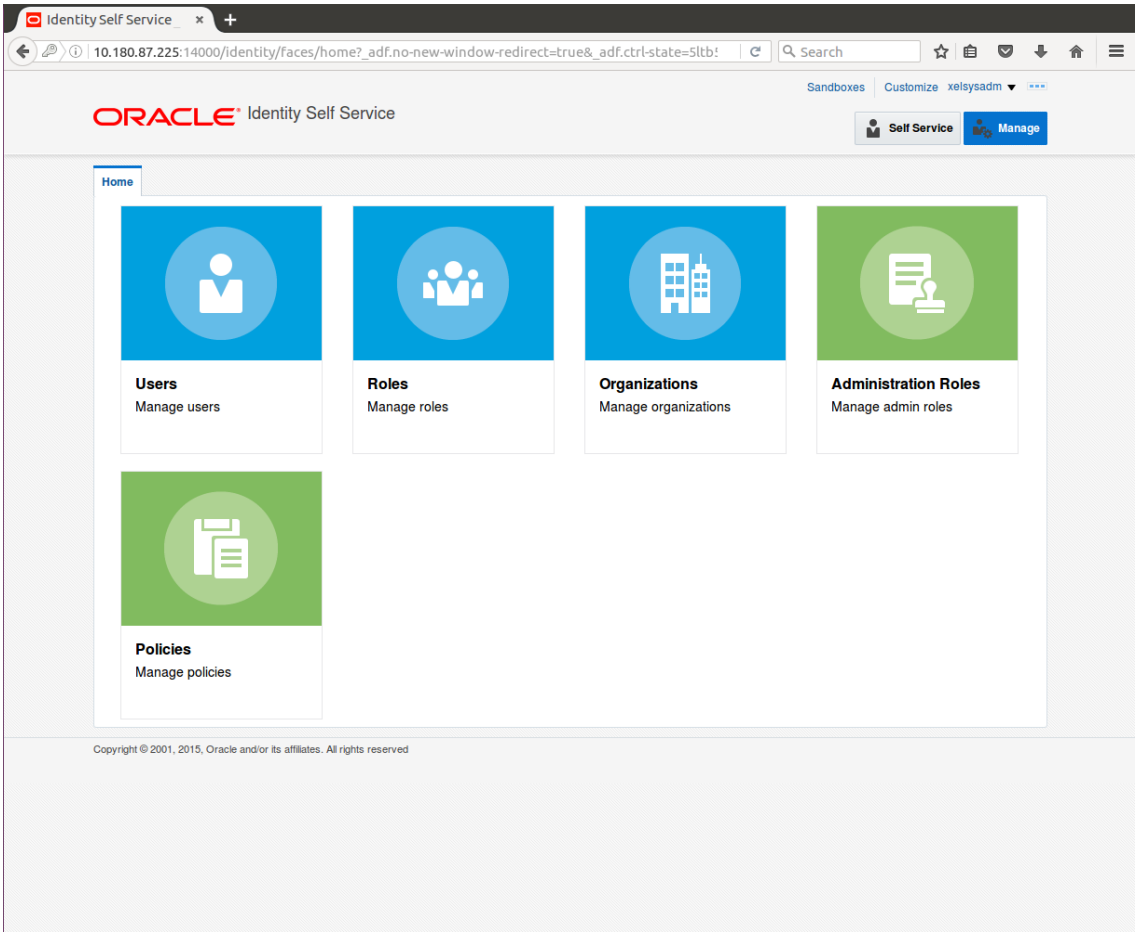
1.2 Creating Roles in Oracle Identity Manager (OIM)

This section explains the procedure to create roles in Oracle Identity Manager (OIM).

To create roles in OIM:

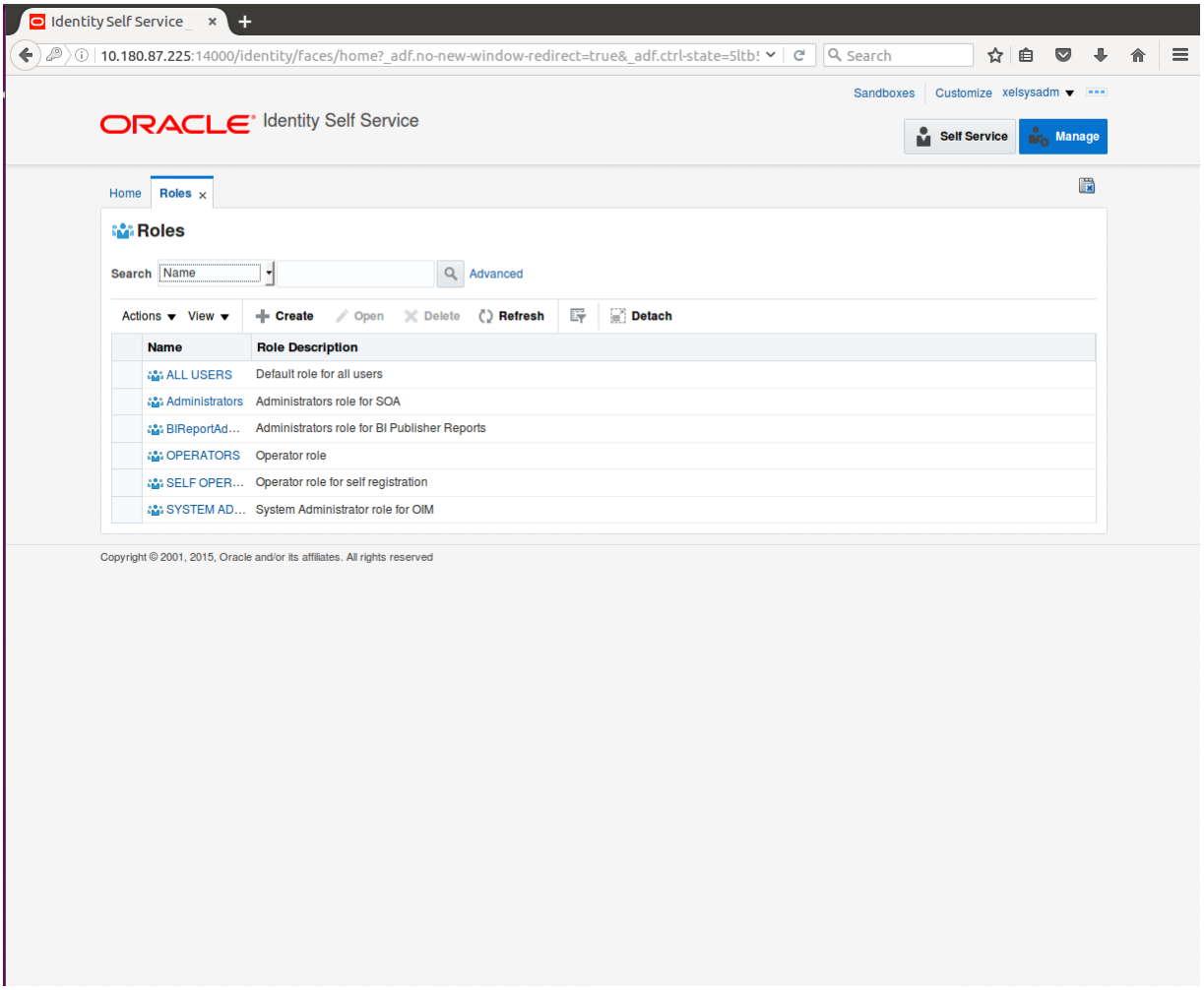
1. Click **Roles** under the Manage section.

Figure 1–6 Creating Roles in OIM - Manage Section



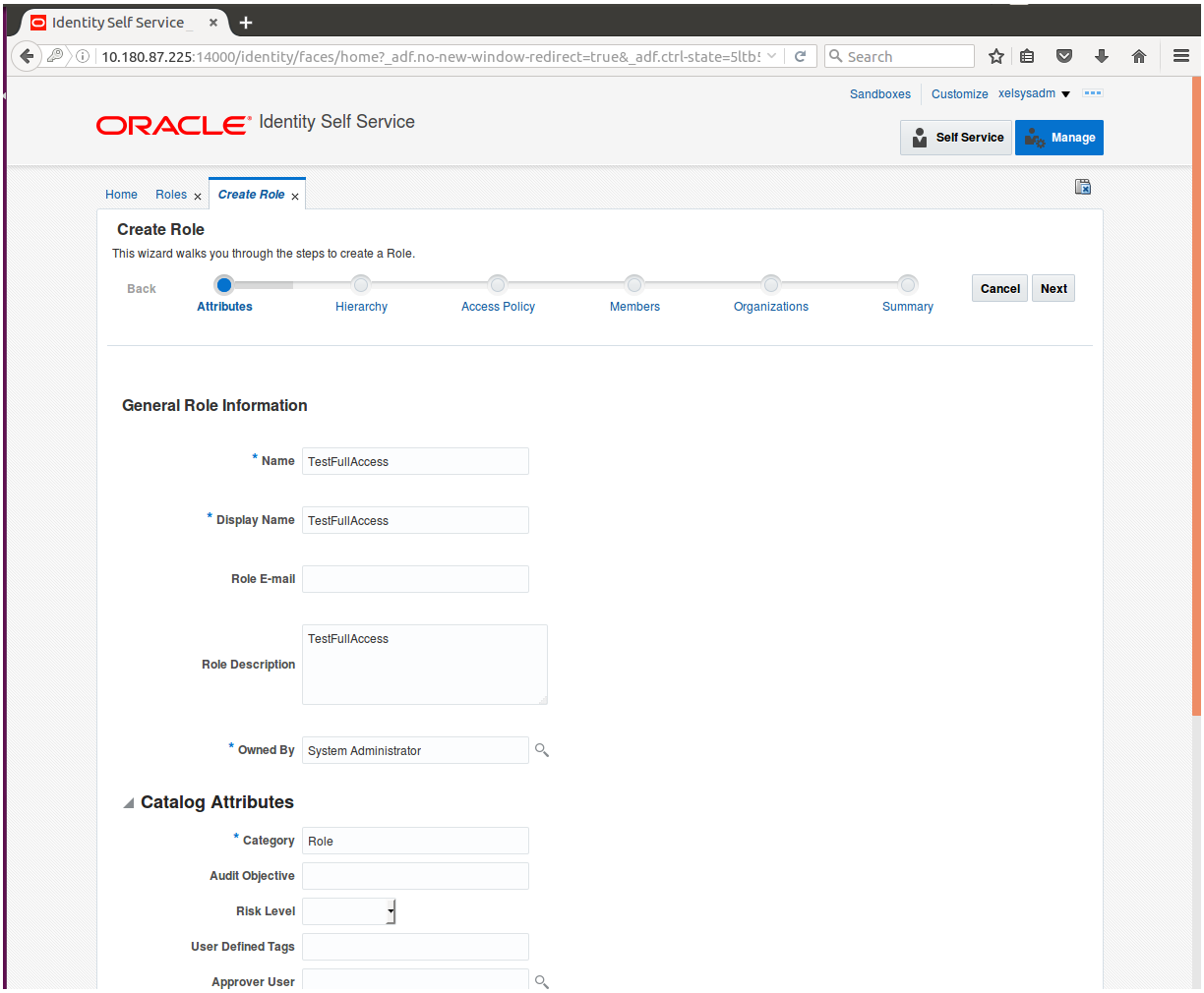
2. In the **Search Roles** page, search for existing roles. The Search Results appear.
3. Click **Create** in the Search Results section to create a new Role.

Figure 1–7 Creating Roles in OIM - Click Create



4. Fill the role details.

Figure 1–8 Creating Roles in OIM - Enter Role Details

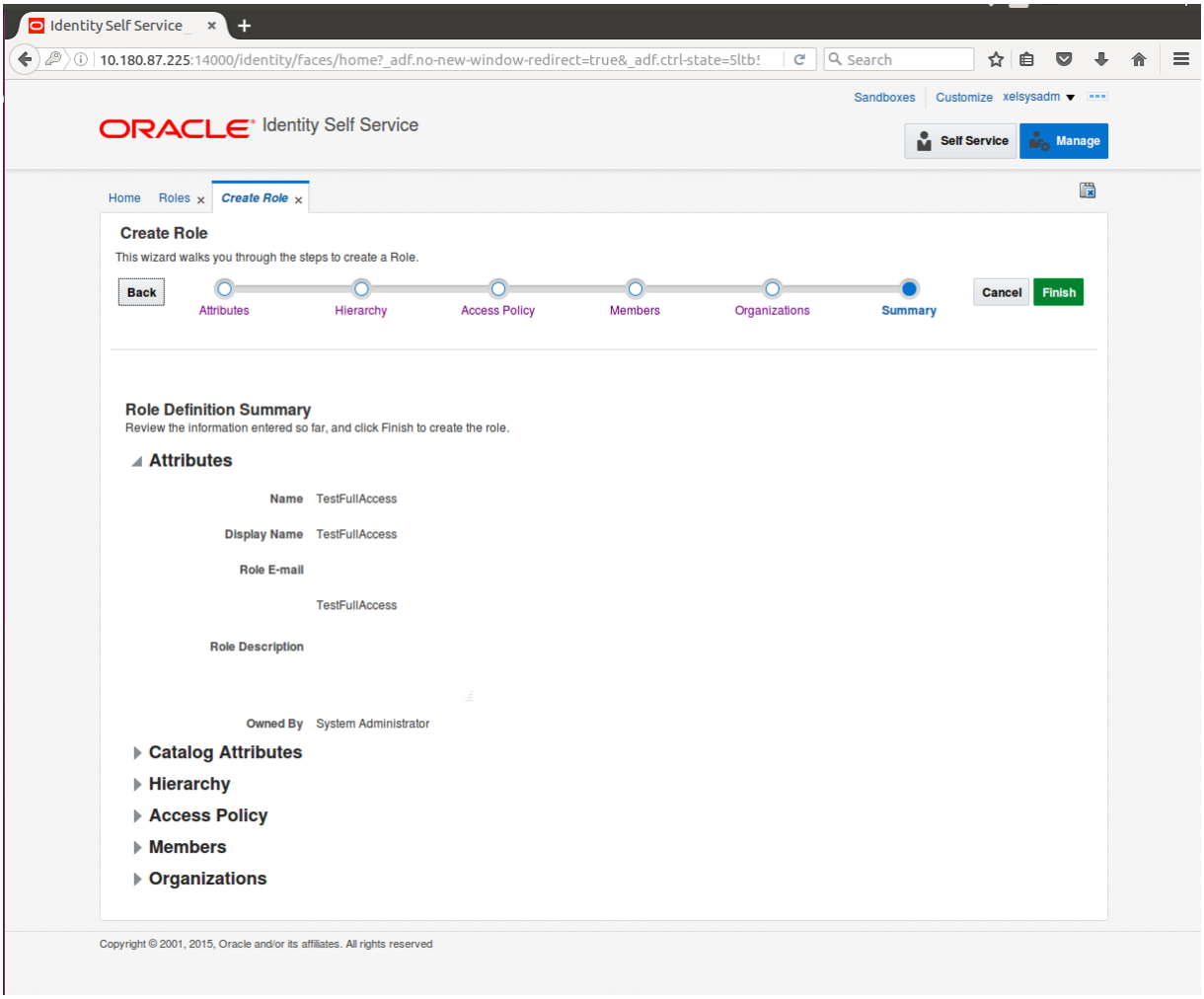


5. Click **Finish.** The role is created successfully.

   This role creates a group in OID.

   While running the PIT (Policy Import tool), the Enterprise role (OIM role or OID group in this scenario) is mapped to the Application Role in OES.

Figure 1–9 Creating Roles in OIM - Role Created Successfully



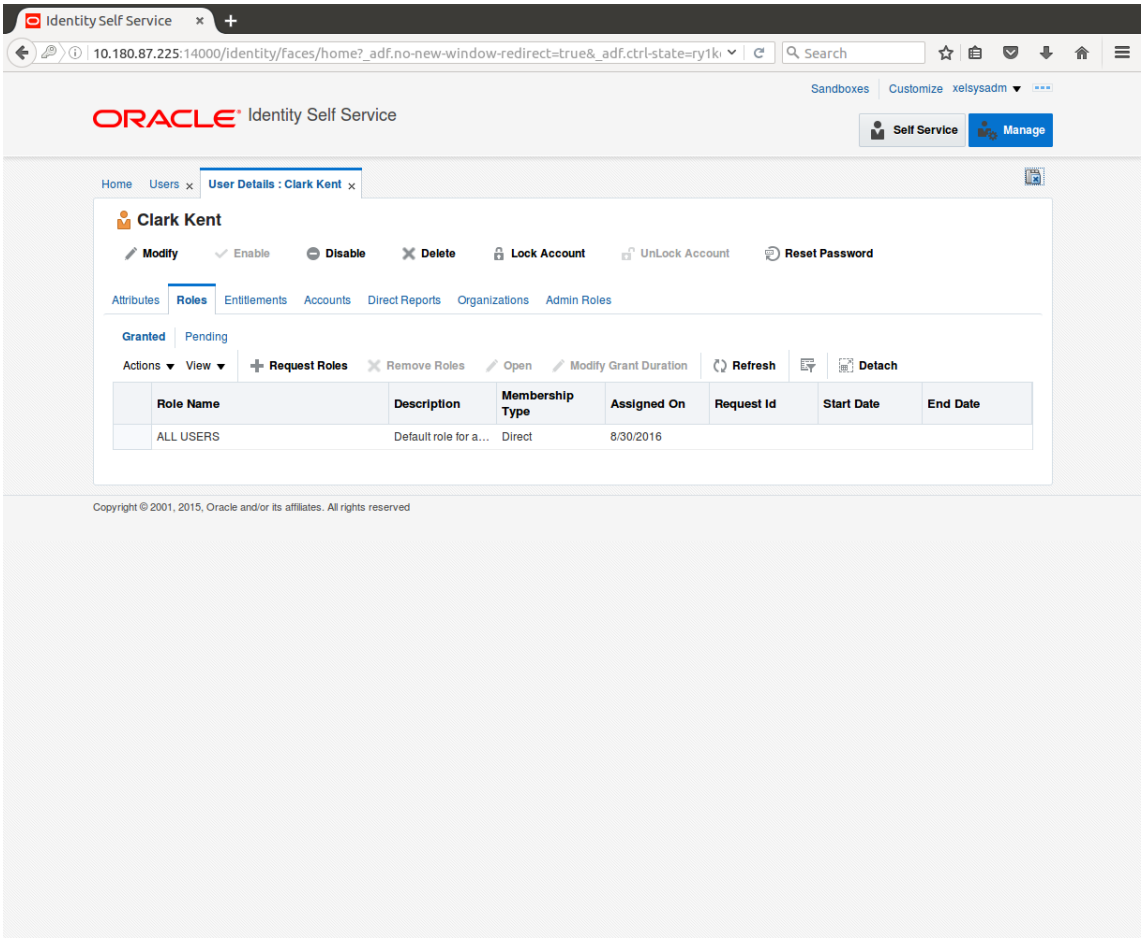
1.3 Assigning Roles to Users in OIM

This section explains how to assign roles to the user in OIM.

To assign a role to a user:

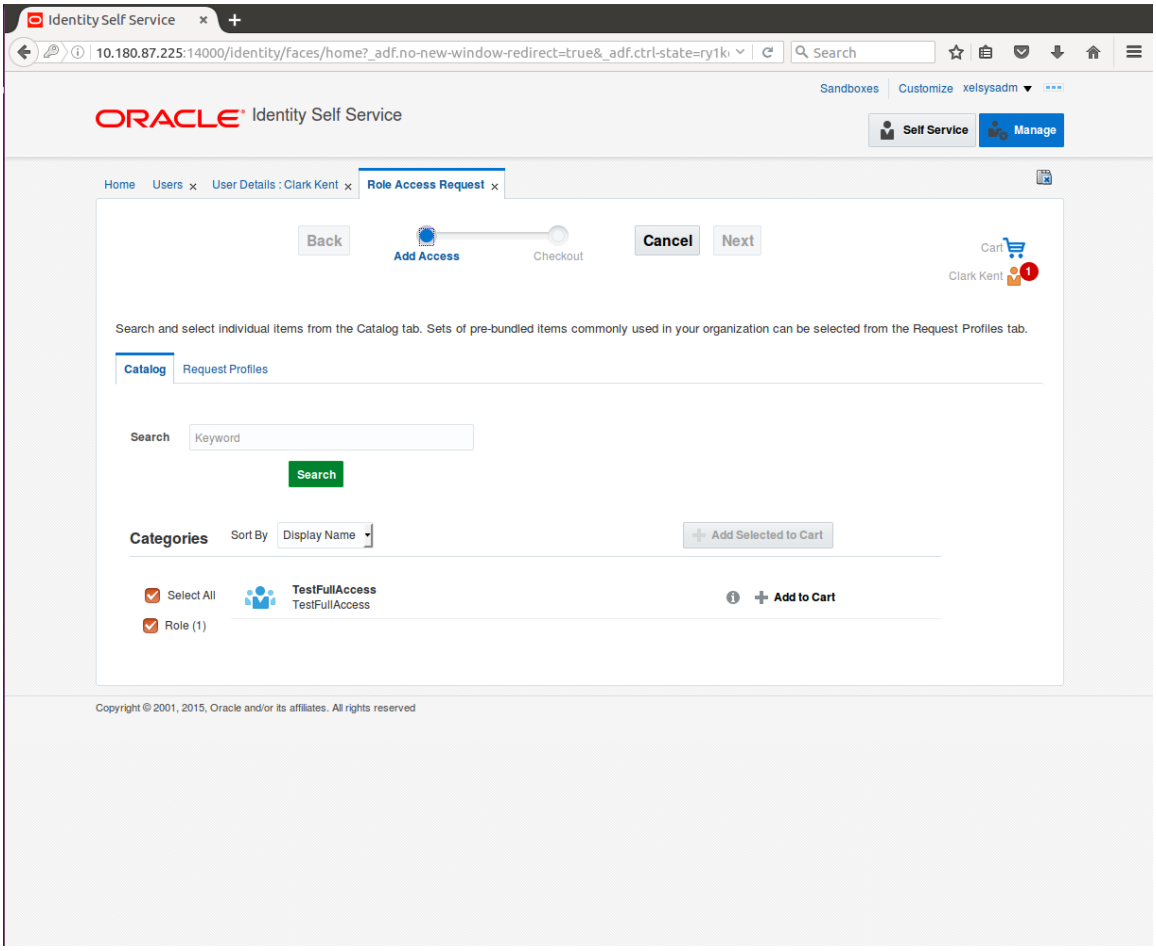
1. Log in to OIM.
2. Navigate to the **Roles Tab** under the User.
3. Click **Request Roles.**

Figure 1–10 Assigning Roles in OIM - Requesting Roles



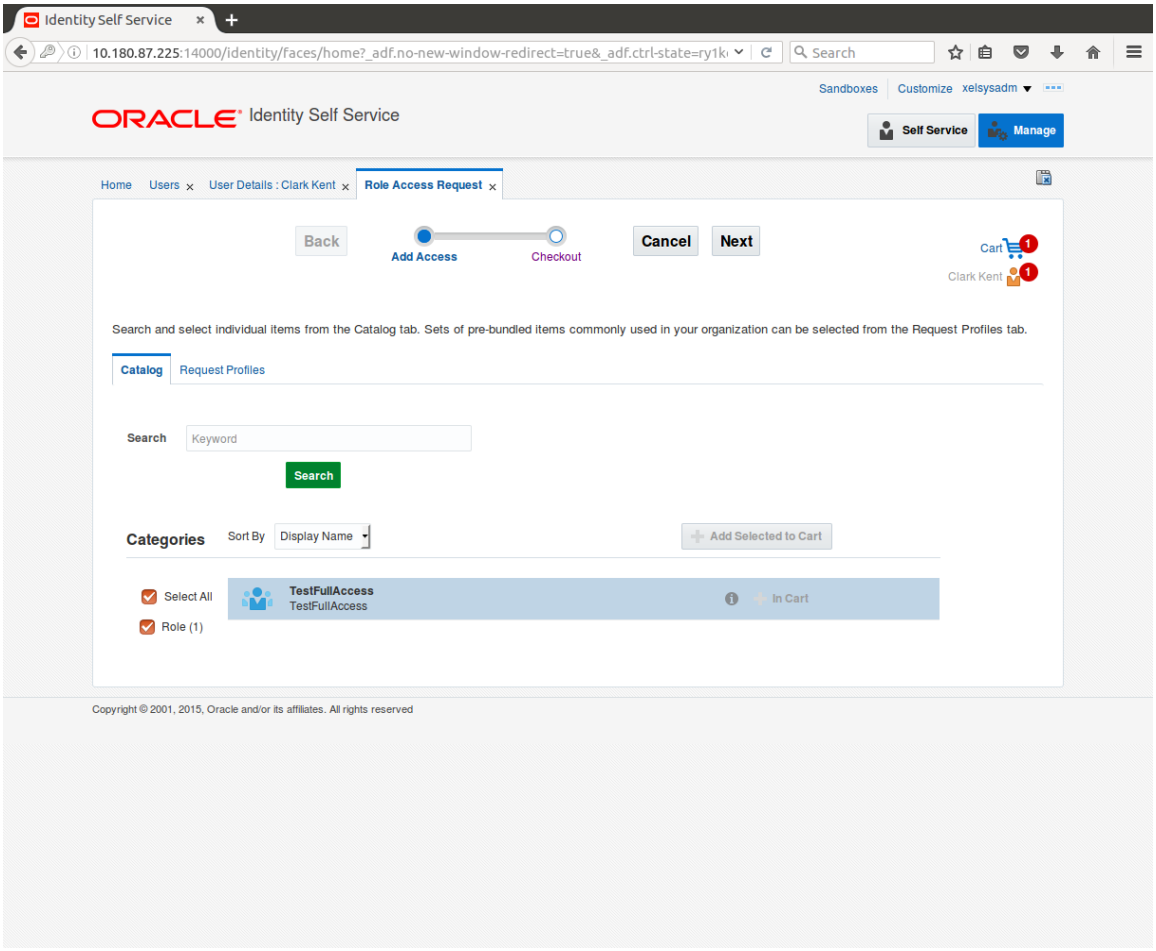
4. In the **Catalog** page, select the required role and click **Add to Cart.** The item gets added to the cart.

Figure 1–11 Assigning Roles in OIM - Adding to Cart



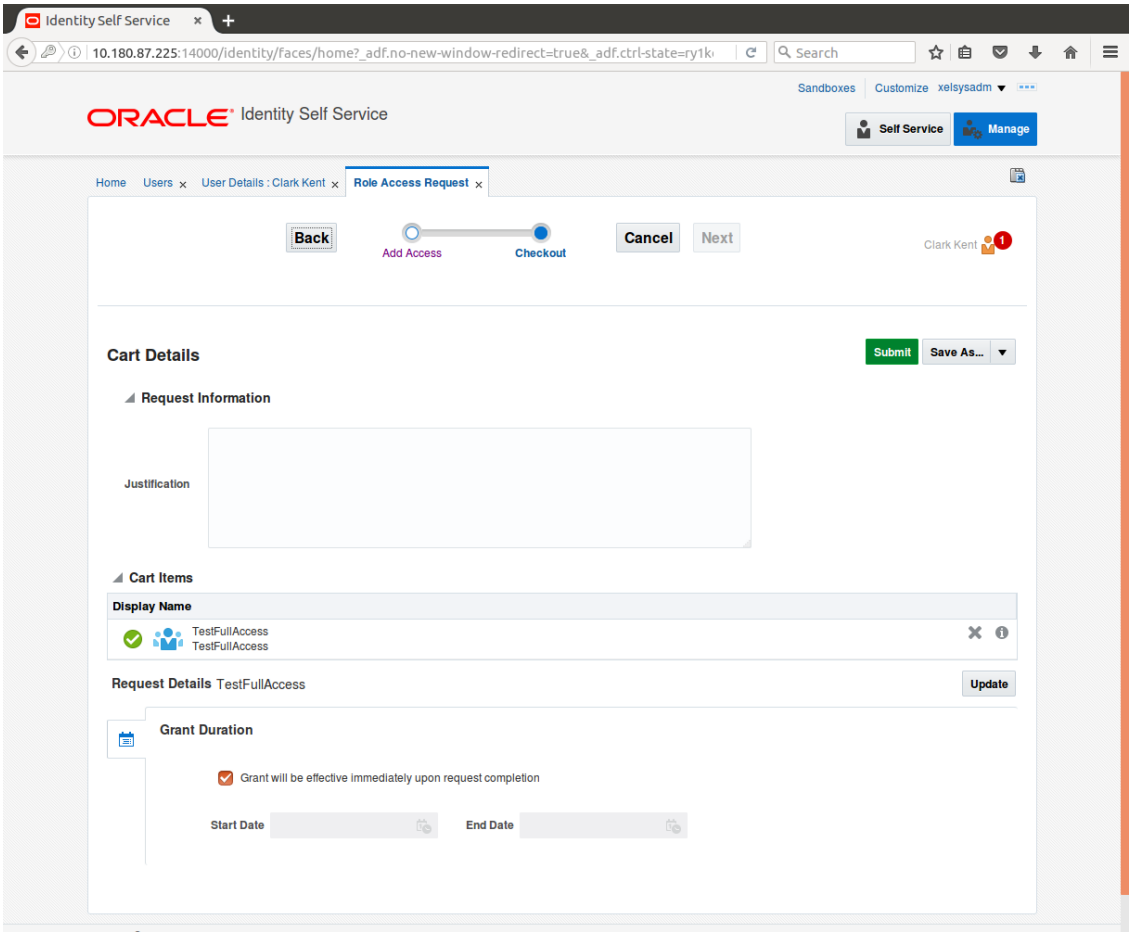
5. Click **Checkout.**

Figure 1–12 Assigning Roles in OIM - Checkout Cart



6. In the **Cart Details** page, click **Submit.**

Figure 1–13 Assigning Roles in OIM - Submit Cart



On completion of this procedure the role gets assigned to the user in OIM.

1.4 Locking Users in OIM

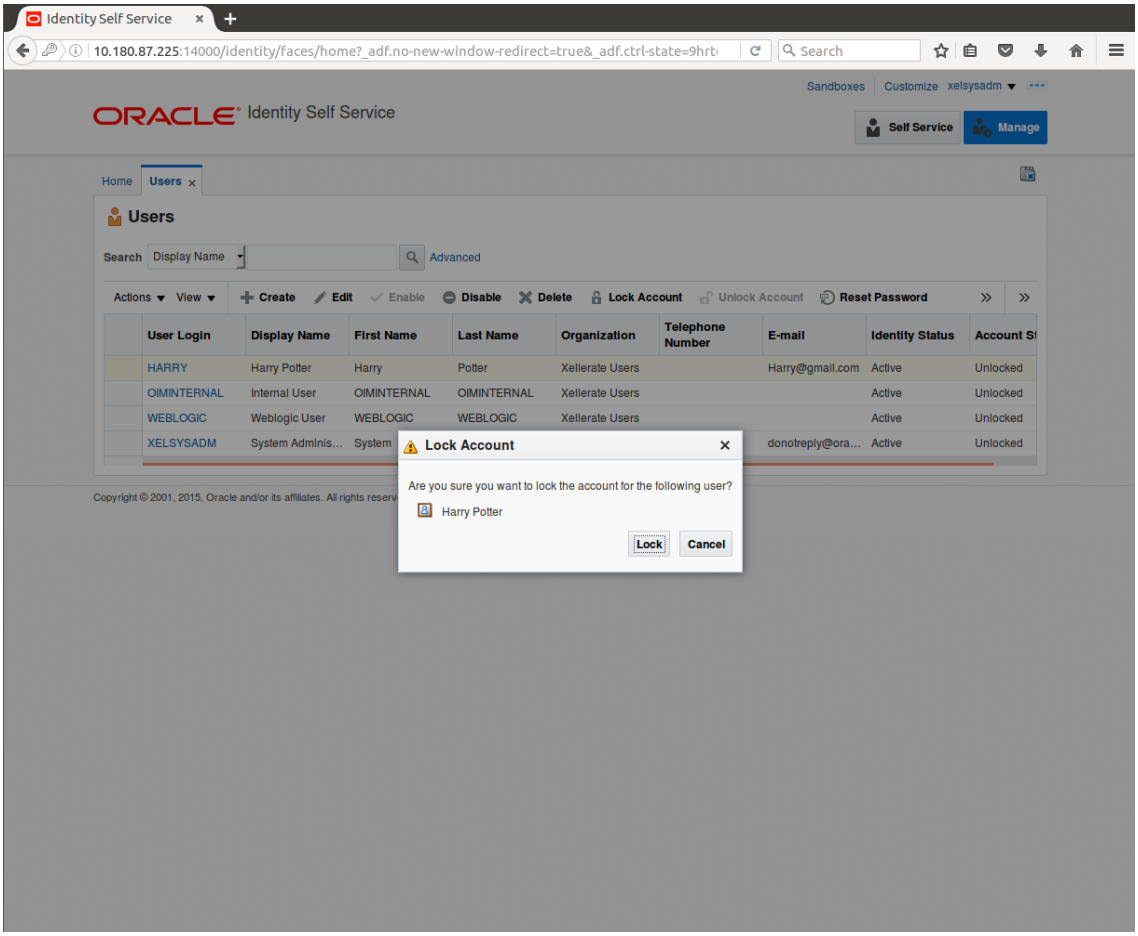
This section explains how to lock the user in OIM.

To lock a user:

1. Log in to OIM.
2. Click **Lock Account** to lock a user.

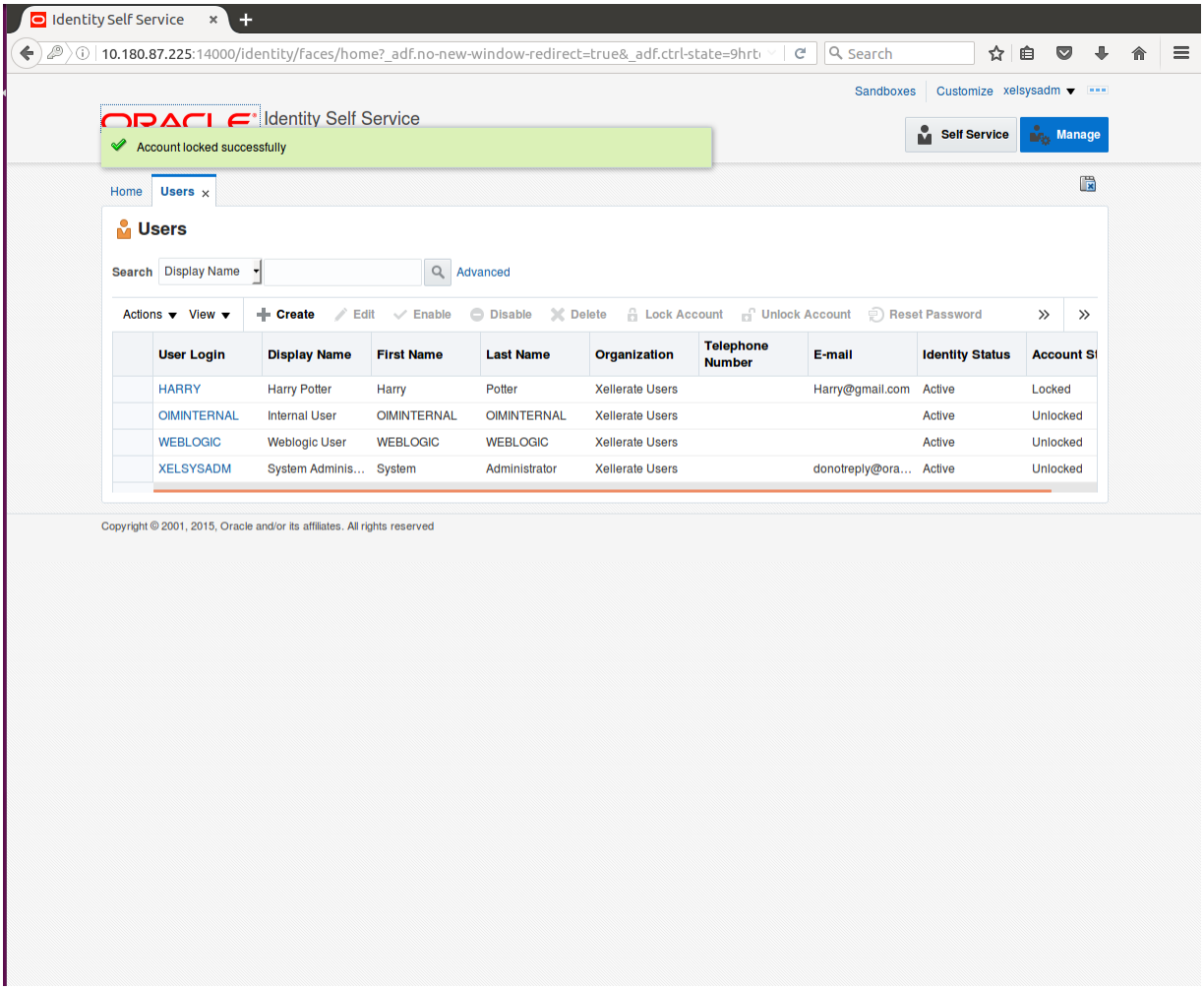
   A message appears, Are you sure you want to lock the account for the following user?
3. Click **Lock.**

Figure 1–14 Locking Users in OIM



The user is locked successfully.

Figure 1–15 User Locked Successfully



1.5 Unlocking Users in OIM

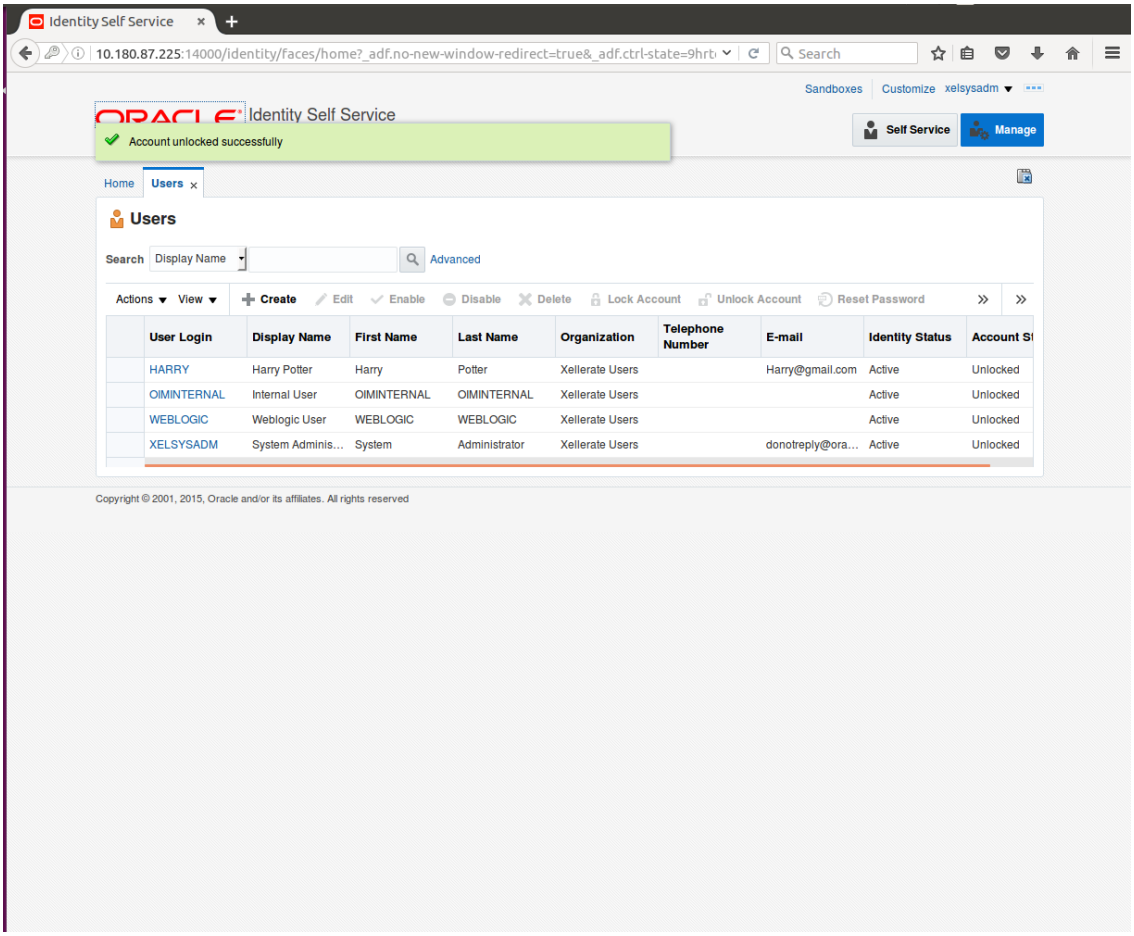
This section explains how to unlock the user in OIM.

To unlock a user:

1. Log in to OIM.
2. Click **Unlock Account** to unlock a user.

   A message appears, Are you sure you want to Unlock these users?
3. Click **Unlock.**

Figure 1–16 Unlocking Users in OIM



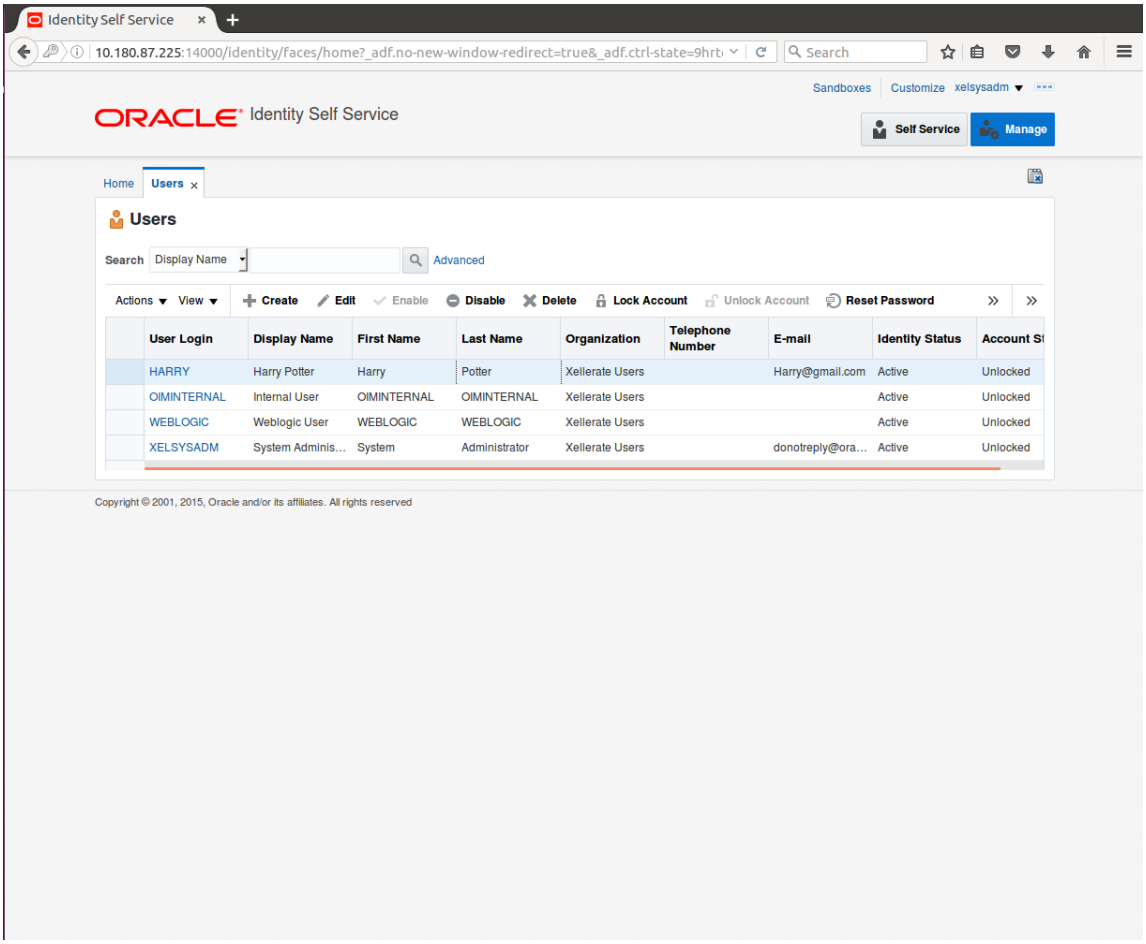
The user is unlocked successfully.

1.6 Resetting User Password in OIM

This section explains how to reset user password in OIM.

1. Log in to OIM.
2. Click **Reset Password** to reset a user password.

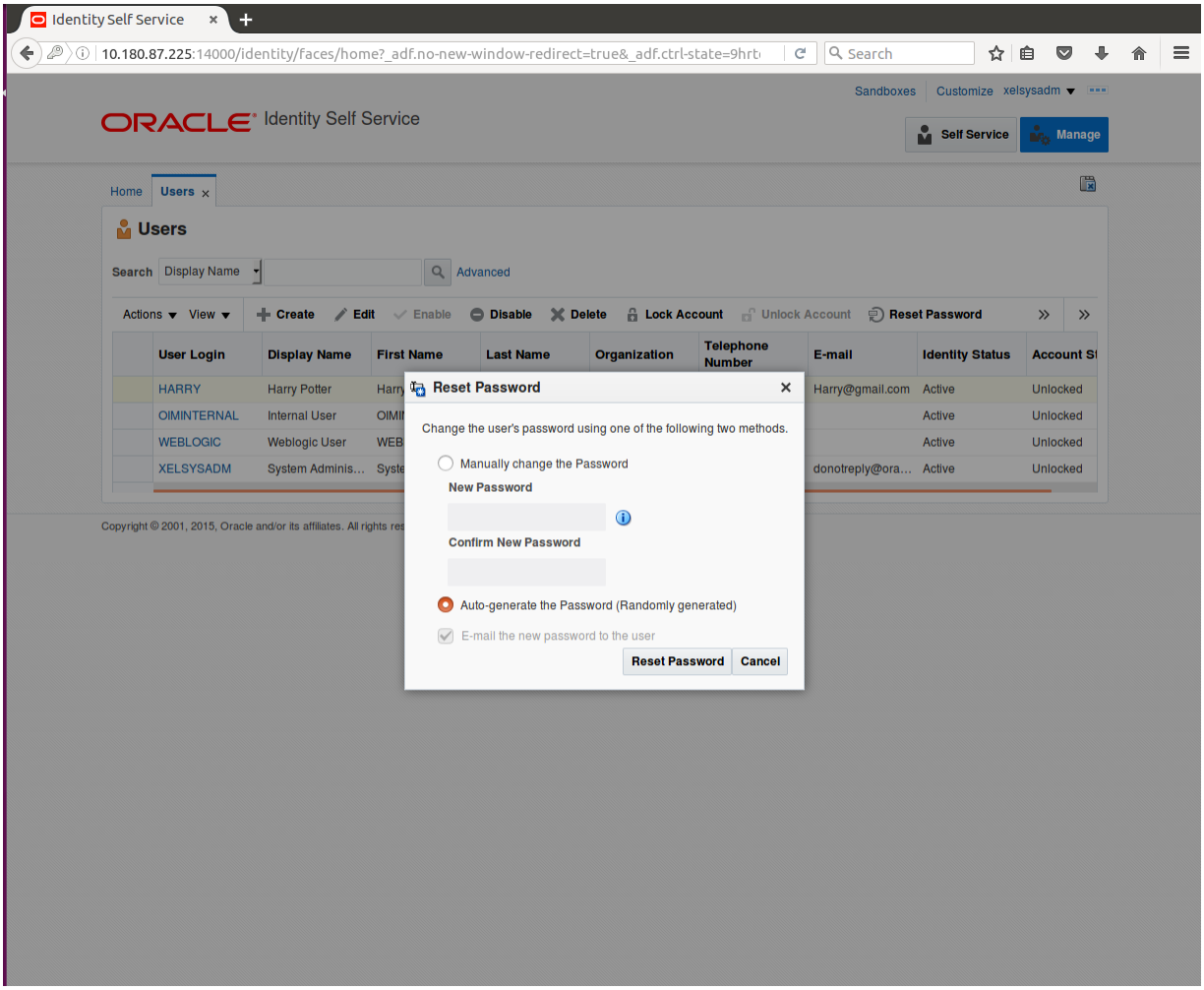
Figure 1–17 Resetting User Password in OIM



The **Reset Password** dialog box appears.

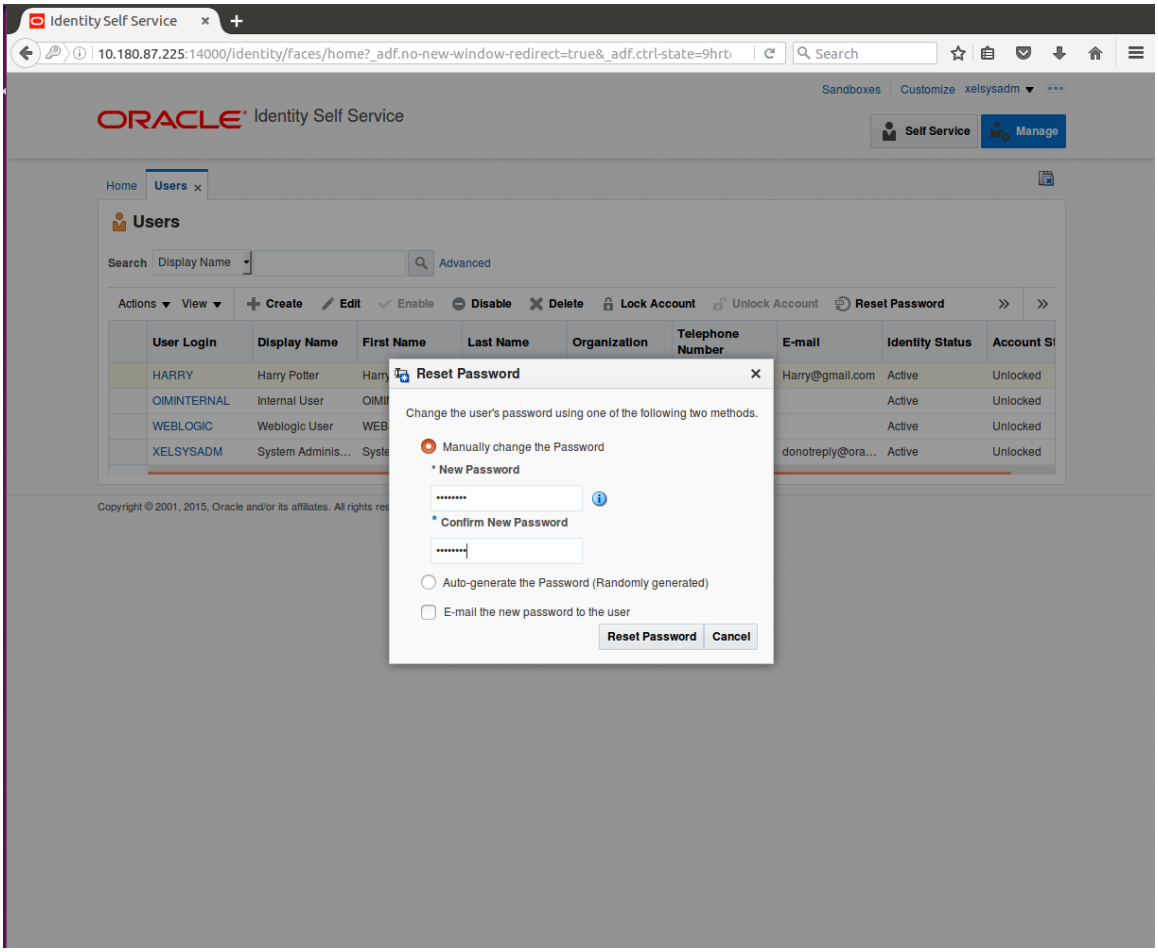
You can select either **Manually change the Password** option to change the password manually or select the **Auto-generate the password (Randomly generated)** option to enable auto generation of the password.

Figure 1–18 Resetting User Password in OIM - Manually or Auto-generate



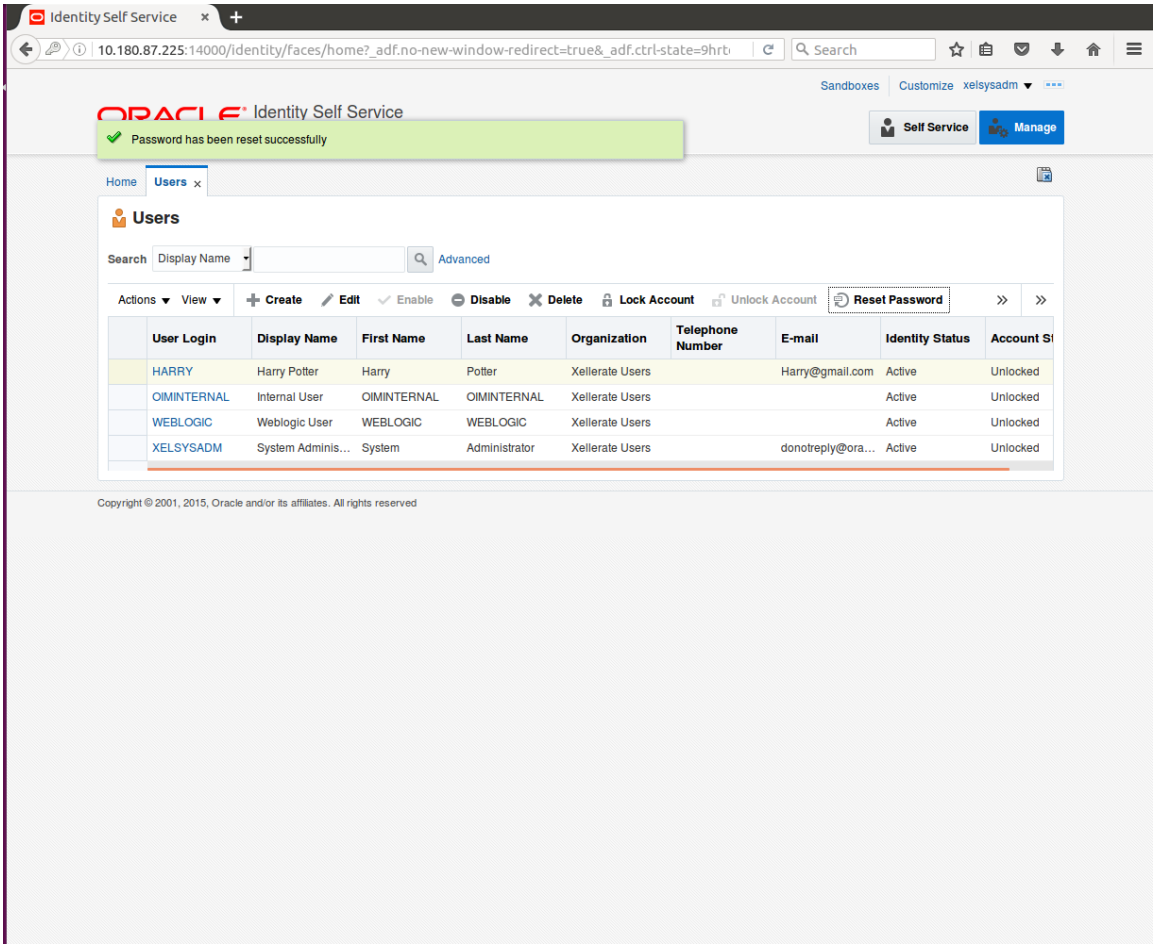
3. If you select the **Manually change the Password** option, enter the new password in the **New Password** and the **Confirm New Password** fields.

Figure 1–19 Resetting User Password in OIM - New Password



The user password is reset successfully.

Figure 1–20 Password Reset Successfully



1.7 User Management Using the Admin Application

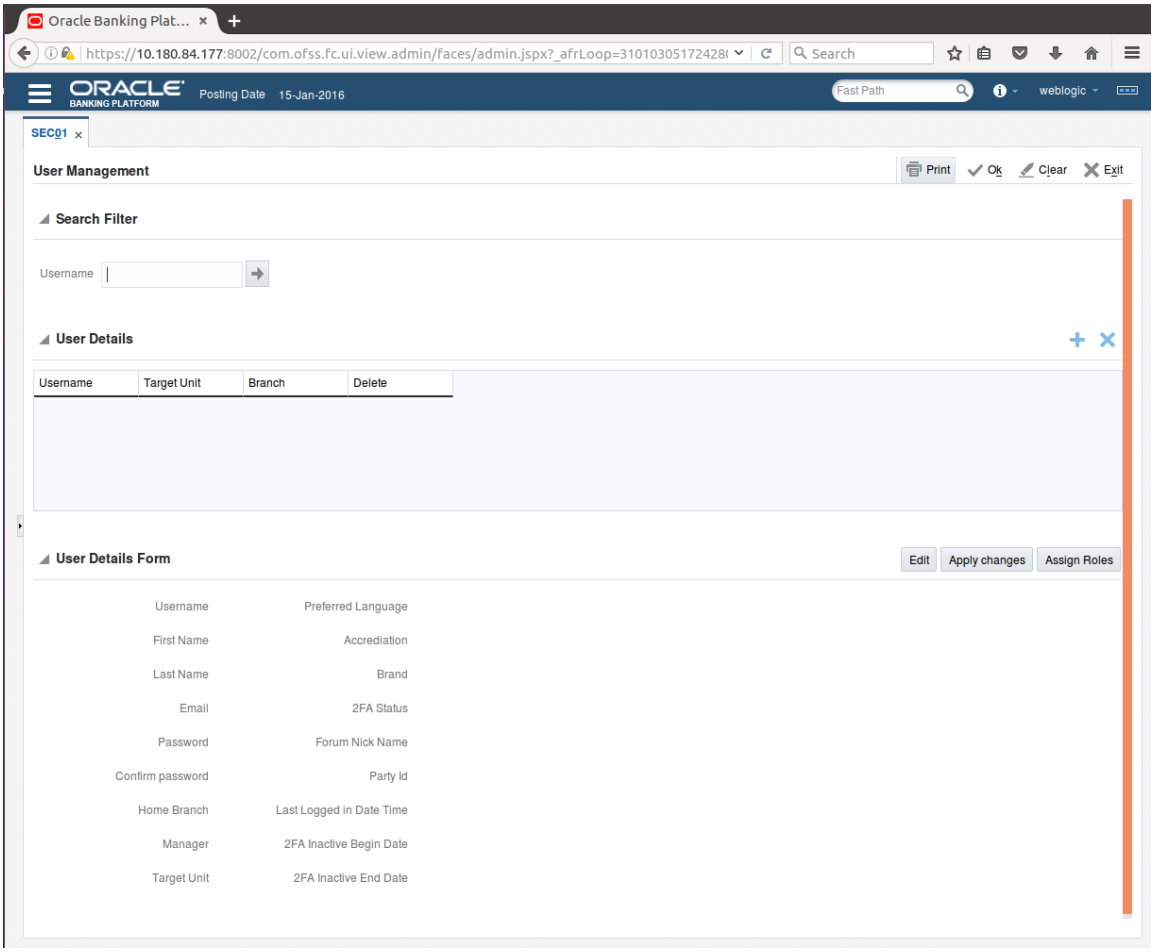
The User Management screen is a quick start UI, provided to create initial users and verify the OBPM installation.

https://<ui-server-name>:<ui-server-port>/com.ofss.fc.ui.view.admin/faces/admin.jspx

To create initial users and verify the installation, perform the below mentioned steps:

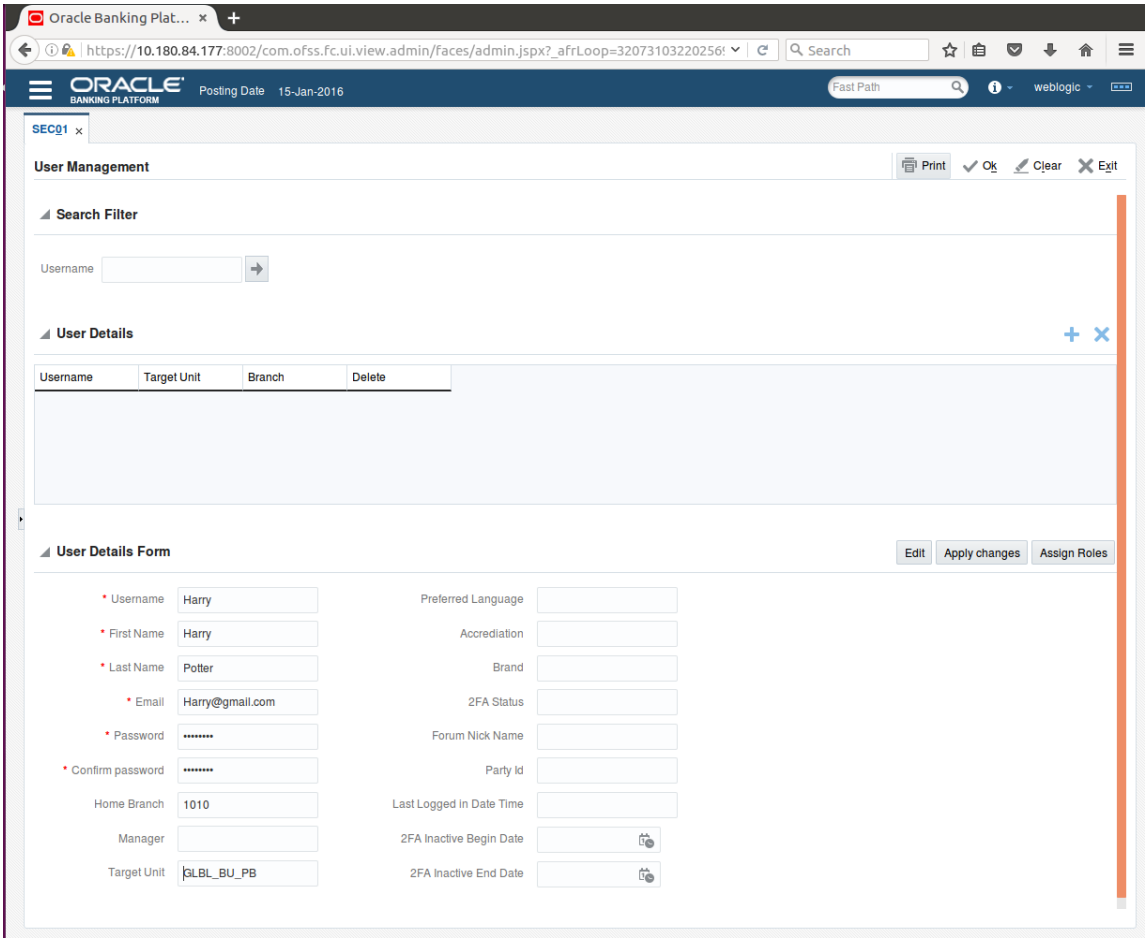
1. Click **Security** tab in **View Admin**.
2. Select **User Management**.
3. Click **+** icon to add a user.

Figure 1–21 Adding a User



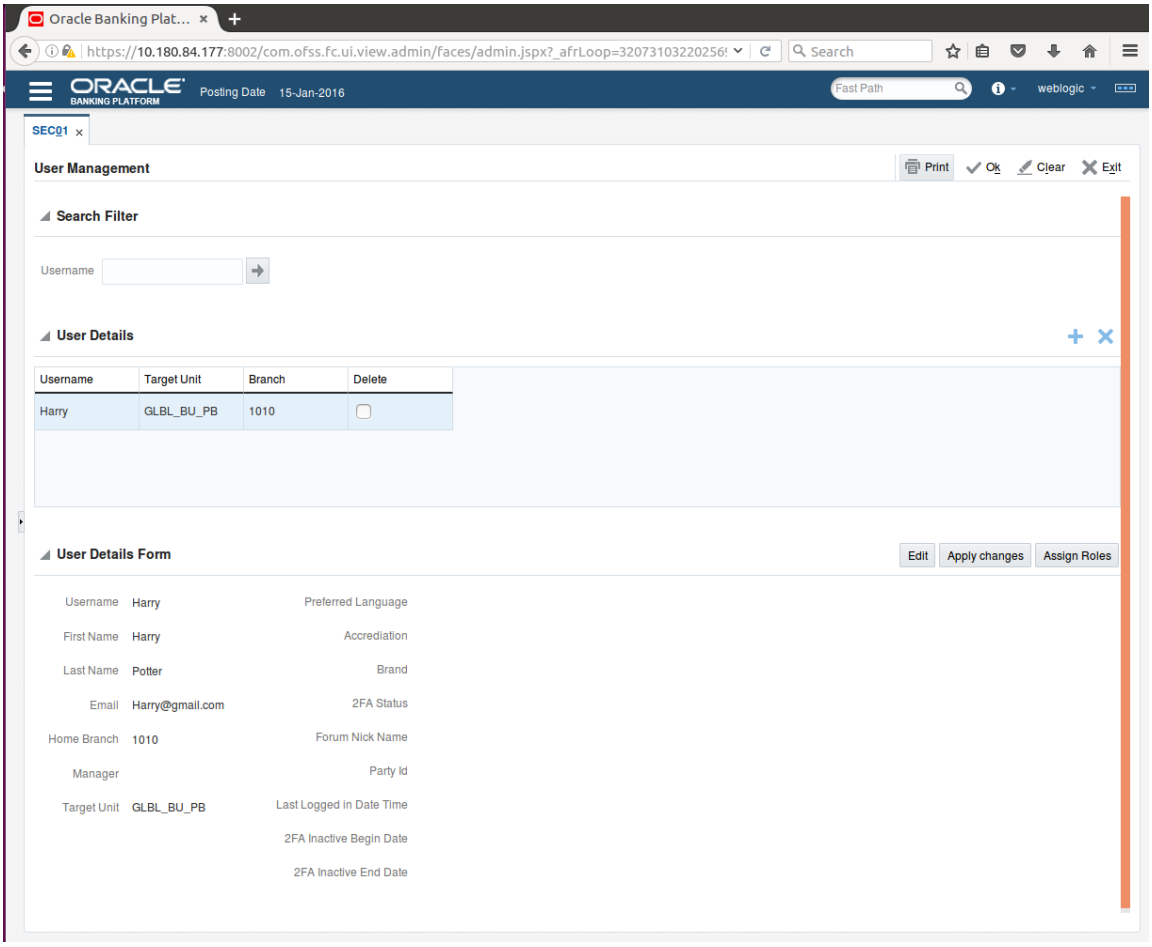
4. Enter the mandatory fields required for creating a user.

Figure 1–22 Enter Mandatory Details



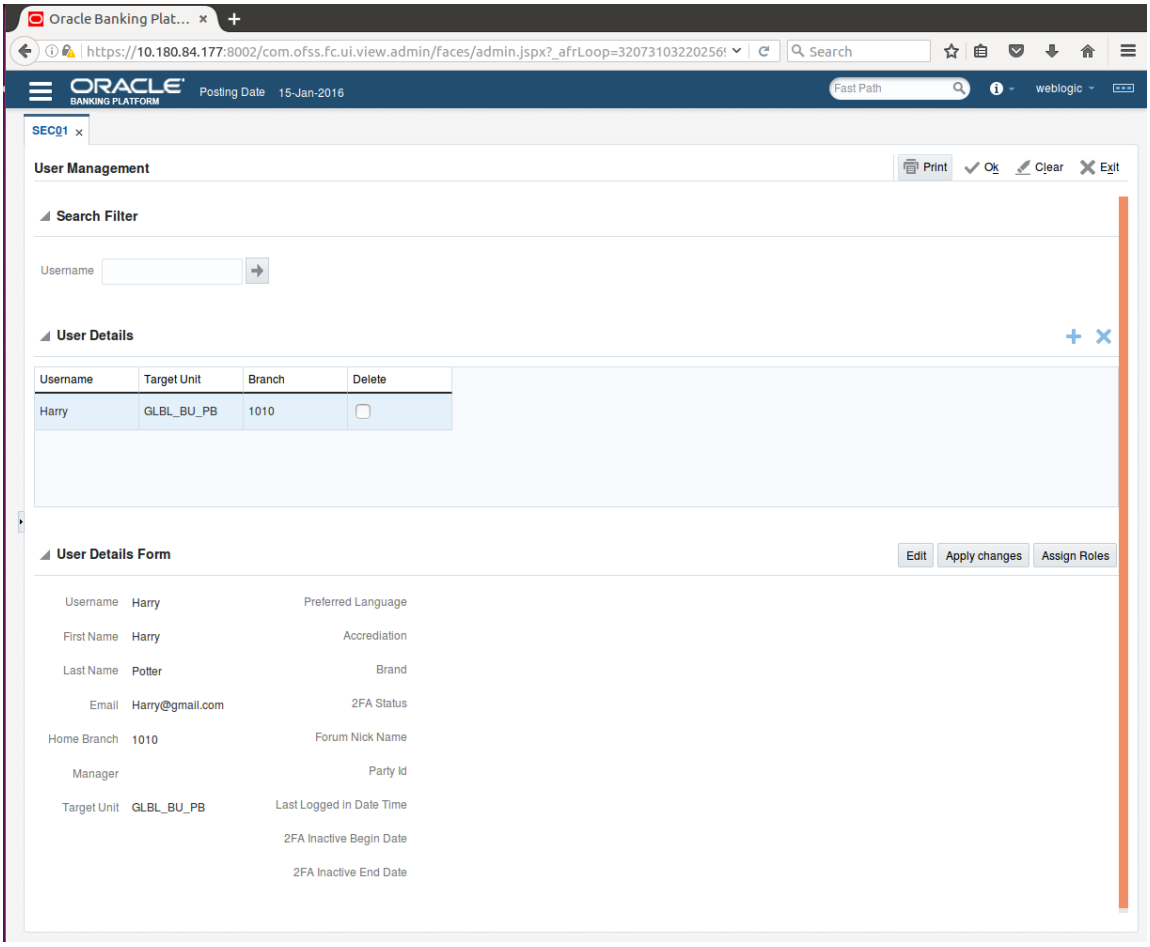
5. Click **Apply Changes** to save the user details locally.

Figure 1–23 Applying Changes



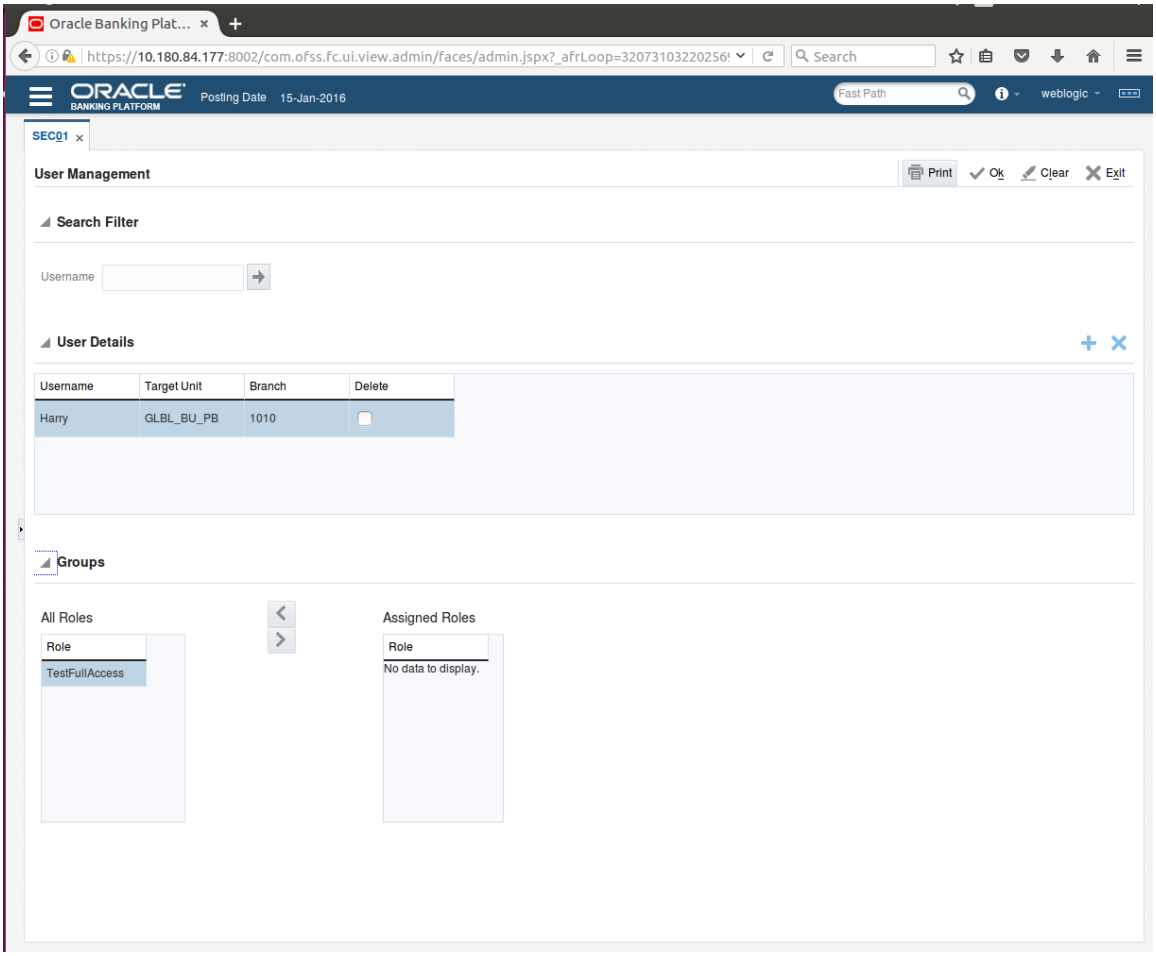
6. To add a user to a group, select the row containing the user and click **Assign Roles**.

Figure 1–24 Adding User to a Group



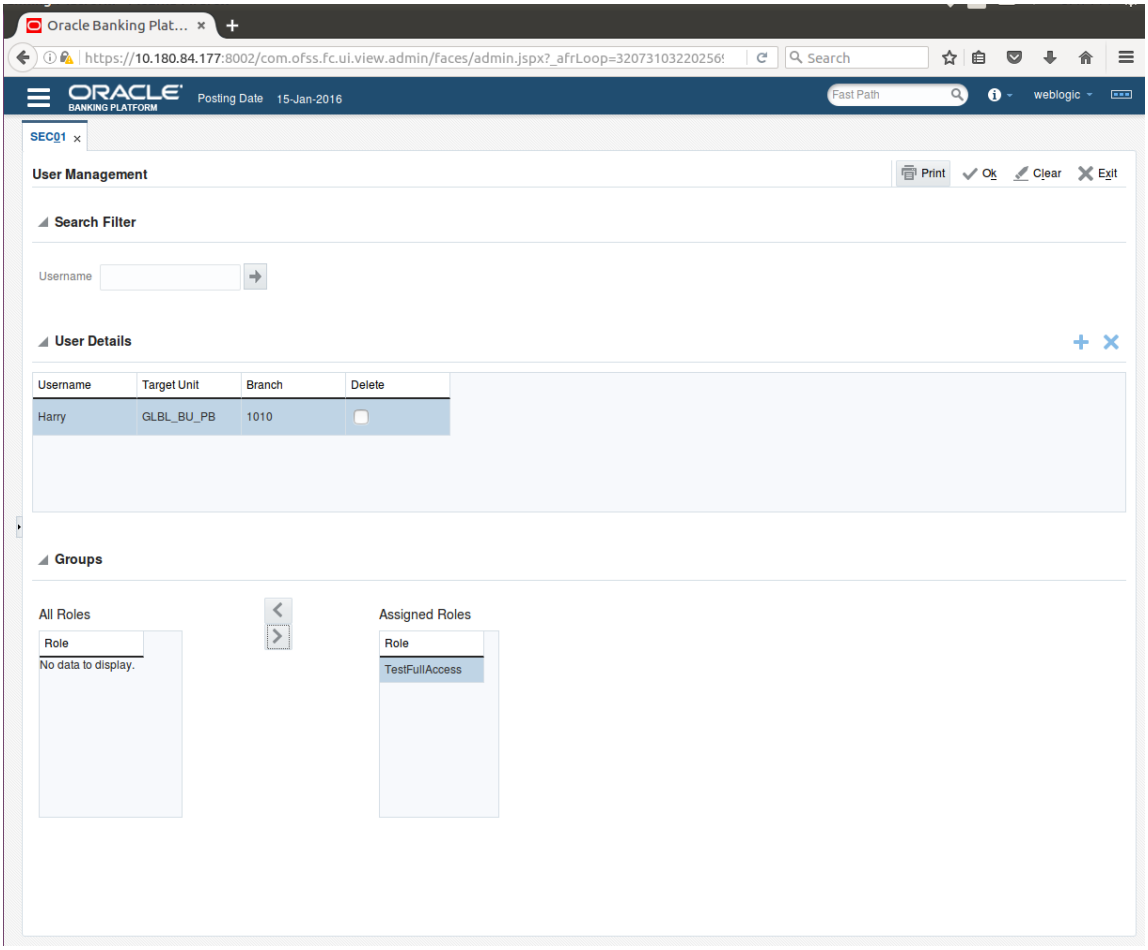
The available and assigned roles appear.

Figure 1–25 Available and Assigned Roles



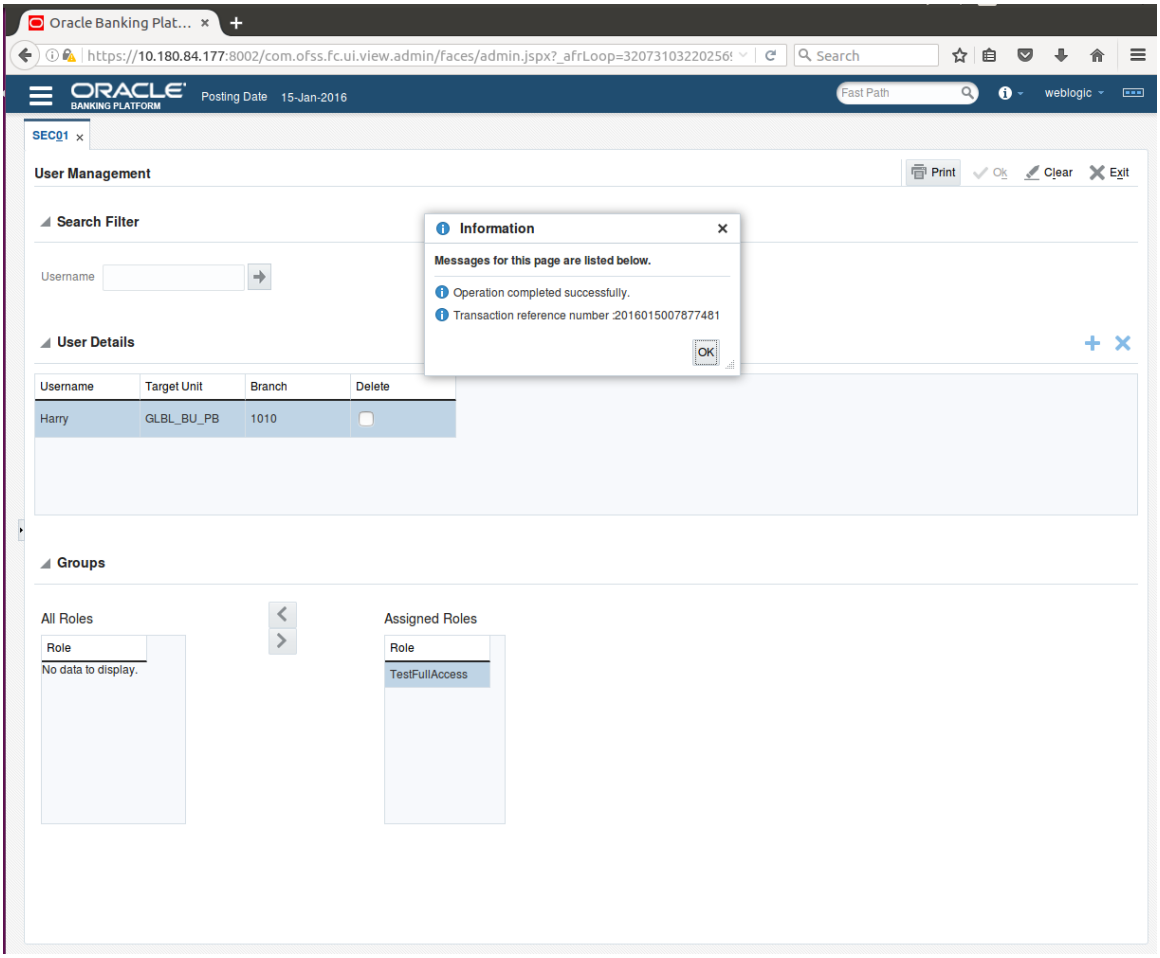
7. Select the group to add user and move it to the **Assigned Roles** table.

Figure 1–26 Adding User to Assigned Roles Table



8. Click **Ok** to save the changes.

Figure 1–27 Save Changes



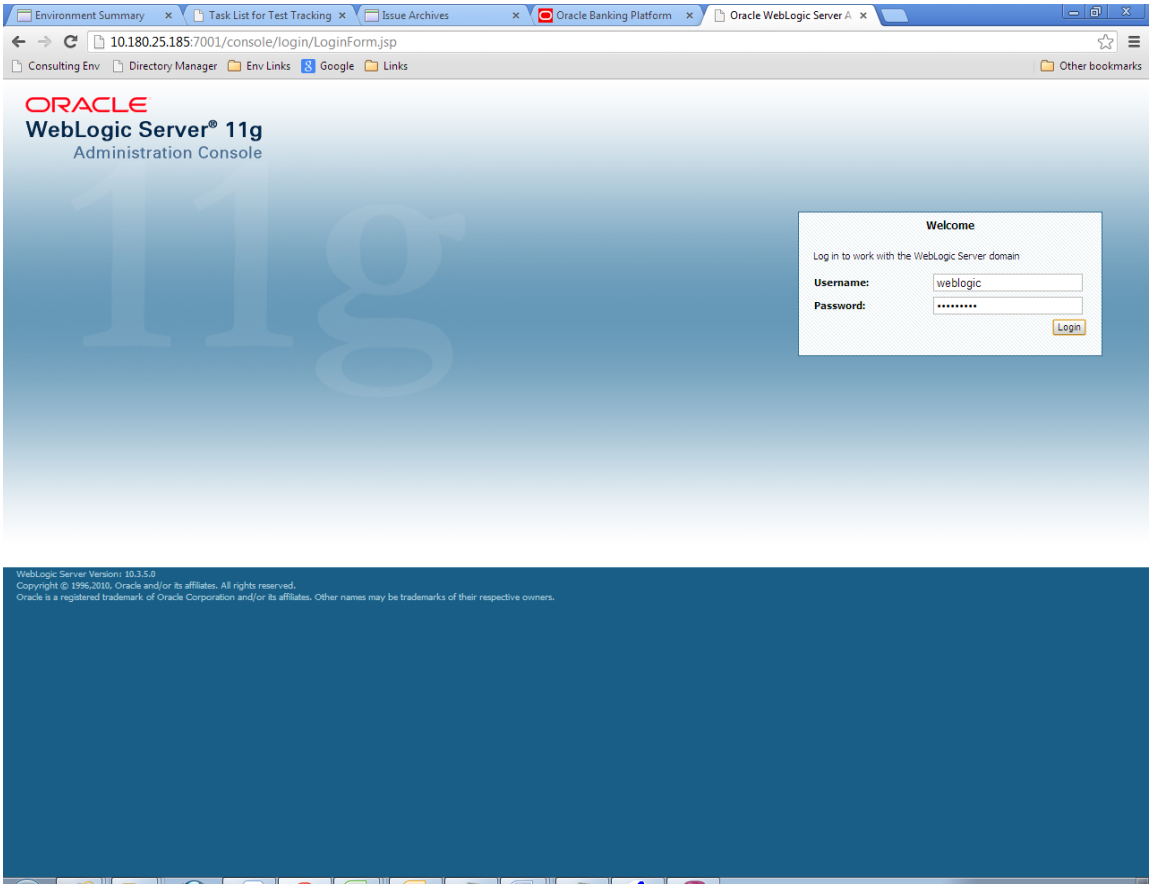
1.8 Unlocking Users in Oracle WebLogic Server (OWS) Administration Console

This section explains the procedure to unlock users in Oracle WebLogic Server (OWS) using Administration Console. If users unsuccessfully attempt to log in to a WebLogic Server instance for more than the configured number of retry attempts, they are locked out of further access. This procedure allows you to unlock locked users so that they can log in again.

To unlock a user in OWS:

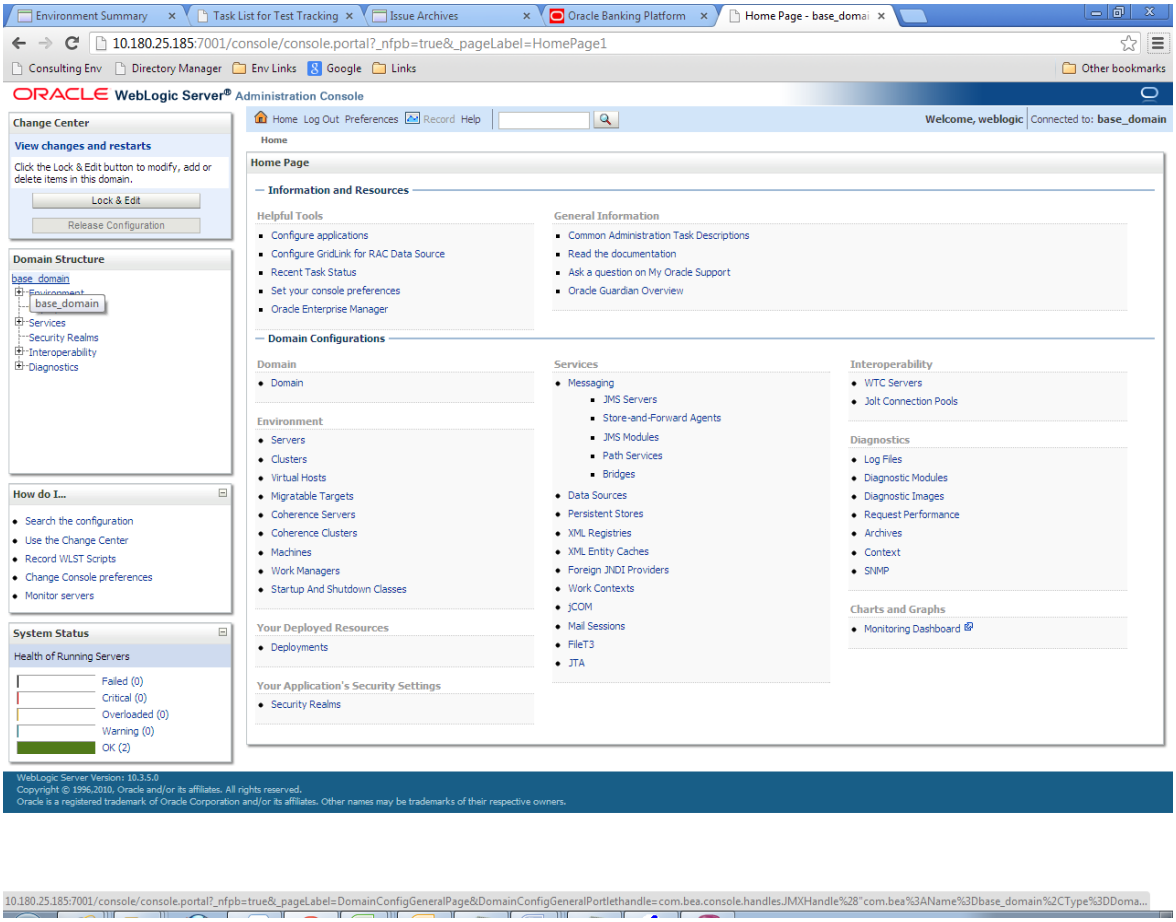
1. Log in to OWS. The **Home Page** of OWS Administration Console appears.

Figure 1–28 OWS Log in



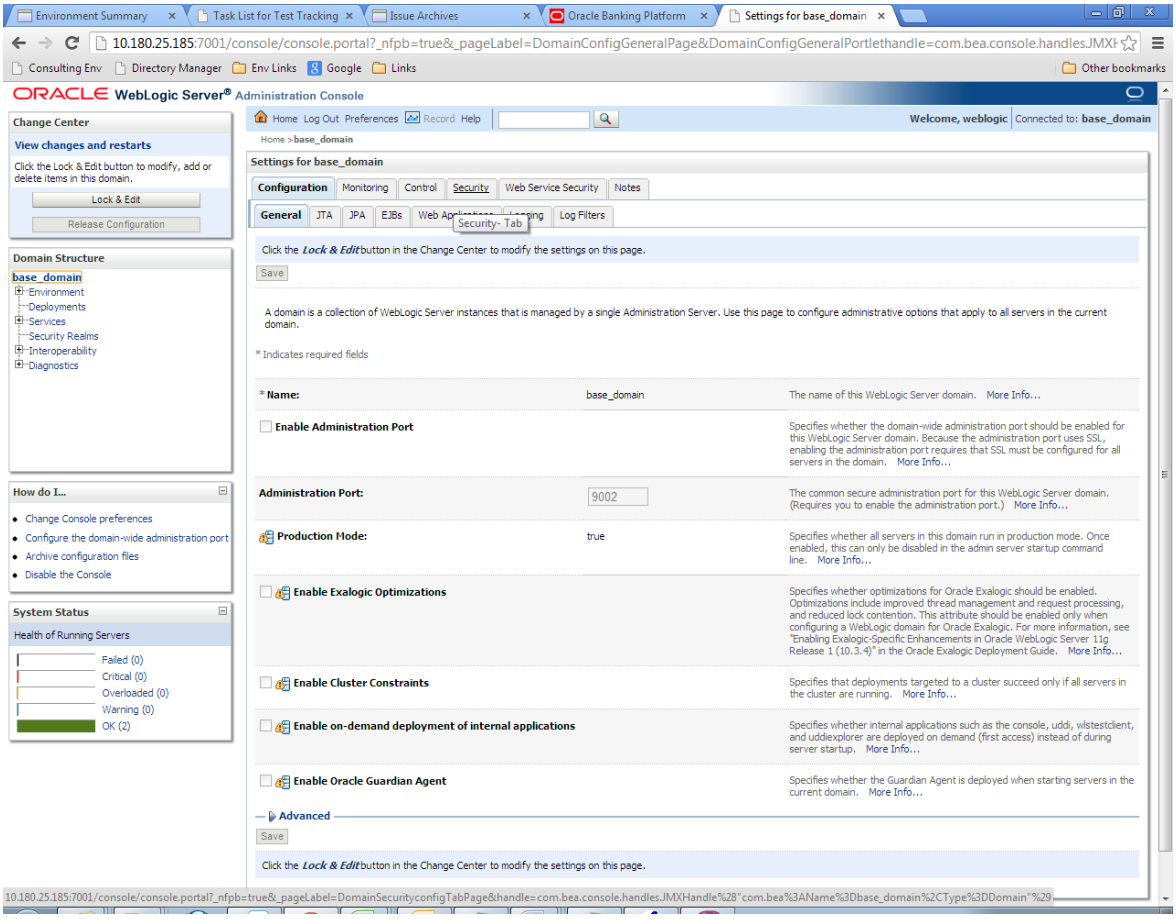
2. In the **Domain Structure** section, click the **base_domain** link.

Figure 1–29 base_domain



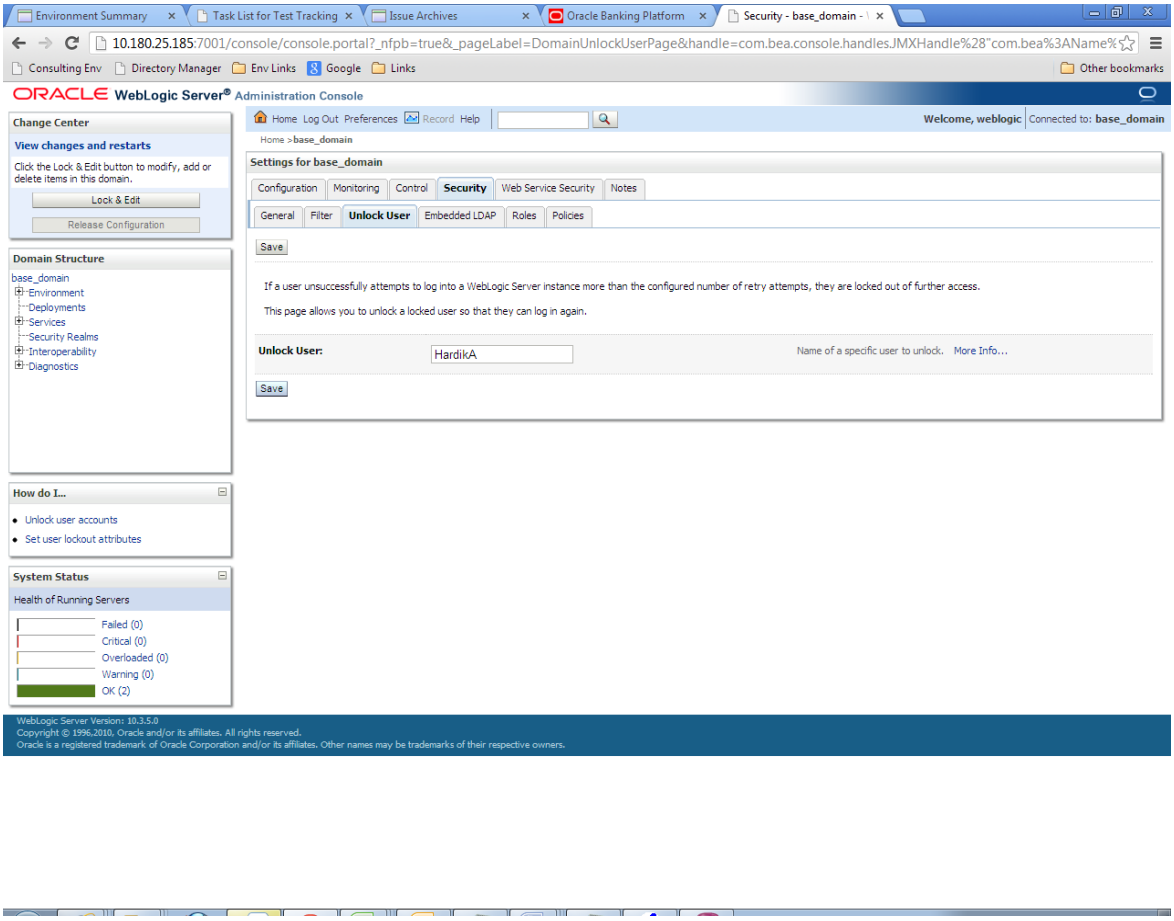
3. In the **Settings for base_domain** page that appears, click the **Security** tab.

Figure 1–30 Security tab



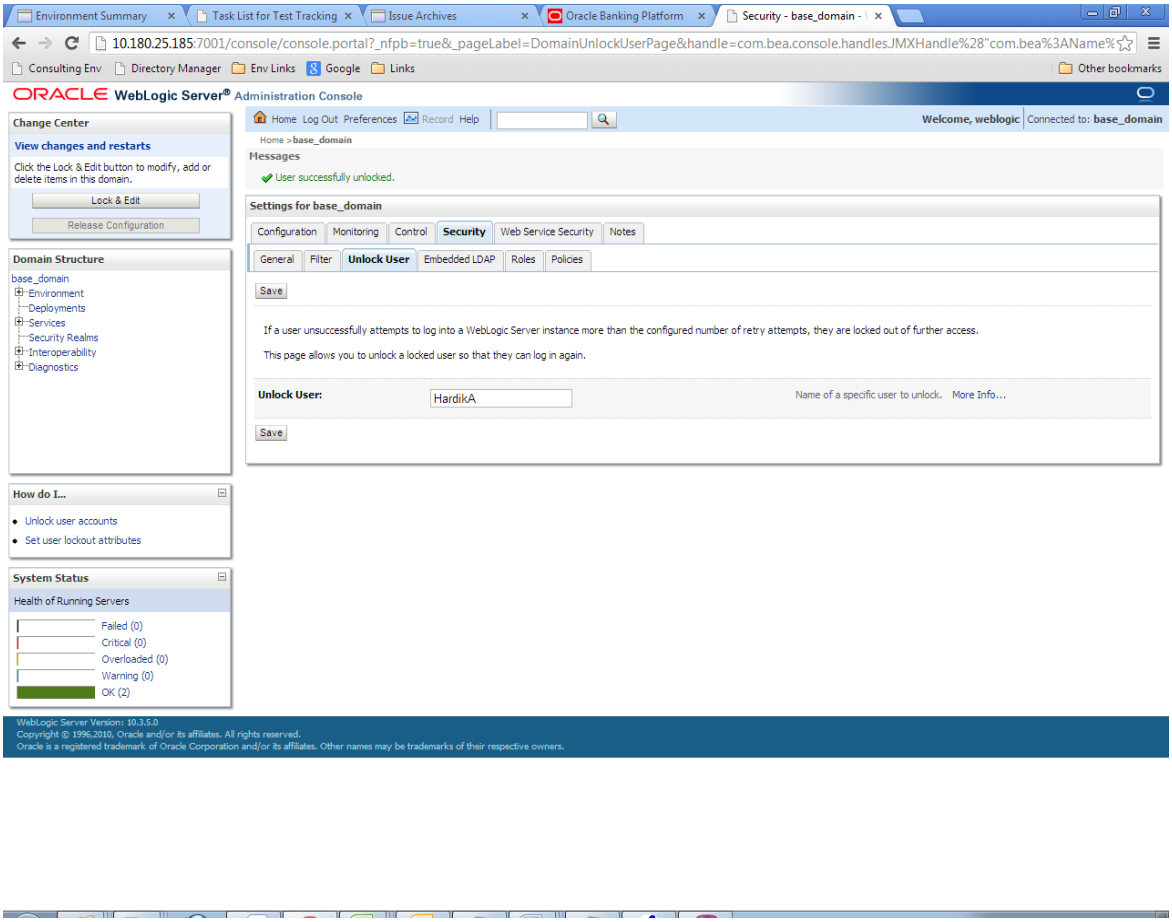
4. Click the **Unlock User** tab.
5. In the **Unlock User** field, enter the User ID to unlock the user.

Figure 1–31 Unlock User



6. Click **Save.** The message *User successfully unlocked* appears.

Figure 1–32 User Successfully Unlocked



On completion of this procedure the user gets unlocked in OWS.

1.9 Creation of first time user to access OBPM

This section explains the procedure to create the first bank user having access to the application.

> **Note**
>
> Make the default authenticator as sufficient in host console and reorder it below OID Authenticator. Also change 'cn' attribute to 'uid' in the All Users Filter and User From Name Filter in OID Authenticator provider specific properties.

1. Log in to OIM using the admin user *xelsysadm*. Create a new role in OIM as described in Section 1.2 Creating Roles in Oracle Identity Manager (OIM). For example, Developer. This creates a group in OID (Developer).
2. Log in to admin application using the weblogic user. Create a user as described in Section 1.7 User Management Using the Admin Application. For example, john.doe.

3. Add the user (john.doe) to the Developer.
4. Map the application role Administrators to the Enterprise Group Developer in EM (refer screenshots below). After doing this, the user should have access to all artifacts assigned to the 'Administrators' role. These access rights can be viewed in OES.

Figure 1–33 Log in Oracle Fusion Middleware Control

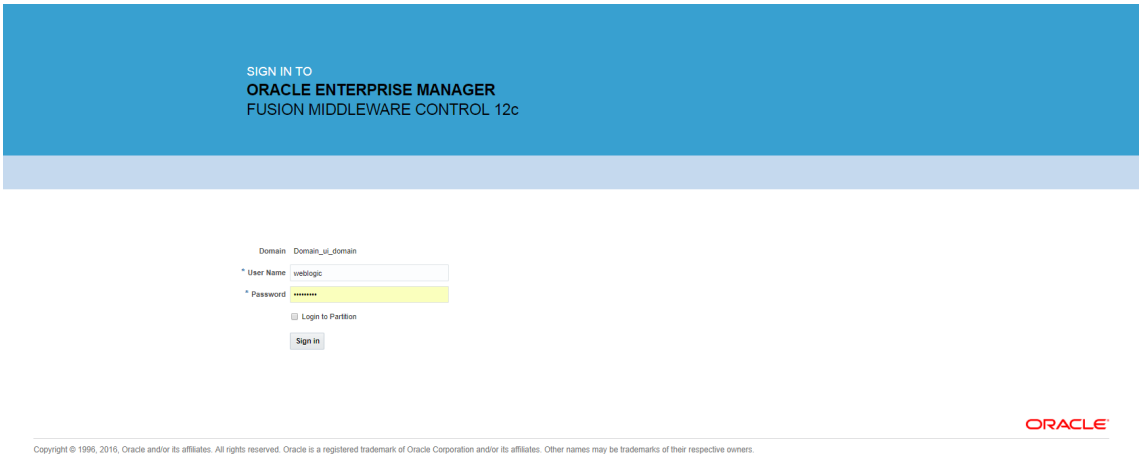


Figure 1–34 Click Application Roles

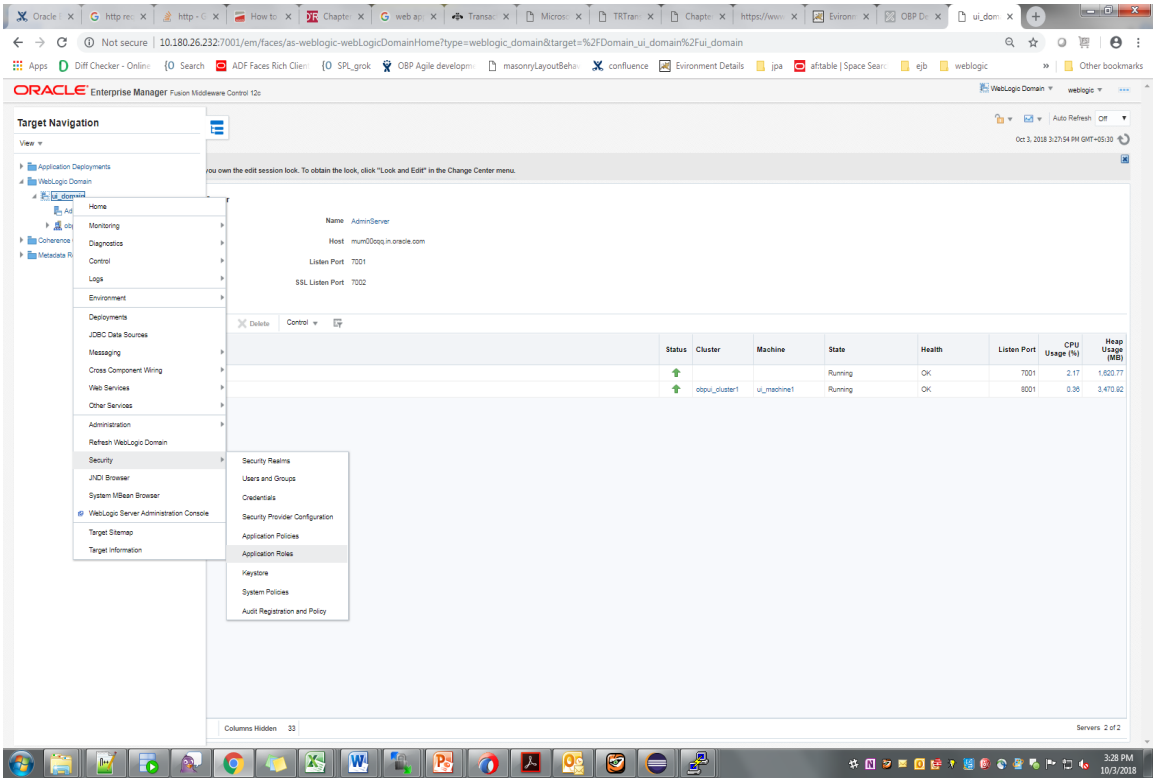


Figure 1–35 Select Administrators Role

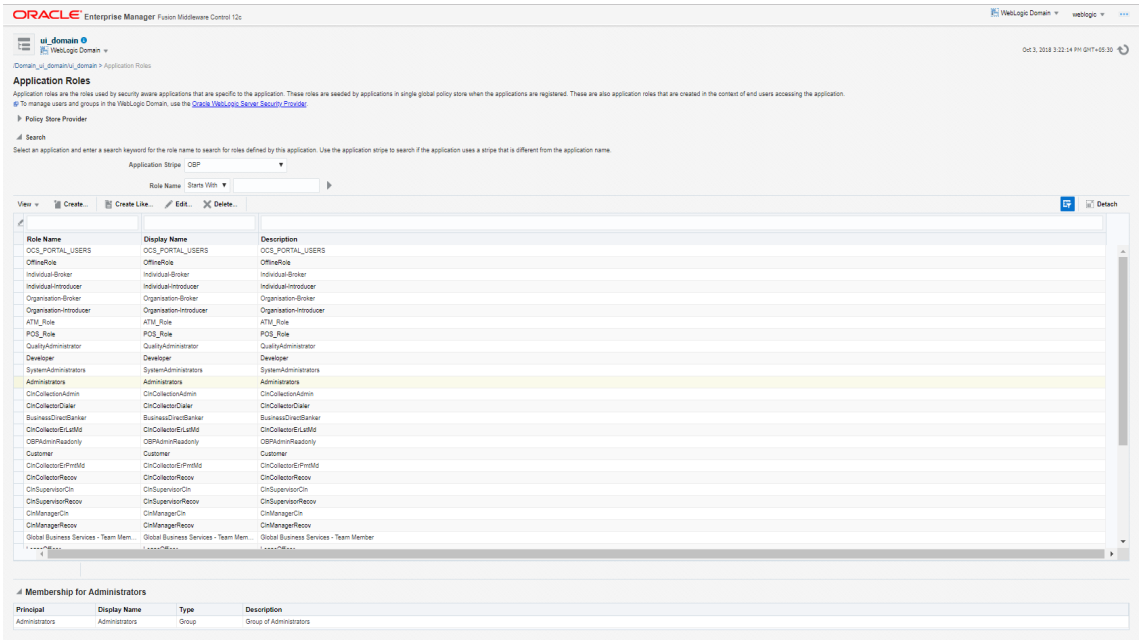
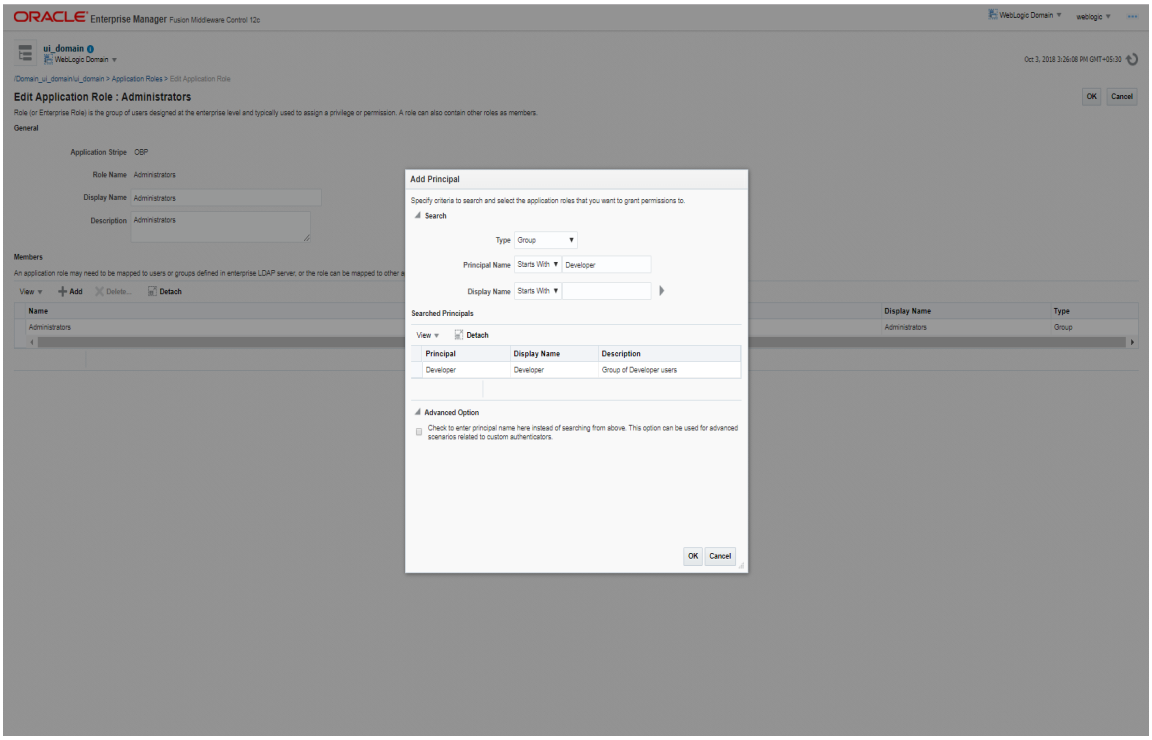


Figure 1–36 Add Principal



2 Approvals Management

This chapter describes worklist authorization related activities and SOA composer rules setup to be performed by an administrator.

2.1 Enabling Worklist Authorization

This section explains the steps in enabling Worklist authorization. Following are the steps:

Step 1 Identify the Service Name

For example,
com.ofss.fc.appx.party.service.contact.ContactPointApplicationServiceSpi.updateAllContactPoints.

Step 2 Enable Dual Authorization or/and adding other severity

Once the services are identified, follow the below steps to enable Dual Authorization and adding other severity.

1. Log in to the application.
2. Navigate to **Artifact Dependency Map (Fast path: SM500)** page.
3. In the **Search Text** field, enter the service-name.
   com.ofss.fc.appx.party.service.contact.ContactPointApplicationServiceSpi.updateAllContactPoints

Figure 2–1 Select the Service to be Configured

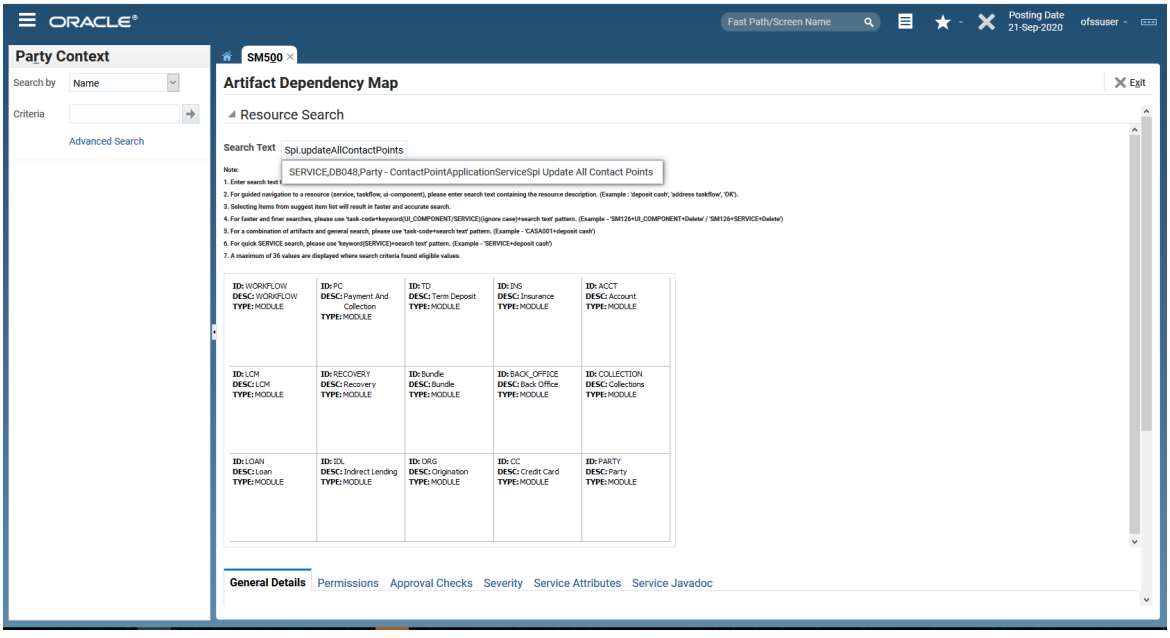
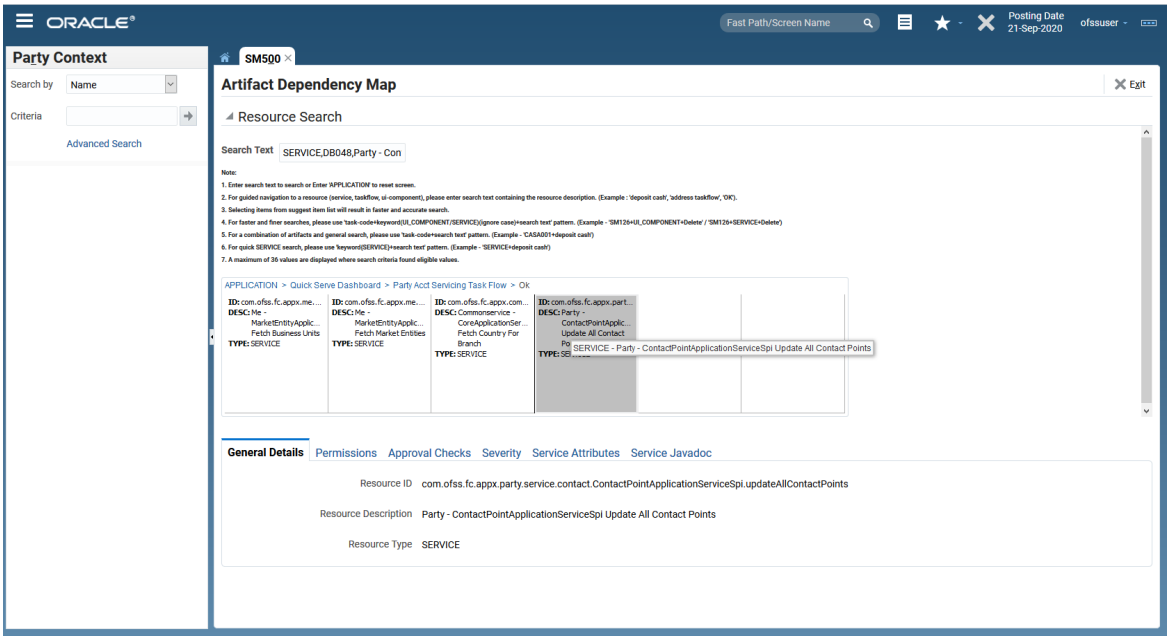
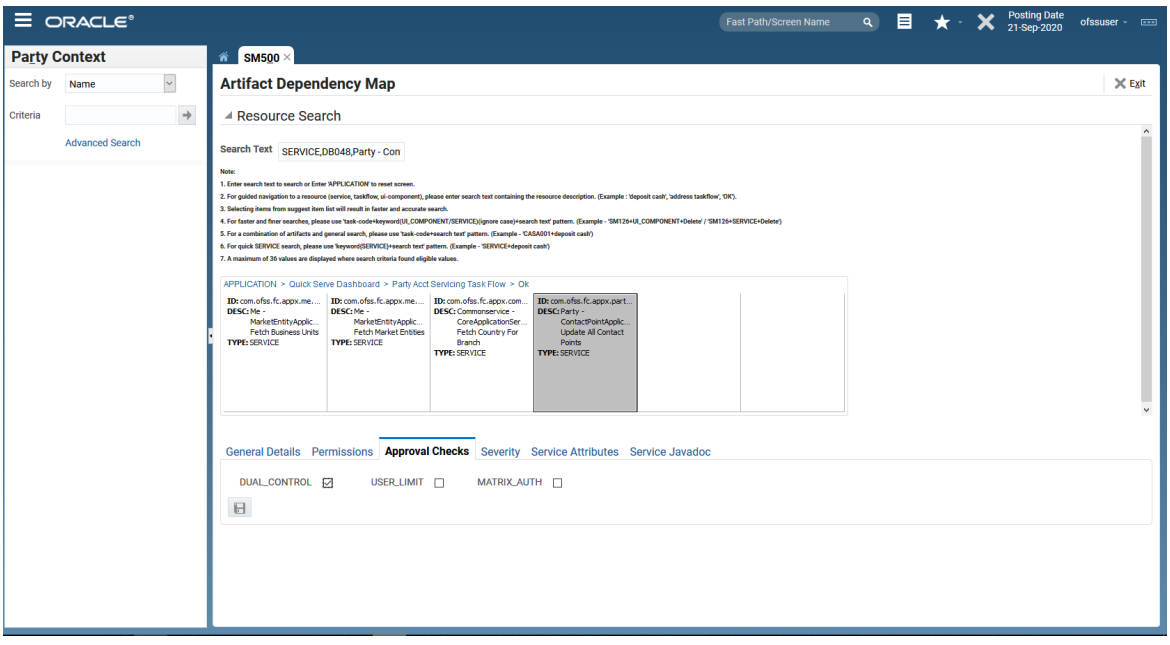


Figure 2–2 Search for Service using TASK CODE + Search text (in case of non Origination)



4. Navigate to service node by following highlighted path (in grey color) and select the service node.
5. Click the **Approval Checks** tab and add approval checks.

Figure 2–3 Approval Checks tab - Add Approval Checks



Step 3 Configure Severity

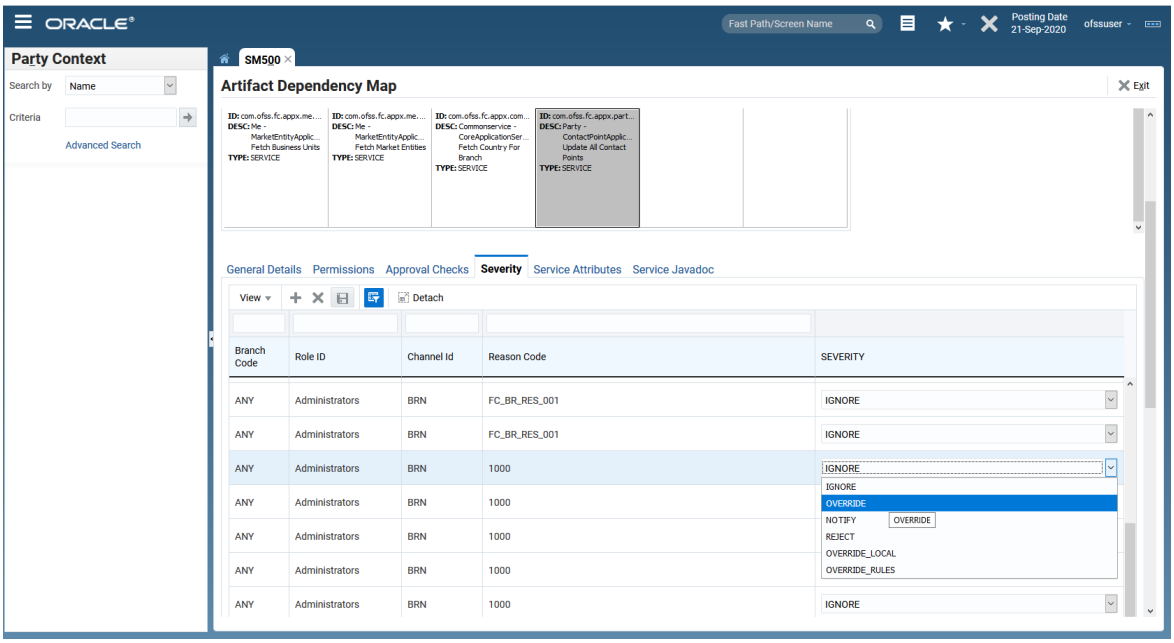
Enabling of dual authorization alone will not send the transaction for approval. Hence, we need to configure Severity for the identified service, to enable the call for approval workflow.

This can be configured from the **Artifact Dependency Map (Fast path: SM500)** page.

Follow the below steps to configure severity:

1. Log in to the application.
2. Navigate to the **Artifact Dependency Map (Fast path: SM500)** page.
3. In the **Search Text** field, enter the service com.ofss.fc.appx.party.service.contact.ContactPointApplicationServiceSpi.updateAllContactPoints.
4. Ensure approval checks are added. If not, then configure the approval checks.
5. Click the **Severity** tab and update the severity, if already maintained. Else, click **Add** button in the toolbar to add new row in the table.

Figure 2–4 Add New Severity



6. Enter the following details in the **Severity** tab.

Branch Code	Branch code from which the transaction is to be performed. Specify ANY to configure for all branches. Example: 082991
Role ID	Security Role to which the user belongs and initiates the transaction.
Channel ID	Channels such as BRN, ATM, IB and so on, through which the transaction is performed.
Reason Code	Select Reason Code 1000 normal approval flow.
Severity	This field contains four values as detailed below: ■ **Ignore:** Allows transaction to complete without any

	authorization, that is Auto Authorization. ■ **Override:** Transaction will be sent for Authorization. ■ **Notify:** In this case, the task is not sent for authorization, but the user is expected to confirm the transaction for proceeding ahead. This option is not applicable in case of Dual Authorization. ■ **Reject:** System does not allow to proceed with transaction.

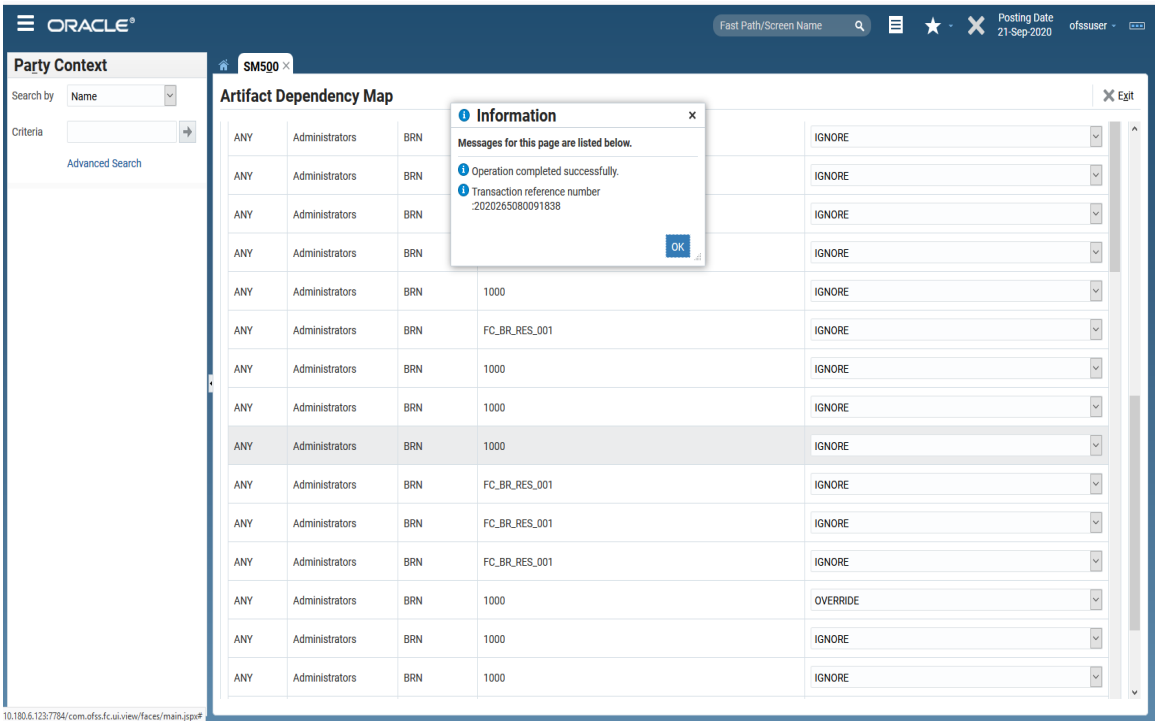
To enable Dual Authorization, select **Override** option.

Note

If the Severity Configuration is already set up do not change it.

7. Save the newly added severity using the **Save.**

Figure 2–5 Save Severity Details

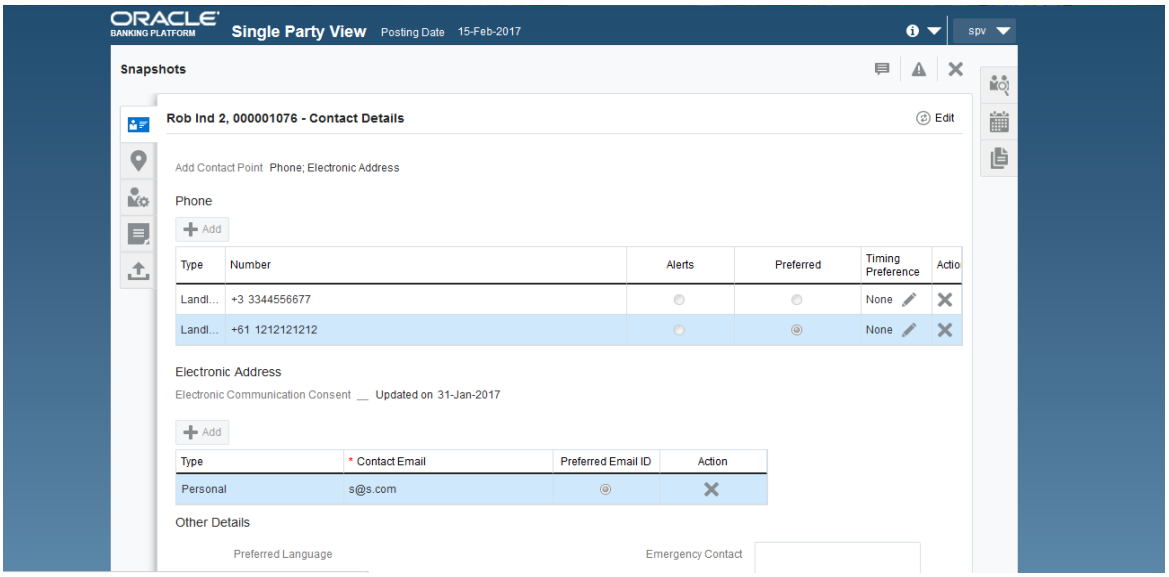


Step 4 Testing Approvals

Once the configurations are done and a transaction is performed from a screen, it will be sent for approval. The process is explained with the following example.

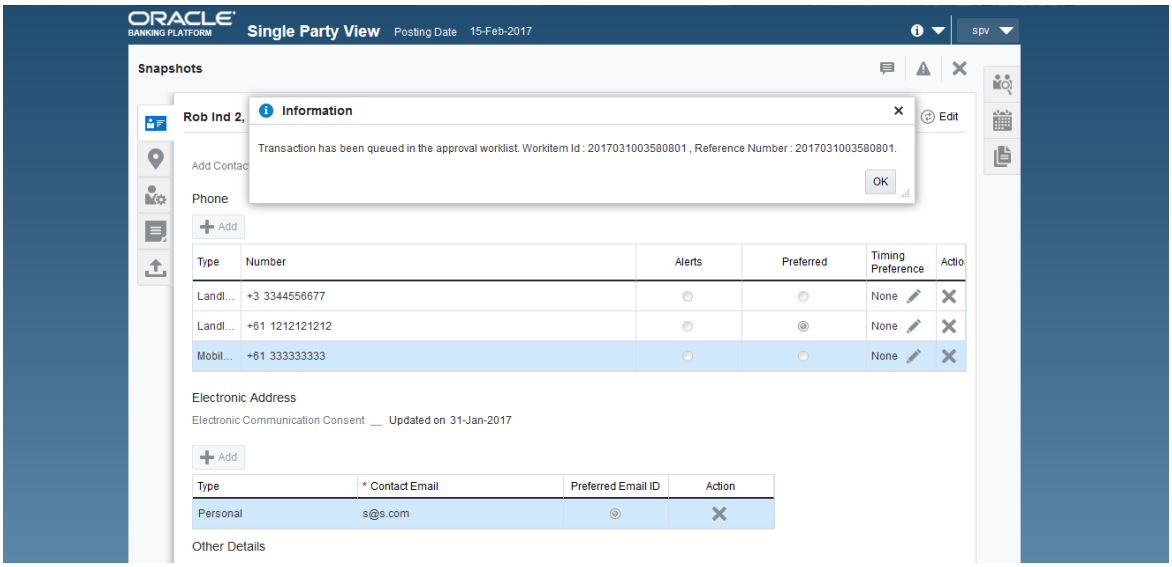
1. Log in to the application.
2. Navigate to the Contact Point page in Single Party View.

Figure 2–6 Contact page of Single Party View



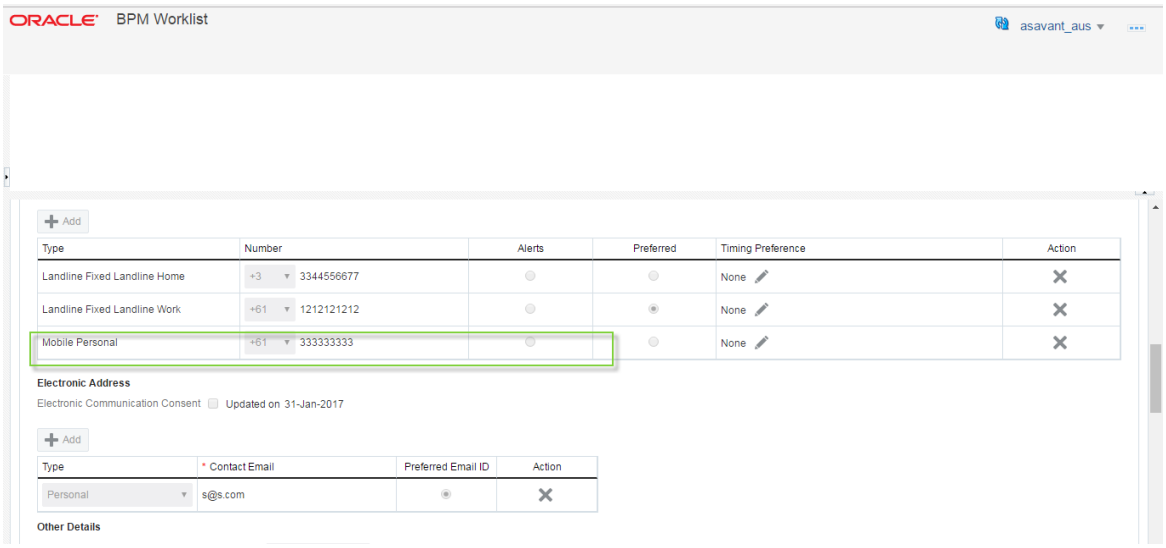
3. Update the details and submit the transaction.

Figure 2–7 Update details and submit



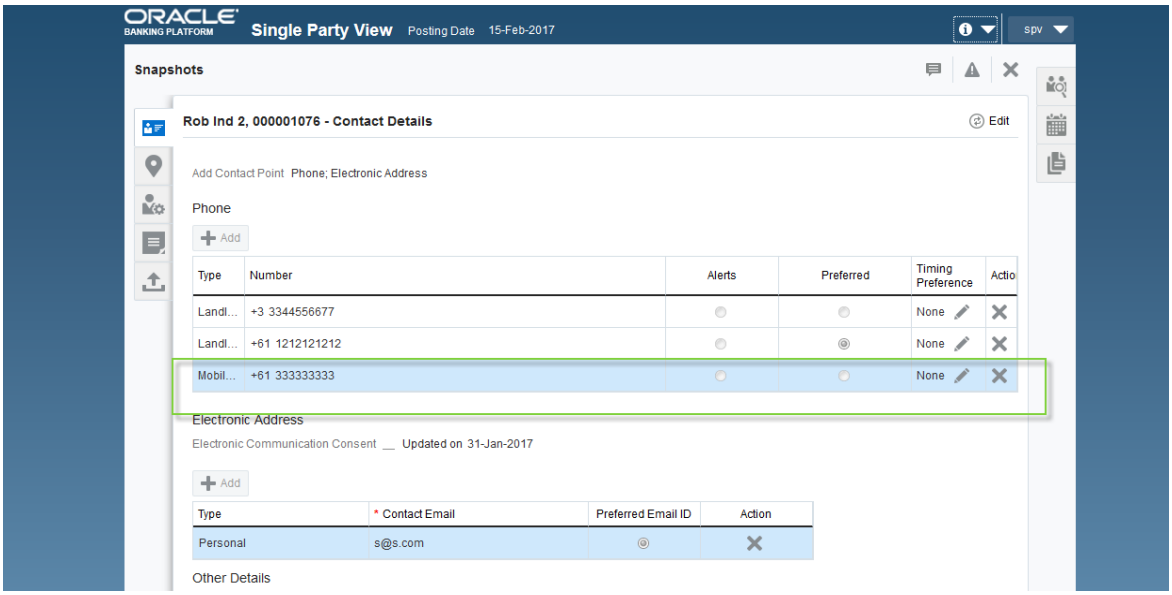
4. Log in to the BMP Worklist application and approve the transaction.

Figure 2–8 Approve transaction in BPM Worklist



5. Verify the details in the Contacts page of Single Party View.

Figure 2–9 Updated Details Approved



3 Defining Task Configuration Rules

Each human task in Origination business process has business rule associated with it. This business rule can be used to set various parameters for the task like SLA period, Assignees, task priority, task owner, STP configuration, and so on. Below sections illustrate the steps to configure such business rules.

3.1 Important Rule Artifacts

This section provides information about important rule artifacts.

3.1.1 Rules Dictionary

For each human task, a *<name>TaskConfigRules.rules* file is provided. For example, *VerifyApplicationTaskConfigRules.rules.* These rules dictionary files have to be used to configure attributes of the respective human task.

Rules dictionaries can be viewed and edited using Oracle SOA Composer. Oracle SOA composer can be accessed using http://<IP-of-SOA-server>:8001/soa/composer.

Figure 3–1 and Figure 3–2 illustrate the procedure of opening the rules dictionary for VerfiyApplicationTask.

Figure 3–1 SOA Composer - Open Rules Dictionary Browser

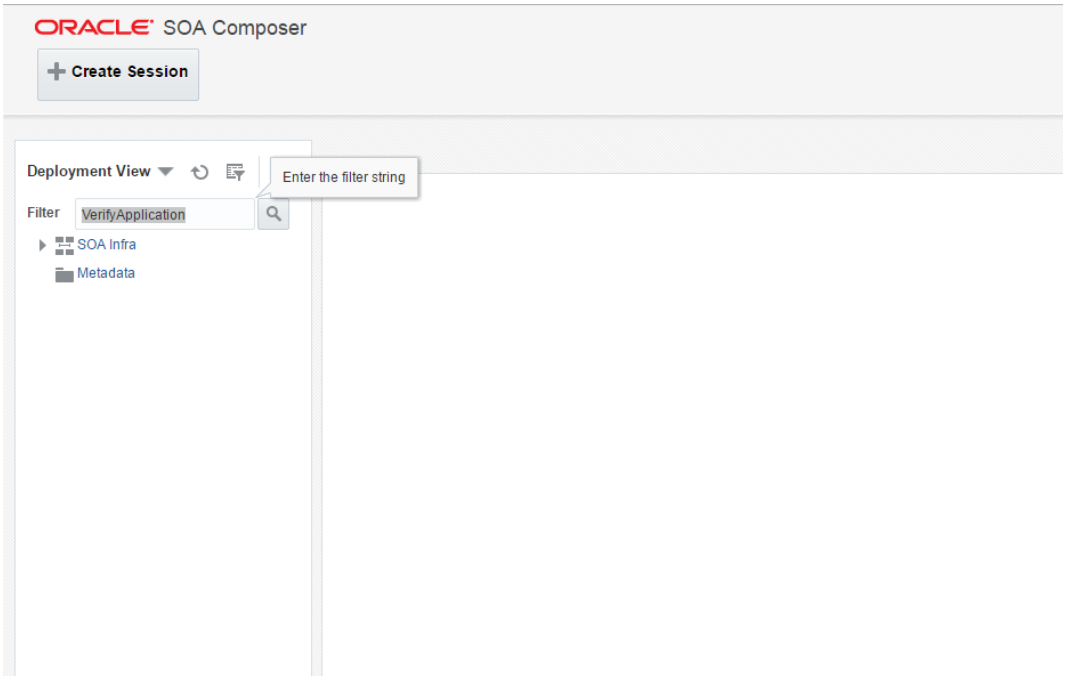
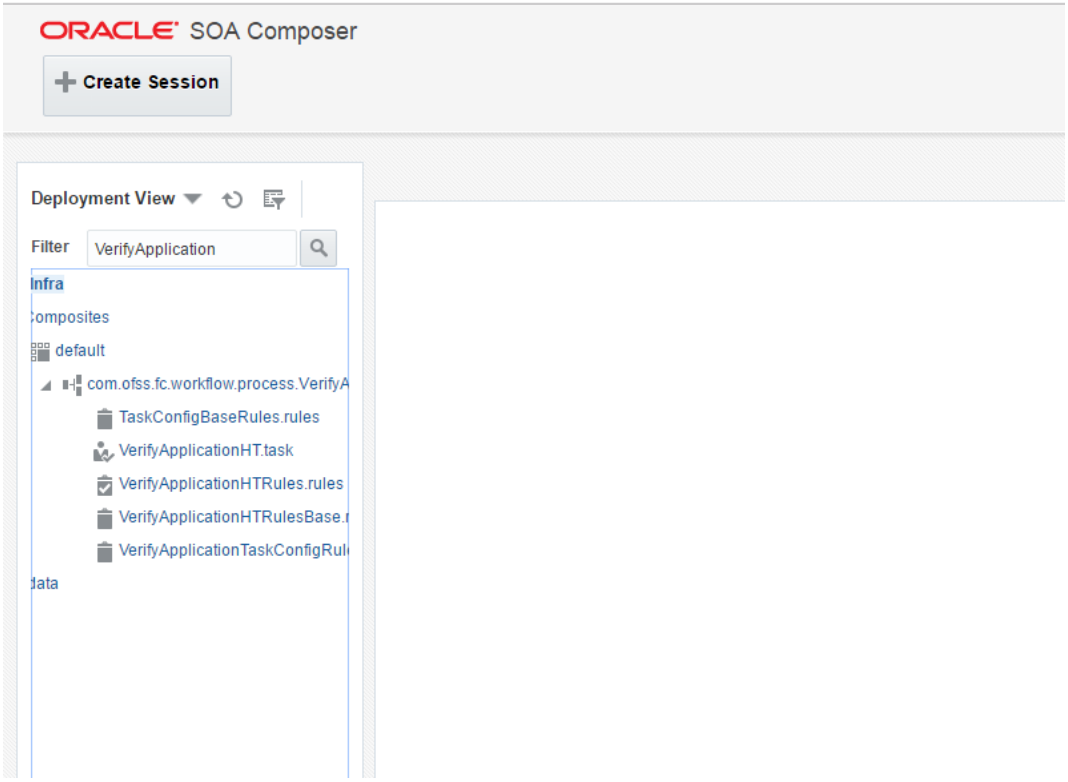


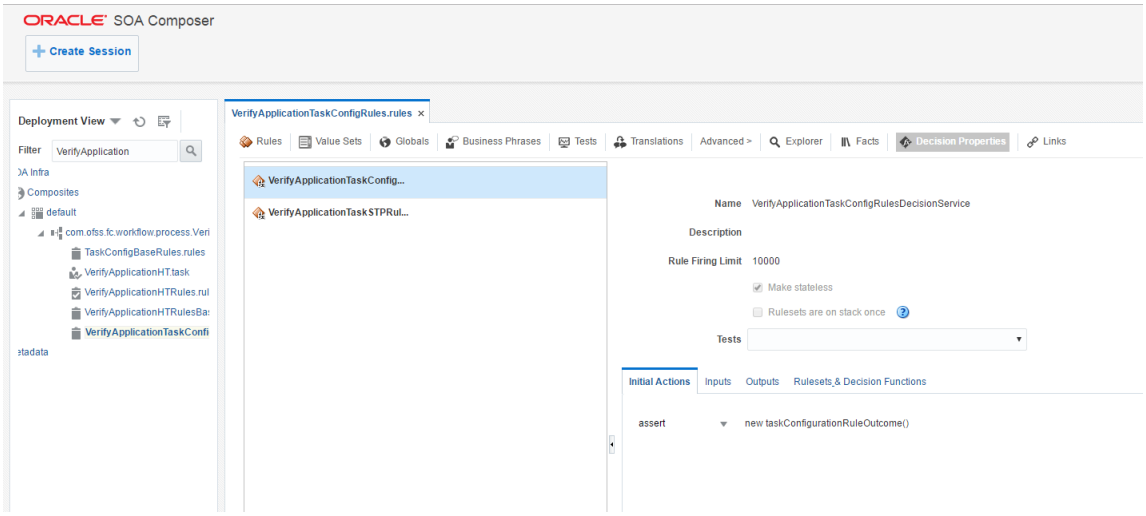
Figure 3–2 SOA Composer – Selecting Rules Dictionary



3.1.2 Decision Function

Configuration rules are written in Rulesets, which are executed through a Decision Function. In each rules dictionary a decision function is provided by the name of <name>TaskConfigurationRulesDecisionService.

Figure 3–3 SOA Composer – Selecting Decision Function



3.1.3 Rulesets

Each decision function executes one or more rulesets. This is where the rules are written. Any rulesets that are defined in the rules dictionary can be added to be used in a decision function.

Figure 3–4 illustrates addition or removal of rulesets from a decision function.

Figure 3–4 SOA Composer - Adding Rulesets to Decision Function

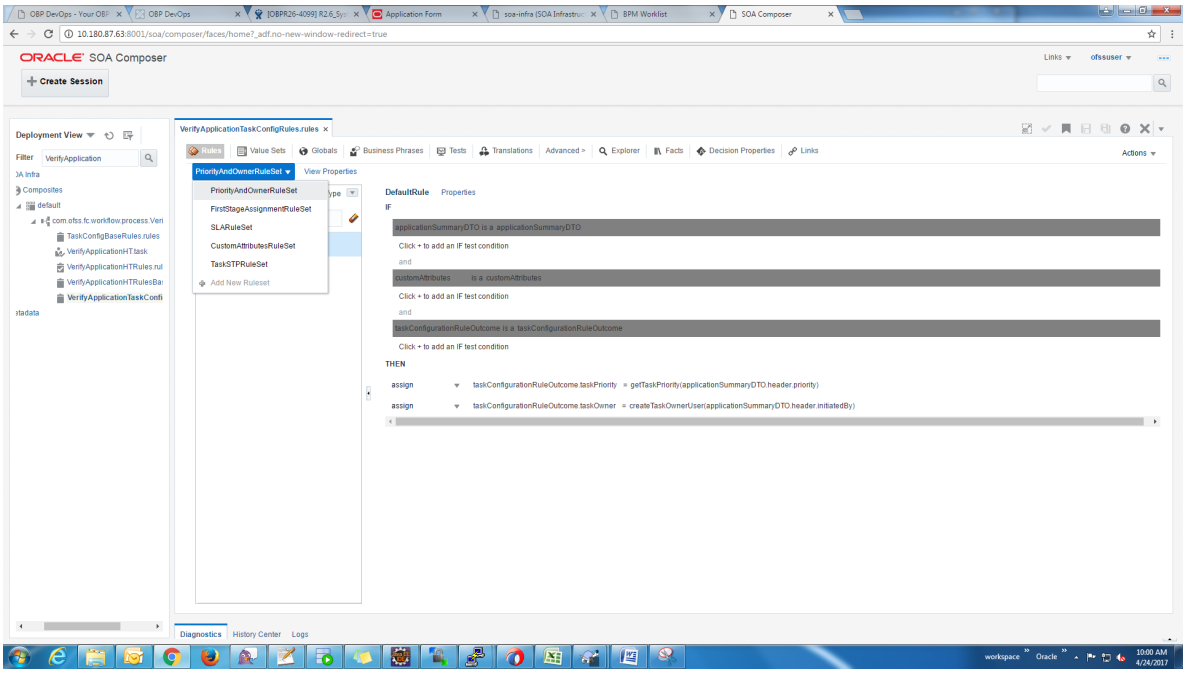
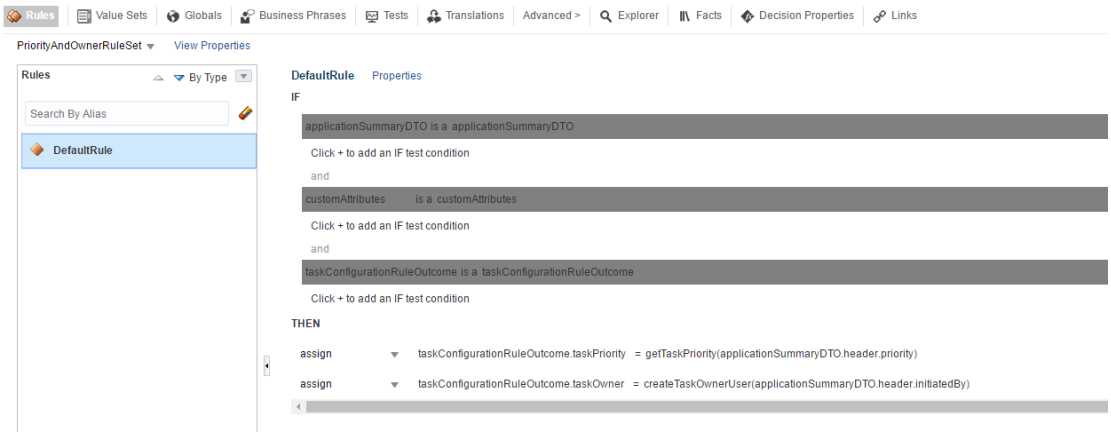


Figure 3–5 shows what a ruleset looks like.

Figure 3–5 SOA Composer - Viewing a ruleset

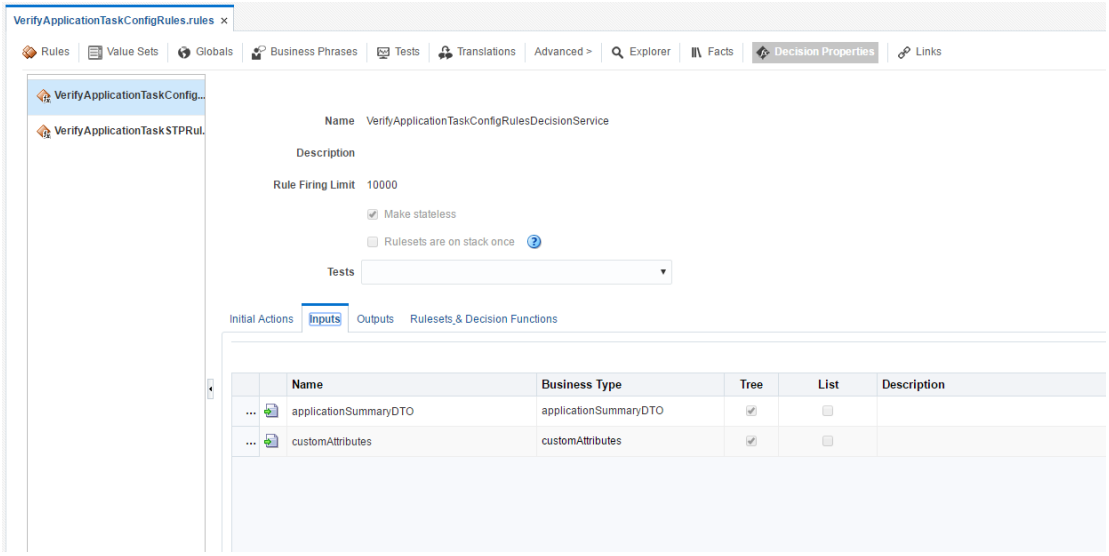


3.2 Inputs to Decision Function

A decision function can take in data objects as input. The rulesets executed by the decision function then work on those inputs to create the output.

Figure 3–6 shows the inputs to a decision function. In this example, there are two input objects - customAttribtues of type CustomAttributes and applicationSummaryDTO of type ApplicationSummaryDTO.

Figure 3–6 SOA Composer - Viewing inputs to a Decision Function



In the example, the applicationSummaryDTO is used in the rules to determine the task priority and the task owner. This is illustrated in Figure 3–5.

3.2.1 Custom Input Attributes

CustomAttributes allow three types of attributes - text, number and date, for which, it has following members, respectively:

- CustomTextAttributeList
- CustomNumberAttributeList
- CustomDateAttributeList

Each of these members has a list of respective types,

- CustomTextAttribute
- CustomNumberAttribute
- CustomDateAttribute

All of these three thee types have a similar steps to configure rules structure and have two members:

- attributeName, of type String
- attributeValue, of type String, int or dateTime, respectively

The UML class diagram of the type CustomAttributes is shown in Figure 2-g. For details on dateTime, please refer http://www.w3.org/TR/xmlschema-2/#dateTime

To access the custom attributes passed as input to the decision service, following three functions are provided:

- getCustomTextAttribute(CustomAttributes customAttributes, String attributeName)
  - Return type - String
- getCustomNumberAttribute(CustomAttributes customAttributes, String attributeName)
  - Return type - int
- getCustomDateAttribute(CustomAttributes customAttributes, String attributeName)
  - Return type - XMLGregorianCalendar

Figure 3–7 shows example usage of custom attributes.

Figure 3–7 SOA Composer – Example Usage of Custom Attributes

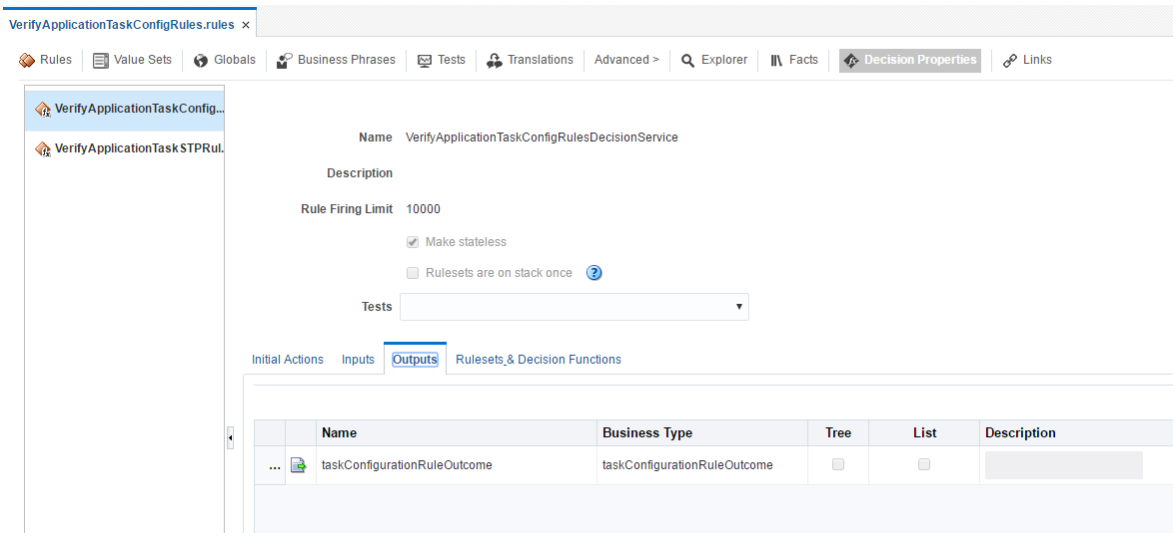
```
assign new ▾  String dummyCustomInputText = getCustomTextAttribute(customAttributes, "dummyCustomInputText")
assign ▾  dummyCustomInputText = dummyCustomInputText.toUpperCase()
addCustomTextAttribute(taskConfigurationRuleOutcome, "dummyCustomOutputText", dummyCustomInputText)
assign new ▾  int dummyCustomInputNumber = getCustomNumberAttribute(customAttributes, "dummyCustomInputNumber")
assign ▾  dummyCustomInputNumber = dummyCustomInputNumber + 7
addCustomNumberAttribute(taskConfigurationRuleOutcome, "dummyCustomOutputNumber" , dummyCustomInputNumber)
assign new ▾  XMLGregorianCalendar dummyCustomInputDate = getCustomDateAttribute(customAttributes, "dummyCustomInputDate")
assign ▾  dummyCustomInputDate = XMLDate.add days to(dummyCustomInputDate, 1)
addCustomDateAttribute(taskConfigurationRuleOutcome, "dummyCustomOutputDate", dummyCustomInputDate)
```

3.3 Output from Decision Function

The output for all task configuration decision functions is of the type TaskConfigurationRuleOutcome. This object holds as its attributes, the parameters needed for task configuration. The values for its attributes are set using the rules in the rulesets.

Figure 3–8 shows output definition of decision function.

Figure 3–8 SOA Composer – Viewing Output of a Decision Function



3.3.1 List of Configurable Attributes in Rule Outcome

The following human task attributes can be set in TaskConfigurationRuleOutcome object:

1. **Task Priority**

   The task priority can be set by assigning an integer value to the taskPriority attribute of the TaskConfigurationRuleOutcome object.For example, assign taskConfigurationRuleOutcome.taskPriority = 3

2. **Service Level Agreement (SLA)**

   SLA consists of taskExpirationDuration and taskDueDuration.

   To set SLA for the human task:

   - create a new Sla object using createSLA(String expirationDuration, String dueDuration)
   - assign it to taskConfigurationRuleOutcome.sla

     For example:

     assign taskConfigurationRuleOutcome.sla = createSLA("P5D", "P1D")

     expirationDuration and dueDuration are of the type xsd:duration encoded in String. The format of *xsd:duration is PnYnMnDTnHnMnS.*

     P is a literal value that starts the expression

     nY represents n years

     nM represents n months

     nD represents n days

     T is a literal value that separates date and time

     nH represents n hours

     nM represents n minutes

     nS represents n seconds

     In the example, we have an expiration duration of 5 days and due duration of 1 day. As another example, duration of 1 Month 15 days are represented by "P1M15D"

     For more details on the Duration type, please refer http://www.w3.org/TR/xmlschema-2/#duration

3. **Task Owner**

   Task owner can be set via following steps:

   - create a new ParticipantSet object using createTaskOwnerUser(String ownerUser) or createTaskOwnerGroup(String ownerGroup)
   - assign the new ParticipantSet object to taskOwner attribute of TaskConfigurationRuleOutcome

For example:

assign taskConfigurationRuleOutcome.taskOwner = createTaskOwnerUser("user1")

4. **Stage Participant**

Stage participant attribute, stageParticipant is of the type StageParticipant and it consists of following members:

- participant: A ParticipantSet object that holds the participant users and groups.
- filter: A UserFilterCriteria object that holds one or more than one UserAttributeFilterCriteria on which the users will be filtered. A criteria has an attribute name, attributeName, the value of which would determine the filter outcome, the reference value, attributeValue and one operator, out of equals, greater than, less than and in, which decided the type of comparison to be made between the actual value of the attribute and the reference value.

  Following steps are supposed to be followed for assigning the stageParticipant:

- Create a ParticipantSet using one of the following functions.
  - createParticipant(String groups, String users): ParticipantSet
  - createParticipantFromUsers(String users): ParticipantSet
  - createParticipantFromGroups(String groups): ParticipantSet
- Create a new UserFilterCriteria.
- Create UserAttributeFilterCriteria objects using CreateUserAttributeFilterCriteria(String attributeName, String attributeValue, UserAttributeFilterOperator operator) and add them to UserFitlerCriteria using addUserAttributeFilterCriteria(UserFilterCriteria filter, UserAttributeFilterCriteria attributeFilter)
- Create a new StageParticiant from the ParticipantSet and the UserFilterCriteria using the function createStageParticipant(ParticipantSet participant, UserFilterCriteria filter)
- Add StageParticipant to the TaskConfigurationRuleOutcome using addStageParticipant or addStageParticipantWithStageName

  Figure 3–9 shows an example for assigning a StageParticipant.

Figure 3–9 SOA Composer – Example for adding Stage Participant

```
assign new ▾  ParticipantSet participant = createParticipantFromGroups("Administrators")
assign new ▾  UserFilterCriteria filter = new UserFilterCriteria()
assign new ▾  UserAttributeFilterCriteria attributeFilter1 = createUserAttributeFilterCriteria("dummyAttributeName1","dummyAttributeValue1",UserAttrib
addUserAttributeFilterCriteria(filter,attributeFilter1)
assign new ▾  UserAttributeFilterCriteria attributeFilter2 = createUserAttributeFilterCriteria("dummyAttributeName2","dummyAttributeValue2",UserAttrib
addUserAttributeFilterCriteria(filter,attributeFilter2)
assign new ▾  StageParticipant stageParticipant = createStageParticipant(participant, filter)
addStageParticipant(taskConfigurationRuleOutcome, stageParticipant)
```

5. **Custom Output Attributes**

Following methods may be used in order to add custom text, number or date attributes, respectively to the TaskConfigurationRuleOutcome:

- addCustomTextAttribute
- addCustomNumberAttribute
- addCustomDateAttribute

Figure 3–10 shows example usage of custom attributes.

Figure 3–10 SOA Composer – Example usage of custom attributes

```
assign new ▾  String dummyCustomInputText = getCustomTextAttribute(customAttributes, "dummyCustomInputText")
assign ▾  dummyCustomInputText = dummyCustomInputText.toUpperCase()
addCustomTextAttribute(taskConfigurationRuleOutcome, "dummyCustomOutputText", dummyCustomInputText)
assign new ▾  int dummyCustomInputNumber = getCustomNumberAttribute(customAttributes, "dummyCustomInputNumber")
assign ▾  dummyCustomInputNumber = dummyCustomInputNumber + 7
addCustomNumberAttribute(taskConfigurationRuleOutcome, "dummyCustomOutputNumber" , dummyCustomInputNumber)
assign new ▾  XMLGregorianCalendar dummyCustomInputDate = getCustomDateAttribute(customAttributes, "dummyCustomInputDate")
assign ▾  dummyCustomInputDate = XMLDate.add days to(dummyCustomInputDate, 1)
addCustomDateAttribute(taskConfigurationRuleOutcome, "dummyCustomOutputDate", dummyCustomInputDate)
```

3.3.2 List of Functions to Set Rule Outcome

Following functions are available to configure the TaskConfigurationRuleOutcome:

1. **getTaskPrioriy(String priority): int**

   The input to this function is a number as a String and it returns the number as int type. For example, variable1.priority has a value of 2
   getTaskPriority(variable1.priority) will return 2.

   This method can be used when assigning the
   TaskConfigurationRuleOutcome.taskPriority

2. **createSLA(String expirationDuration, String dueDuration): Sla** This method takes as inputs the expiration duration and due duration as Strings written in xsd:duration format. Please refer to the description of xsd:duration in section 3.2.1 - 2 Service Level Agreement (SLA) for more details.

3. **createParticipantFromUsers(String users): ParticipantSet**

   Use this method to create ParticipantSet from a string containing user names separated with commas, that is, user1,user2,user3. The ParticipantSet can then be used as an input parameter to createStageParticipant function.

4. **createParticipantFromGroups(String groups): ParticipantSet**

   Use this method to create ParticipantSet from a string containing group names separated with commas, that is, group1,group2. The ParticipantSet can then be used as an input parameter to createStageParticipant function.

5. **createParticipant(String groups, String users): ParticipantSet**

   Use this method to create ParticipantSet containing users as well as groups. The first argument, groups, is a string containing group names separated with commas, that is, group1,group2, and the second argument, users, is a string containing user names separated with commas, that is,

user1,user2,user3. The ParticipantSet can then be used as an input parameter to createStageParticipant function.

6. **getCustomTextAttribute(CustomAttributes customAttributes,String attributeName): String**

   This method is used to fetch a text attribute from a CustomAttributes object. Its inputs are:

   - customAttributes: the CustomAttributes object from which attribute is to be fetched.
   - attributeName: string containing the name of attribute that is to be fetched.

   It returns the value for the specified attribute as a String.

7. **getCustomNumberAttribute (CustomAttributes customAttributes, String attributeName): int**

   This method is used to fetch a number attribute from a CustomAttributes object. Its inputs are:

   - customAttributes: the CustomAttributes object from which attribute is to be fetched
   - attributeName: string containing the name of attribute that is to be fetched

   It returns the value for the specified attribute as a int.

8. **getCustomDateAttribute (CustomAttributes customAttributes,String attributeName): XMLGregorianCalendar**

   This method is used to fetch a date attribute from a CustomAttributes object. Its inputs are:

   - customAttributes: the CustomAttributes object from which attribute is to be fetched.
   - attributeName: string containing the name of attribute that is to be fetched.

   It returns the value for the specified attribute as an XMLGregorianCalendar.

   XMLGregorianCalendar is the java representation for xml dateTime. For more information see, http://docs.oracle.com/javase/1.5.0/docs/api/javax/xml/datatype/XMLGregorianCalendar.html

9. **createUserAttributeFilterCriteria (String attributeName,String attributeValue,UserAttributeFilterOperator operator): UserAttributeFilterCriteria**

   This method constructs a new UserAttributeFilterCriteria object using the given parameters. Its inputs are:

   - attributeName: string containing name of the attribute on which the filter criteria is based on.
   - attributeValue: string containing the reference value with which the actual value of the attribute is compared to.
   - operator: userAttributeFilterOperator object specifying the operator to be used for comparison. The allowed values are GREATER_THAN, LESS_THAN, EQUALS and IN. Refer Figure 3-7 for the UML class diagram of UserAttributeFilterOperator and the related types.

10. **addUserAttributeFilterCriteria (UserFilterCriteriauserFilterCriteria, UserAttributeFilterCriteria UserAttributeFilterCriteria): UserFilterCriteria**

    This method adds a UserAttributeFilterCriteria object to the given UserFilterCriteria object. Its input are:

- userFilterCriteria: userFilterCriteria object to which the attribute filter criteria needs to be added.
- userAttributeFilterCriteria: userAttributeFilterCriteria object which needs to be added to userFilterCriteria. UserAttributeFilterCriteria can be created using the function createUserAttributeFilterCriteria.

11. **createStageParticipant (ParticipantSet participant,UserFilterCriteria userFilter): StageParticipant**

    This method is used to create a StageParticipant from ParticipantSet and a UserFilterCriteria, which are passed in as following parameters.

    - participant: participantSet object which can be created using any of the three functions createParticipant, createParticipantFromUsers or createParticipantFromGroups.userFilter:
    - userFilterCriteria object.

12. **addStageParticipant (TaskConfigurationRuleOutcometaskConfigurationRuleOutcome,StageParticipant stageParticipant):TaskConfigurationRuleOutcome**

    This method is used to add a StageParticipant to a TaskConfigurationRuleOutcome object.

    - taskConfigurationRuleOutcome: taskConfigurationRuleOutcome to which the stage participant is to be added.
    - stageParticipant: stageParticipant object which is added to taskConfigurationRuleOutcome. A StageParticipant can be created using createStageParticipant function

    The function addStageParticipantWithStageName can also be used to the same task, and it also has the capability of setting the stage name.

13. **createTaskOwnerUser(String ownerUser): ParticipantSet**

    This method is used to create a ParticipantSet, that is suitable to be set to TaskConfigurationRuleOutcome.taskOwner, from a string containing the owner user name, for example, user1. The ParticipantSet can then be assigned to TaskConfigurationRuleOutcome.taskOwner.

14. **createTaskOwnerGroup(String ownerGroup): ParticipantSet**

    This method is used to create a ParticipantSet, that is suitable to be set to TaskConfigurationRuleOutcome.taskOwner, from a string containing the owner group name, for example, group1. The ParticipantSet can then be assigned to TaskConfigurationRuleOutcome.taskOwner.

15. **addCustomTextAttribute (TaskConfigurationRuleOutcome ruleOutcome, String attrName,String attrValue): void**

    This method is used to add a custom text attribute to a TaskConfigurationRuleOutcome object. Its inputs are:

    - ruleOutcome: the TaskConfigurationRuleOutcome object to which attribute is to be added.
    - attrName: string containing the name of attribute that is to be added.

- attrValue: string containing the value of attribute that is to be added.

16. **addCustomNumberAttribute(TaskConfigurationRuleOutcome ruleOutcome,String attrName,int attrValue): void**

    This method is used to add a custom number attribute to a TaskConfigurationRuleOutcome object. Its inputs are:

    - ruleOutcome: the TaskConfigurationRuleOutcome object to which attribute is to be added.
    - attrName: string containing the name of attribute that is to be added.
    - attrValue: int containing the value of attribute that is to be added.

17. **addCustomDateAttribute (TaskConfigurationRuleOutcome ruleOutcome, String attrName,XMLGregorianCalendar attrValue): void**

    This method is used to add a custom date attribute to a TaskConfigurationRuleOutcome object. Its inputs are:

    - ruleOutcome: the TaskConfigurationRuleOutcome object to which attribute is to be added.
    - attrName: string containing the name of attribute that is to be added.
    - attrValue: XMLGregorianCalendar object containing the value of attribute that is to be added.

      XMLGregorianCalendar is the java representation for xml dateTime. For more information see, http://docs.oracle.com/javase/1.5.0/docs/api/javax/xml/datatype/XMLGregorianCalendar.html

18. **addStageParticipantWithStageNam (TaskConfigurationRuleOutcometaskConfigurationRuleOutcome,StageParticipant stageParticipant,StringstageName): TaskConfigurationRuleOutcome**

    This method is has a similar function as that of addStageParticipant and is used to add a StageParticipant to a TaskConfigurationRuleOutcome object and also specify a stage name.

    - taskConfigurationRuleOutcome: TaskConfigurationRuleOutcome to which the stage participant is to be added.
    - stageParticipant: StageParticipant object which is added to taskConfigurationRuleOutcome. A StageParticipant can be created using createStageParticipant function.
    - stageName: String containing the desired stage name.

4 Data Management

This chapter describes data related activities to be performed as an administrator.

4.1 Batch Execution

Batch Execution refers to bulk processing of records to perform business operations in real-time environment. Business operations include complex processing of large volumes of information, that is most efficiently processed with minimal or no user interaction using Batch Execution.

The batch process is run through the **End of Day (Fast path: EOD10)** page with a varied combination of category, job code and job type for a particular business day.

This section explains the steps involved in Batch Execution.

> **Note**
>
> To view the detailed procedure to be followed in the application page **End of Day (Fast Path: EOD10)**, see its context-sensitive help in the application.

4.1.1 Database Backup

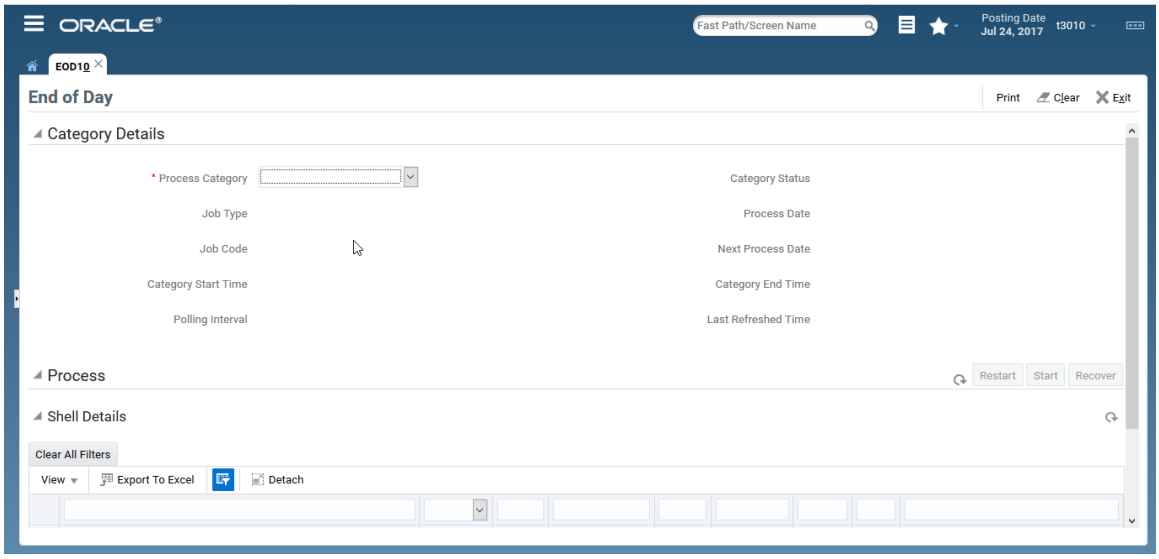
Perform Database Backup before starting with the Batch Execution.

4.1.2 Navigate to End of Day Page

To navigate to the End of Day page:

1. Log in to the Admin Application.
2. Navigate to *End of Day* page either by entering the Fast path **EOD10** or through the menu **Administration > End of Day.**

Figure 4–1 End of Day (Fast path:EOD10)



4.1.3 Cutoff Category Execution

This category marks the logical closure of business in the system to ensure that all online transactions during batch run get processed with the next process date.

To execute the Cutoff category:

1. Select the relevant **Category Details** as shown in the table below:

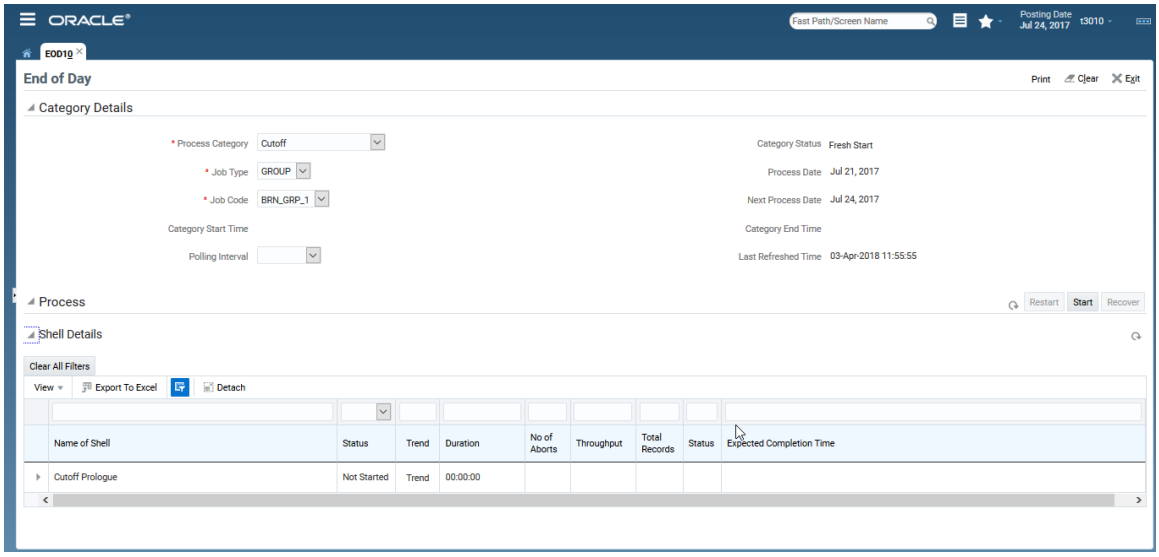
Process Category	Cutoff
Job Type	GROUP
Job Code	BRN_GRP_1

2. Click the **Refresh** button. The rest of the **Category Details** and the **Process Details** appear.

   Here, the **Shell State** is *Not Started.*

   The **Category Status** is *Fresh Start.*

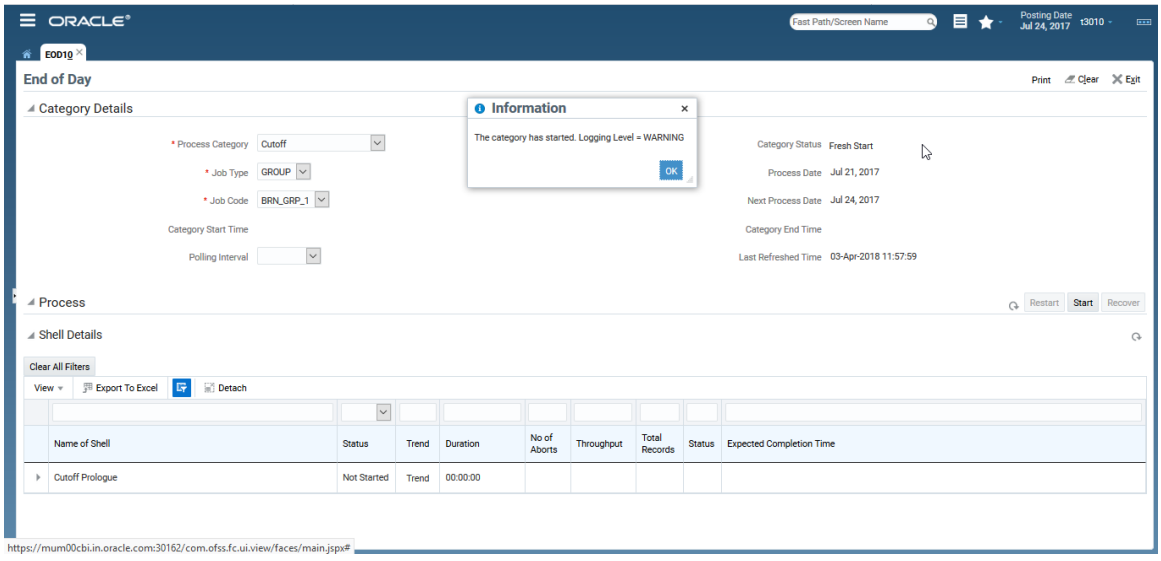
Figure 4–2 Cutoff Category - Not Started



3. Verify the **Process Date** and the **Next Process Date.**
4. Click the **Start** button to begin the execution.

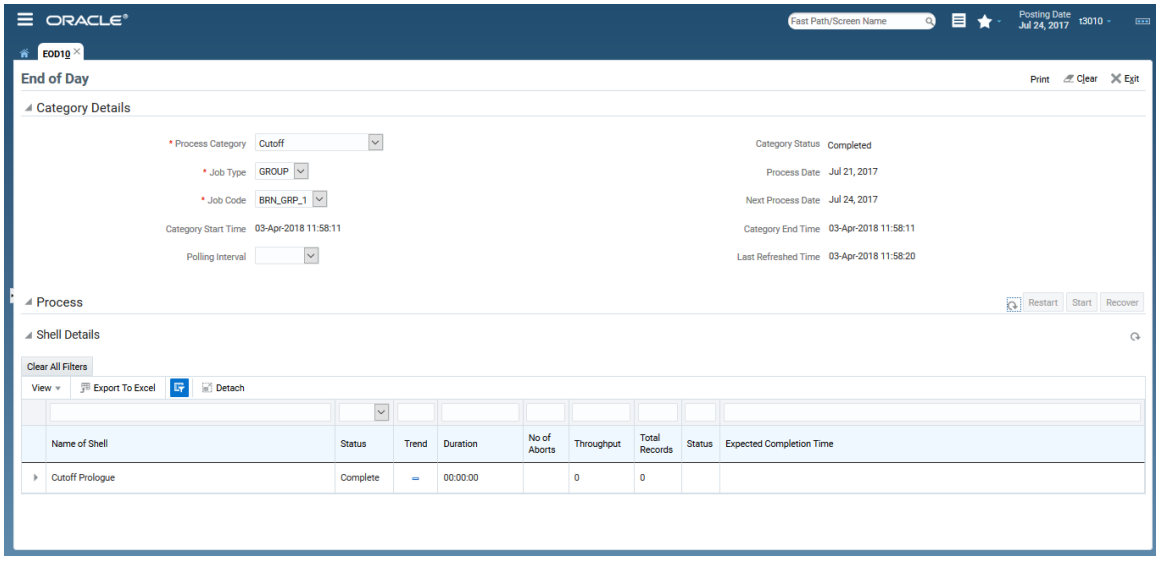
   Once the process starts the **Category Status** and the **Shell State** of currently running process display *In Progress*.

Figure 4–3 Cutoff Category - Start



5. On completion of the category, the **Category Status** and the **Shell State** of all the processes display *Completed*.

Figure 4–4 Cutoff Category - Complete



4.1.4 End of Day (EOD) Category Execution

This category performs the tasks required to mark closure of a business day in a bank. For example, value date cleaning, instruction expiry, auto disbursement instruction execution, bundle expiry, report generation and so on. Each task or transaction is performed by a shell in a predefined dependency and sequence.

To execute the End of Day category:

1. Select the relevant **Category Details** as shown in the table below:

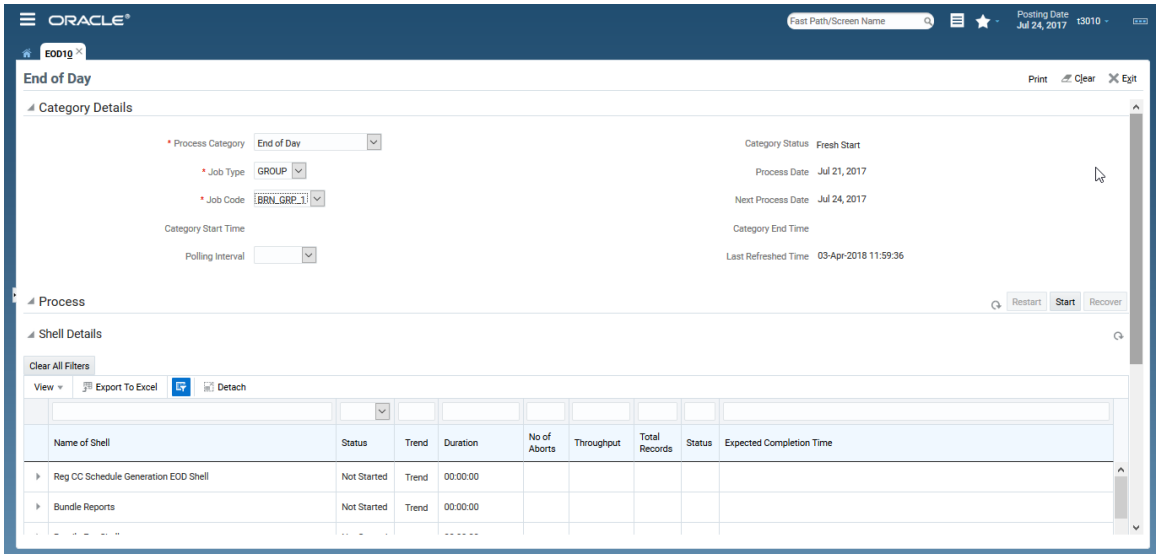
Process Category	End of Day
Job Type	GROUP
Job Code	BRN_GRP_1

2. Click the **Refresh** button. The rest of the **Category Details** and the **Process Details** appear.

   Here, the **Shell State** is *Not Started.*

   The **Category Status** is *Fresh Start.*

Figure 4–5 EOD Category - Not Started



3. Verify the **Process Date** and the **Next Process Date.**
4. Click the **Start** button to begin the execution.

   Once the process starts the **Category Status** and the **Shell State** of currently running process display *In Progress*.

Figure 4–6 EOD Category - Start

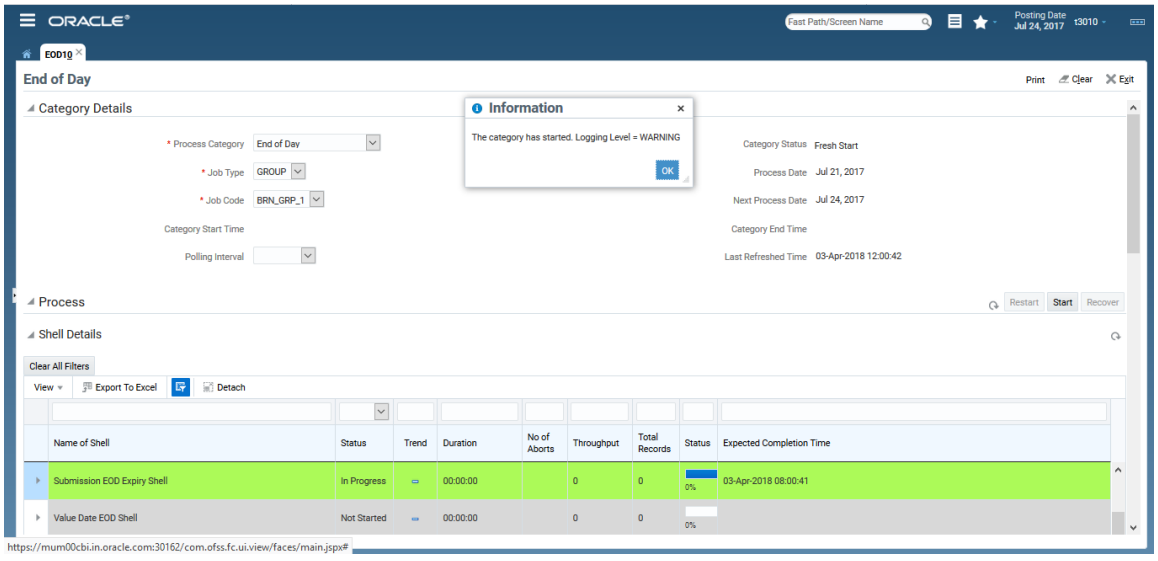
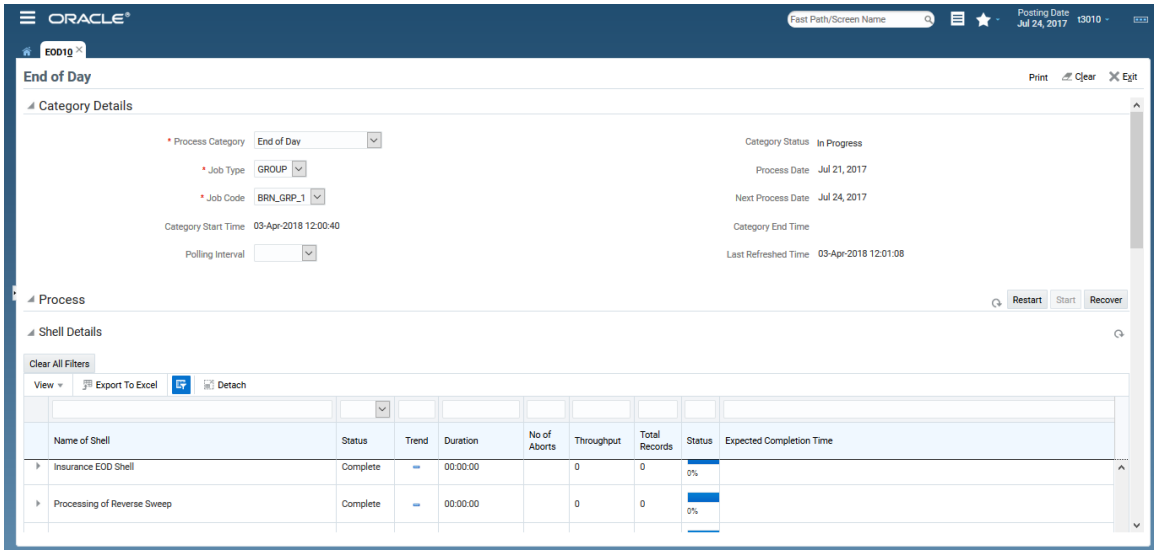
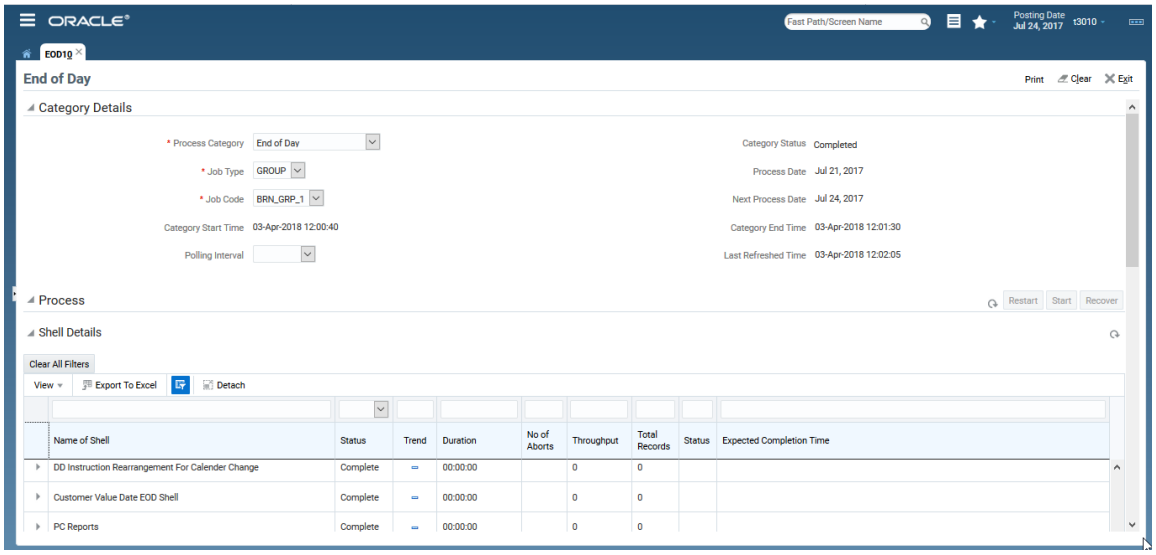


Figure 4–7 EOD Category - In Progress



5. On completion of the category, the **Category Status** and the **Shell State** of all the processes display *Completed.*

Figure 4–8 EOD Category - Complete



4.1.5 Internal System EOD Category Execution

This category performs interest accrual, interest capitalisation, interest compounding, accounting balance verification, ledger balance verification and update and related reporting.

To execute the Internal System EOD category:

1. Select the relevant **Category Details** as shown in the table below:

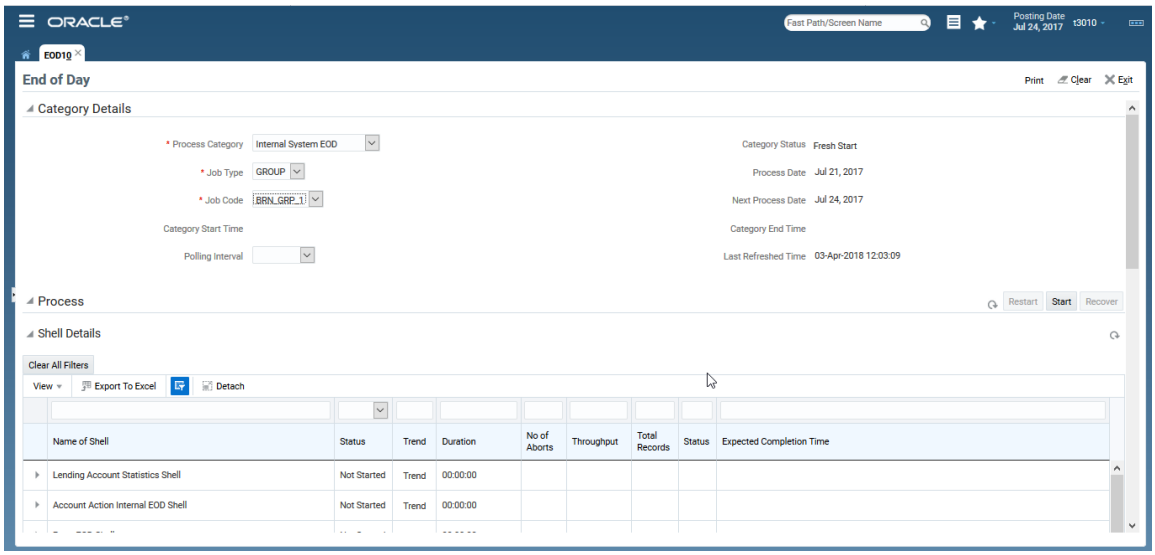
Process Category	Internal System EOD
Job Type	GROUP
Job Code	BRN_GRP_1

2. Click the **Refresh** button. The rest of the **Category Details** and the **Process Details** appear.

   Here, the **Shell State** is *Not Started.*

   The **Category Status** is *Fresh Start.*

Figure 4–9 Internal System EOD Category - Not Started



3. Verify the **Process Date** and the **Next Process Date.**

4. Click the **Start** button to begin the execution. Once the process starts the **Category Status** and the **Shell State** of currently running process display *In Progress.*

Figure 4–10 Internal System EOD Category - Start

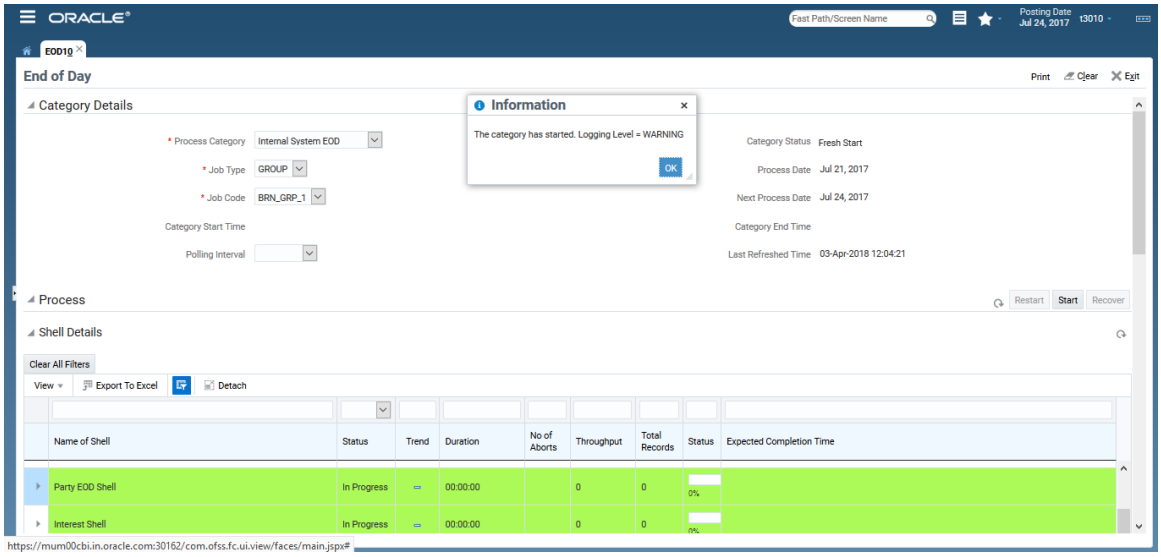
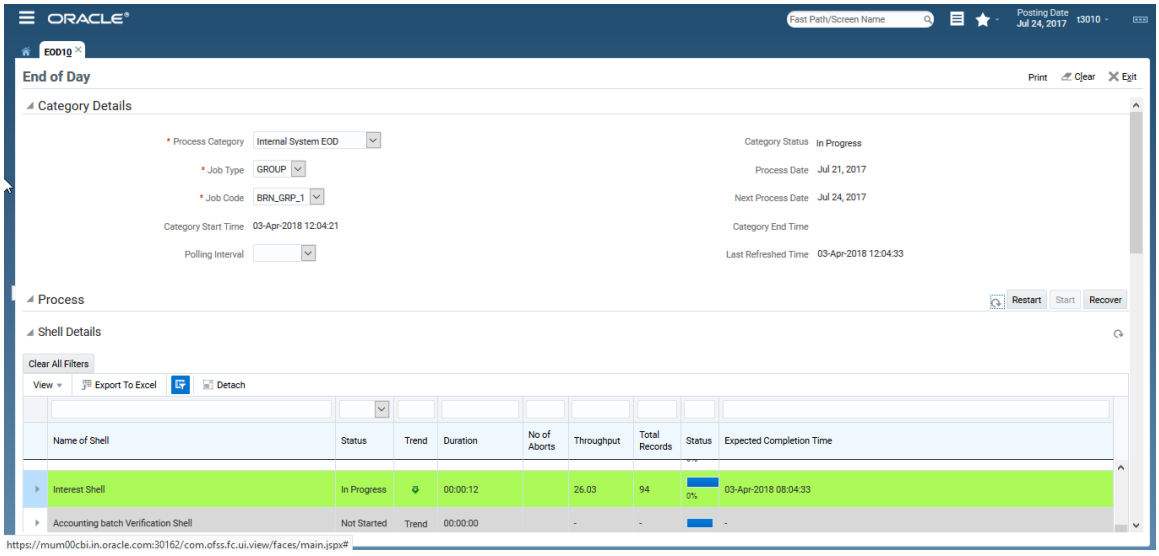
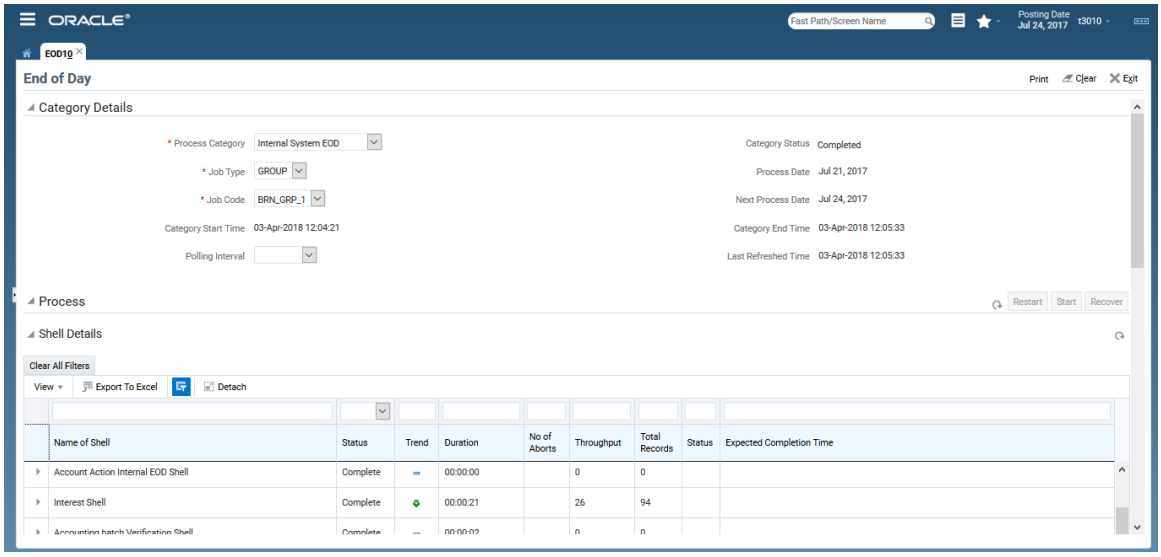


Figure 4–11 Internal System EOD Category - In Progress



5. On completion of the category, the **Category Status** and the **Shell State** of all the processes display *Completed.*

Figure 4–12 Internal System EOD Category - Complete



4.1.6 Beginning of Day (BOD) Category Execution

This category performs the tasks required for opening a business day in a bank. For example, standing instruction, sweepout instruction, loan account charging, periodic repayment instruction execution, period fee charging, and report generation. Each task or transaction is performed by a shell in a predefined dependency and sequence.

To execute the Beginning of Day category:

1. Select the relevant **Category Details** as shown in the table below:

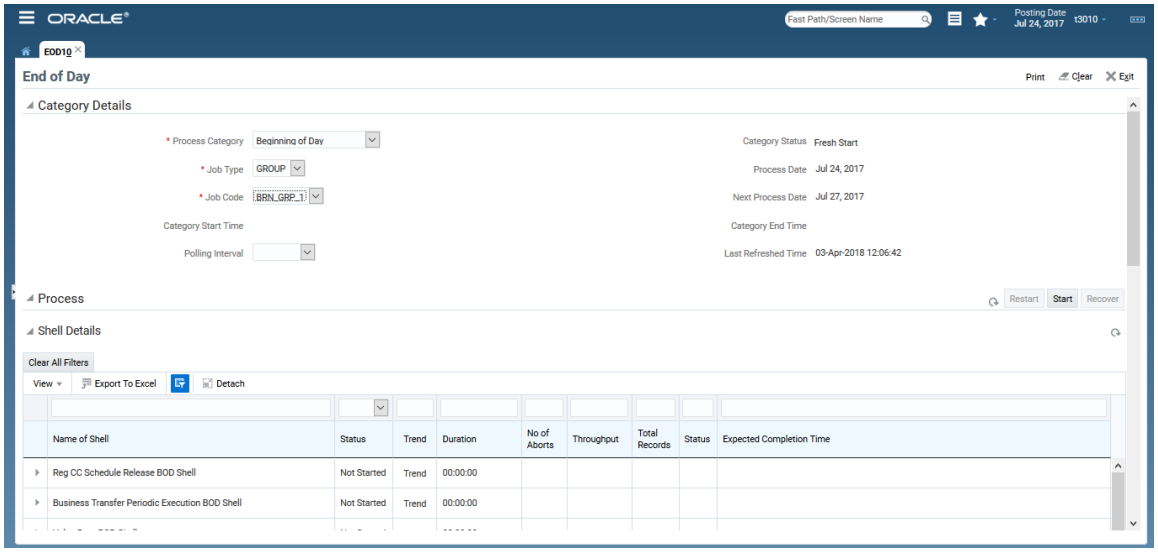
Process Category	Beginning of Day
Job Type	GROUP
Job Code	BRN_GRP_1

2. Click the **Refresh** button. The rest of the **Category Details** and the **Process Details** appear.

   Here, the **Shell State** is *Not Started.*

   The **Category Status** is *Fresh Start.*

Figure 4–13 BOD Category - Not Started



3. Verify the **Process Date** and the **Next Process Date.**
4. Click the **Start** button to begin the execution.

   Once the process starts the **Category Status** and the **Shell State** of currently running process display *In Progress*.

Figure 4–14 BOD Category - Started

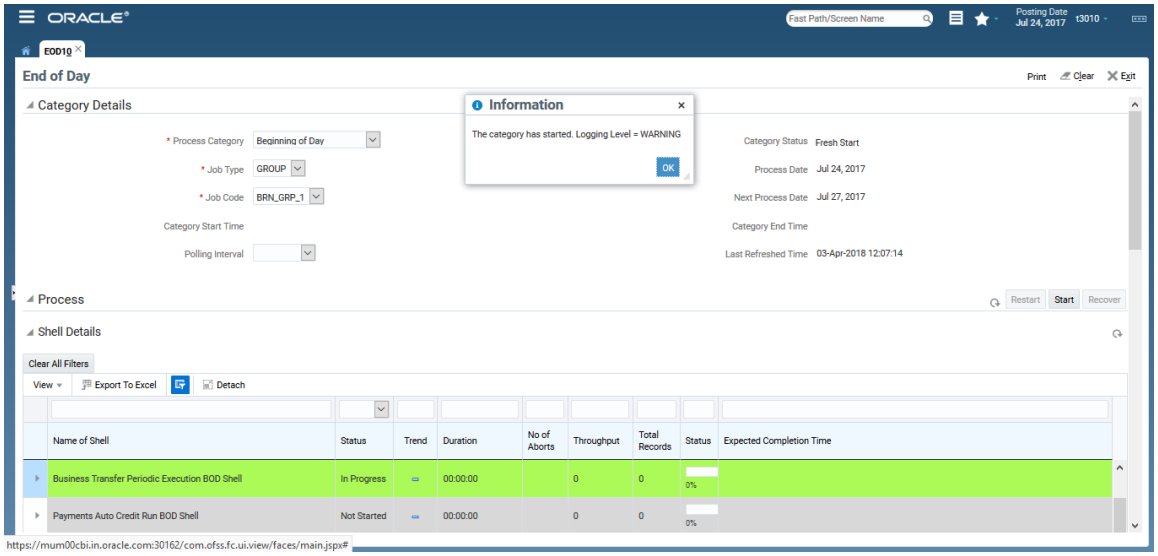
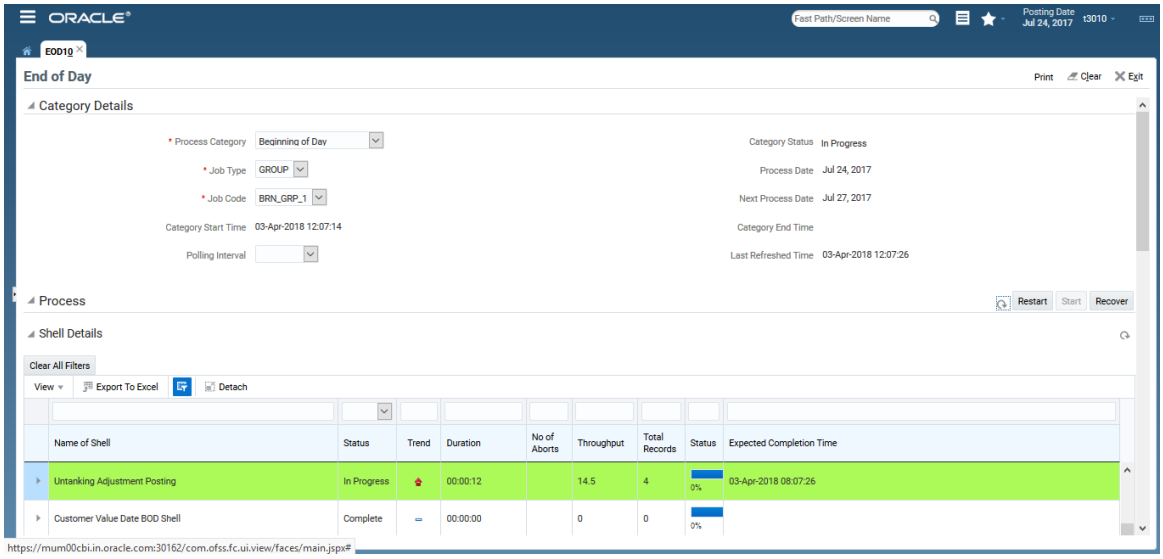
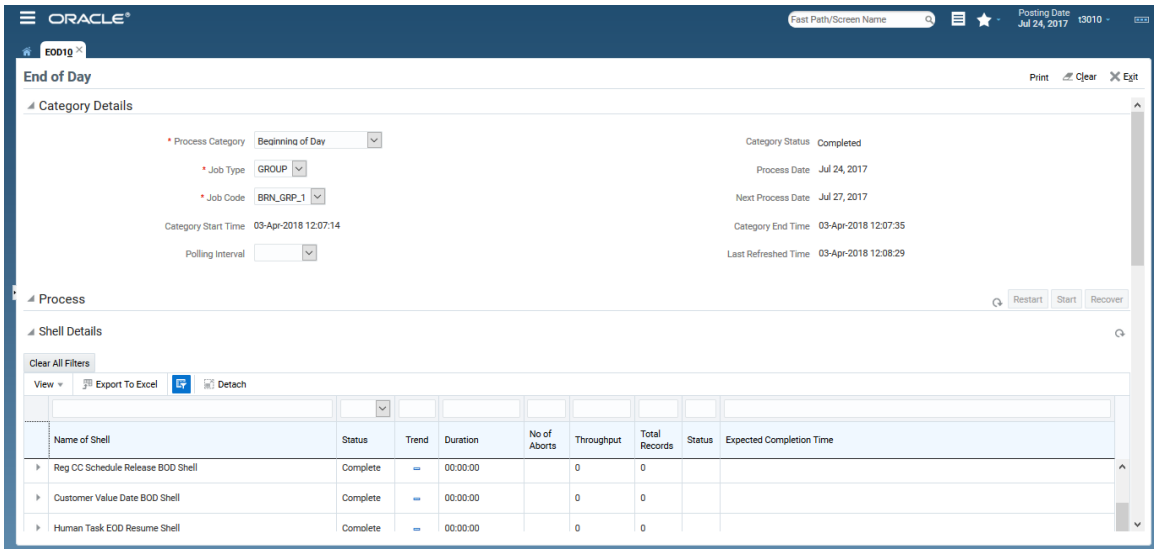


Figure 4–15 BOD Category - In Progress



5. On completion of the category, the **Category Status** and the **Shell State** of all the processes display *Completed*.

Figure 4–16 BOD Category - Completed



4.1.7 Housekeeping Category Execution

This category performs the tasks such as statement generation, alert generation, exposure tracking, offset benefit calculation, and facility closure.

To execute the Housekeeping category:

1. Select the relevant **Category Details** as shown in the table below:

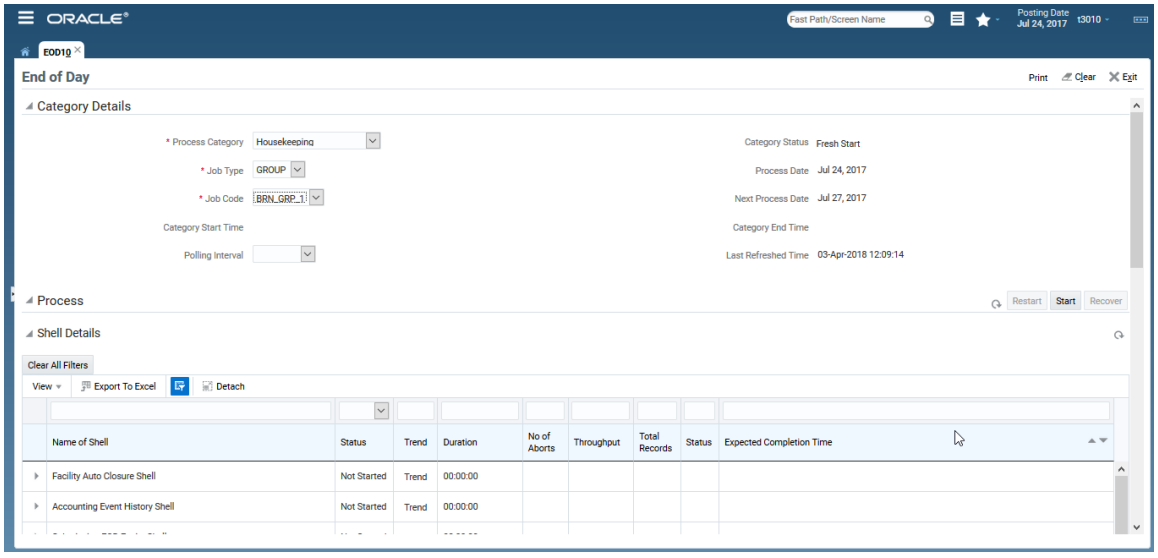
Process Category	Housekeeping
Job Type	GROUP
Job Code	BRN_GRP_1

2. Click the **Refresh** button. The rest of the **Category Details** and the **Process Details** appear.

   Here, the **Shell State** is *Not Started.*

   The **Category Status** is *Fresh Start.*

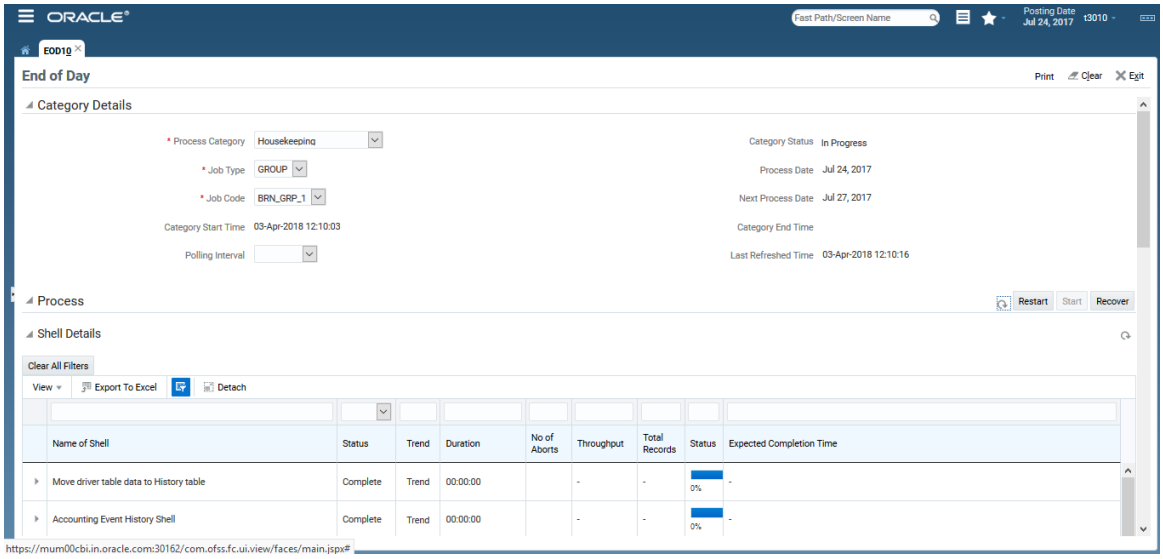
   Figure 4–17 Housekeeping Category - Not Started



3. Verify the **Process Date** and the **Next Process Date.**
4. Click the **Start** button to begin the execution.

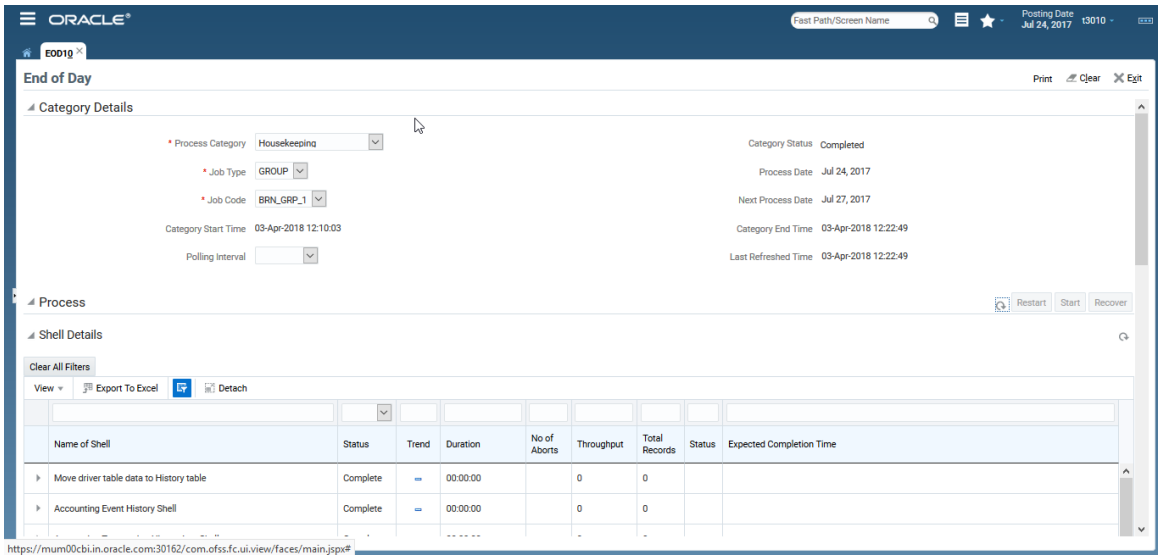
   Once the process starts the **Category Status** and the **Shell State** of currently running process display *In Progress*.

Figure 4–18 Housekeeping Category - In Progress



5. On completion of the category, the **Category Status** and the **Shell State** of all the processes display *Completed.*

Figure 4–19 Housekeeping Category - Completed



4.1.8 Alert Generation Category Execution

This category is used to generate previously logged alerts.

To execute the Alert Generation category:

1. Select the relevant **Category Details** as shown in the table below:

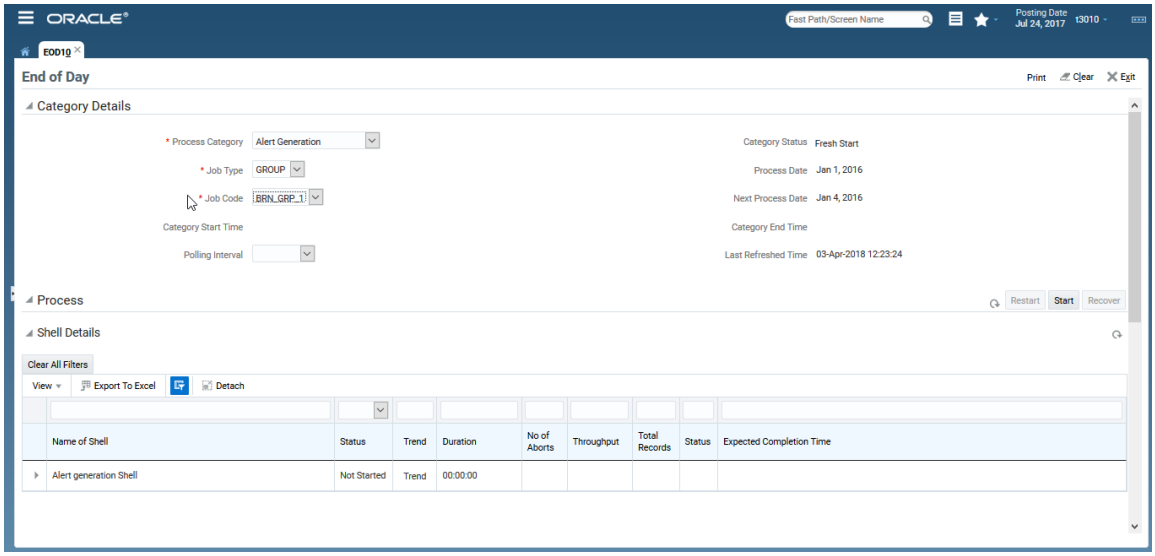
Process Category	Alerts Generation
Job Type	GROUP
Job Code	BRN_GRP_1

2. Click the **Refresh** button. The rest of the **Category Details** and the **Process Details** appear.

   Here, the **Shell State** is *Not Started.*

   The **Category Status** is *Fresh Start.*

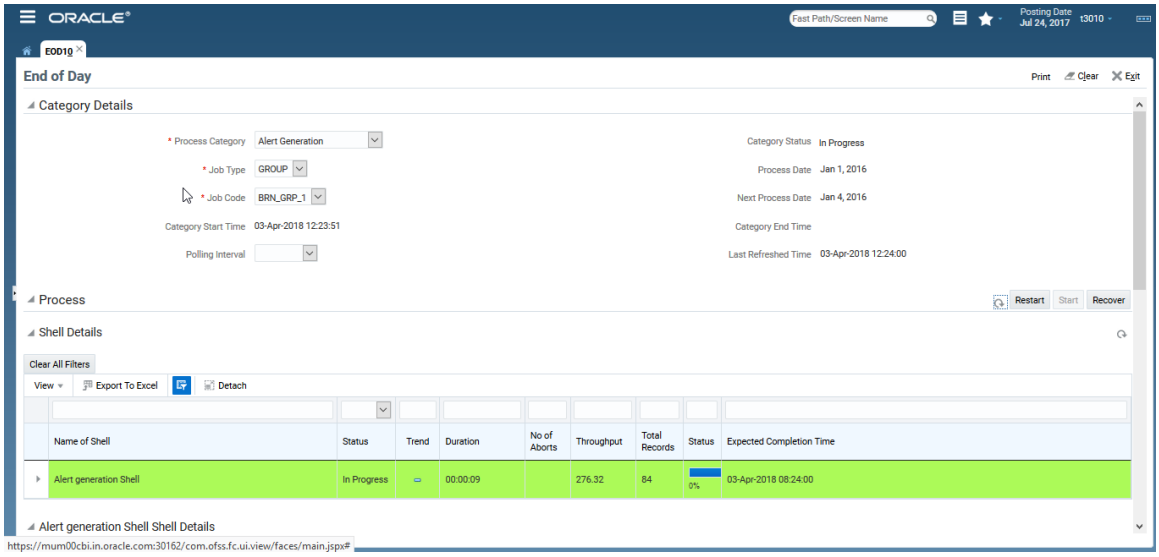
   Figure 4–20 Alert Generation Category - Not Started



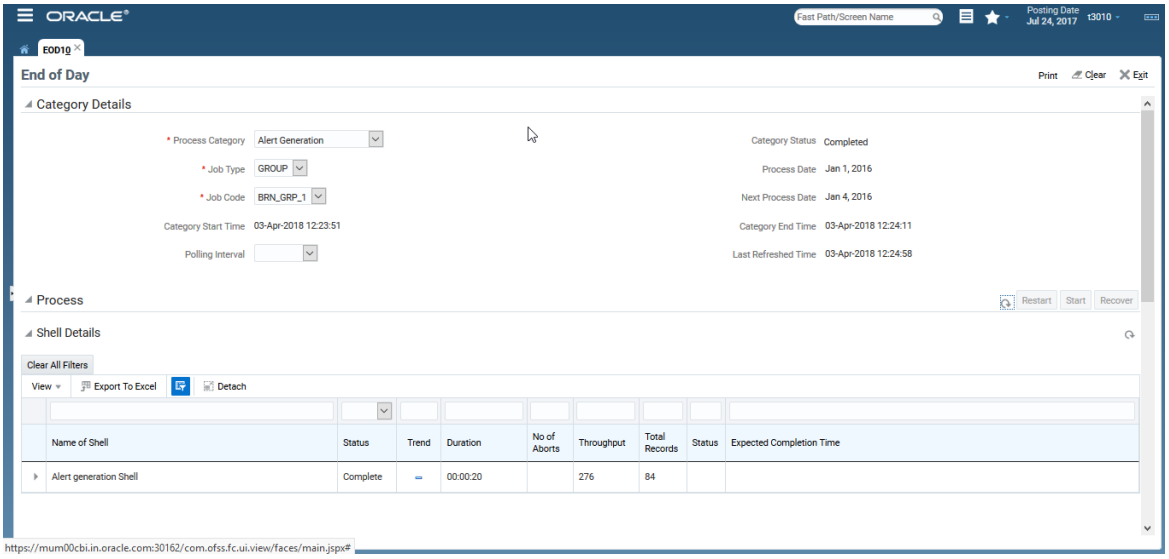
3. Verify the **Process Date** and the **Next Process Date.**
4. Click the **Start** button to begin the execution.

   Once the process starts the **Category Status** and the **Shell State** of currently running process display *In Progress*.

Figure 4–21 Alert Generation Category - In Progress



5. On completion of the category, the **Category Status** and the **Shell State** of all the processes display *Completed.*



4.2 Batch Exception Recovery

Batch Exception Recovery refers to mechanism to allow support and business users perform actions on the records that were skipped during batch execution. During batch execution, if the number of failures due to business exception is less than pre-configured threshold, such records are skipped for future processing.

The batch exception recovery can be done using the Batch Exception Recovery (Fast Path: OPA007) page. It is recommended that user in support or operations role, checks this page after every batch processing is completed for any PENDING records.

This section explains the steps involved in Batch Exception Recovery.

Batch exception recovery actions can be broadly classified in two categories:

- Actions for Support/Operations user (Performed on Batch Exception Recovery page)
- Actions for Business user (Viewed in worklist application and actioned using OBP screens, data patches.)

In its entire life cycle, the batch exception record will go through the above mentioned actions starting with PENDING and ending with either IGNORED or REPROCESSED. Support or Operations user acts on exception record using the Batch Exception Recovery page.

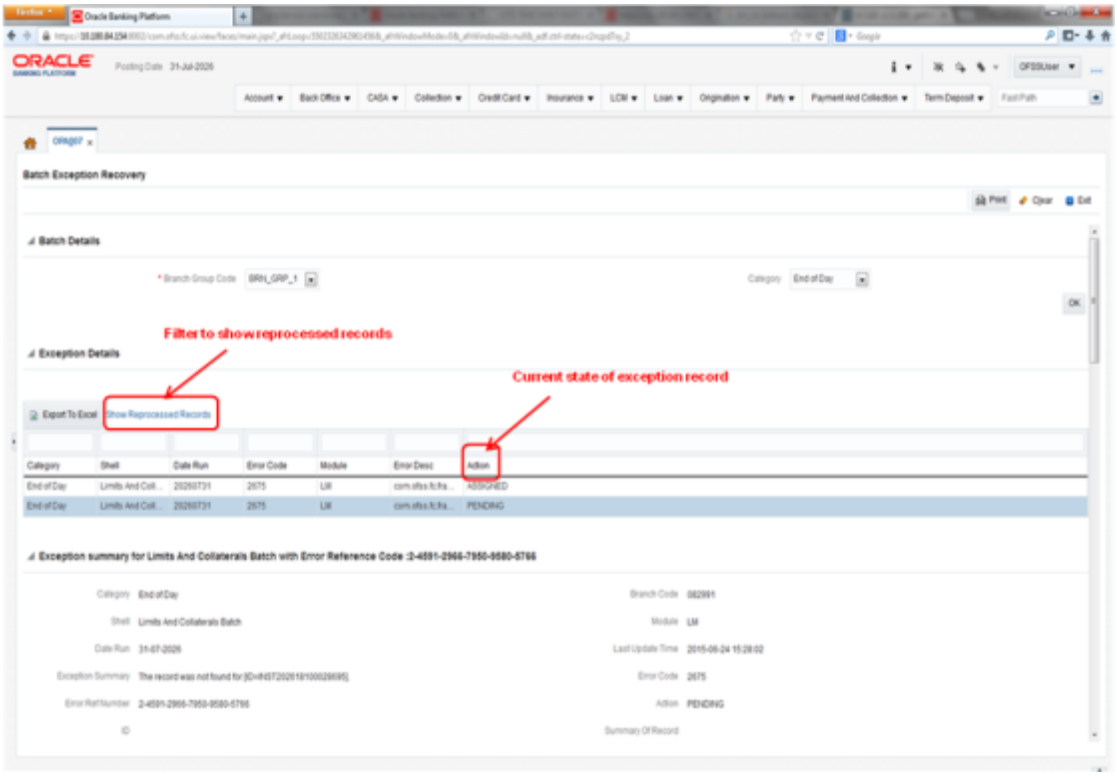
To navigate to the Batch Exception Recovery page:

1. Log in to the Admin application.
2. Navigate to Batch Exception Recovery page either by entering the Fast Path **OPA007** or through the menu **Administration** > **Batch Exception Recovery**.
3. Select the relevant Category Details as shown in the following table:

Branch Group Code	BRN_GRP_1
Category	End of Day

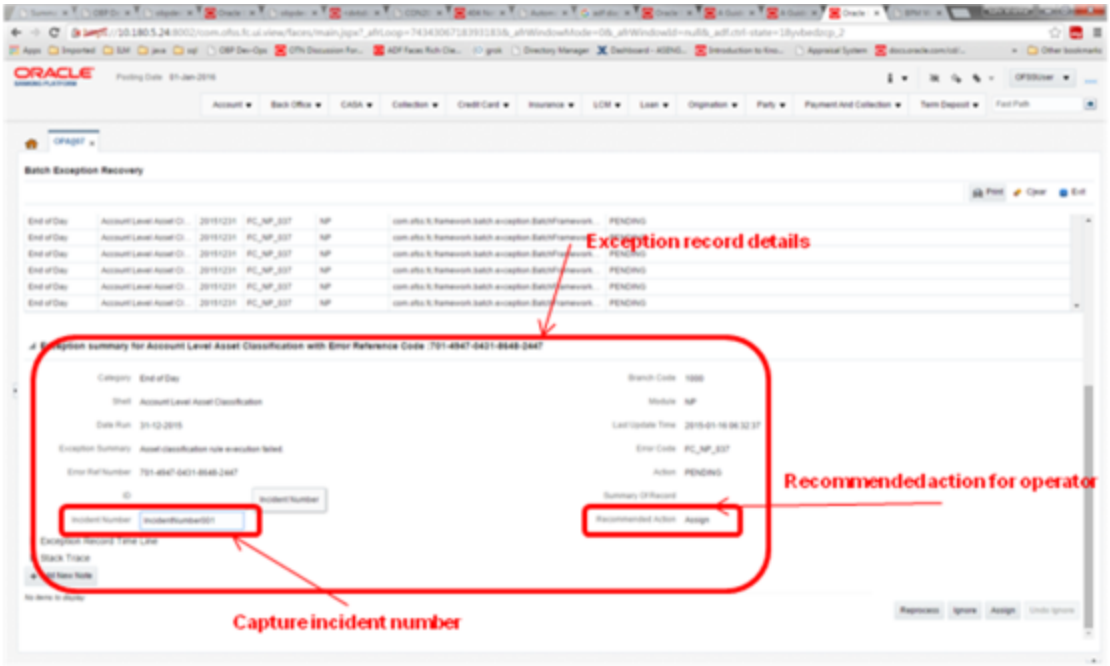
4. Click OK. The Exception Details appear.

Figure 4–22 Exception Details



5. Select an exception record. The additional details such as Stack Trace and Comments appear. One of the important attributes is Recommended Action for the operator.

Figure 4–23 Exception Record Details



6. Check the Stack Trace and Comments. It is recommended that if the current action on an exception record is PENDING and there are no Comments, click Assign to create a task for business user to take appropriate corrective actions on the exception record. Also, look for the Recommended Action. By looking at the Stack Trace, if the support or operator users find a similar previous incident, they can capture the same in the Incident Number field as shown in the above figure.

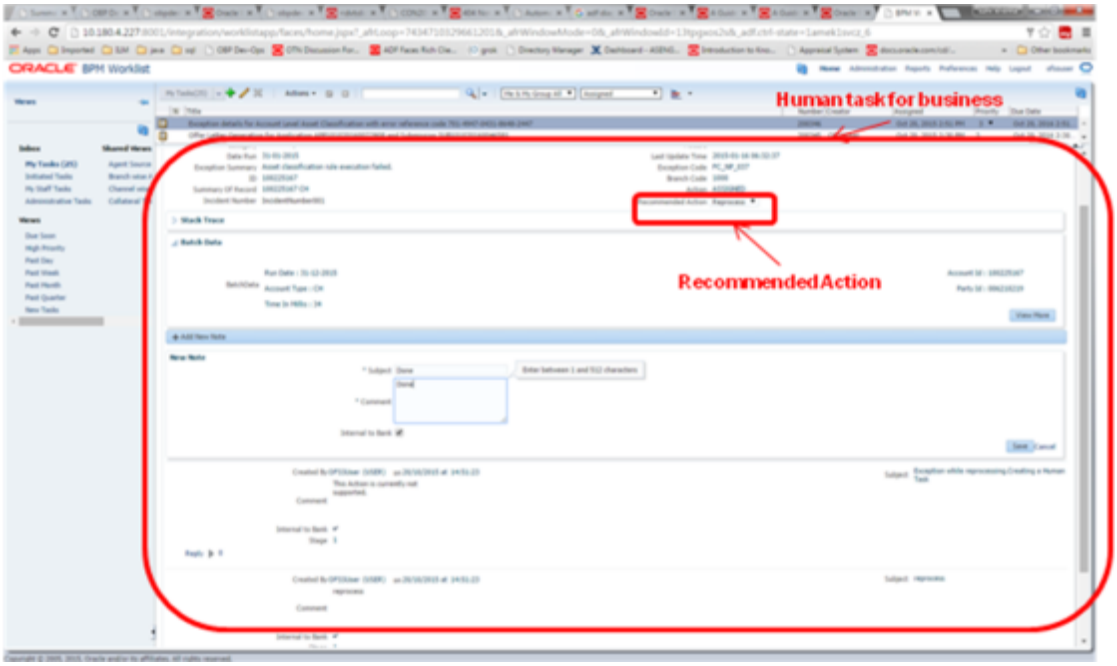
> **Note**
>
> It is mandatory to capture valid meaningful Comments while performing any action on the exception record.

Business user acts on exception record using the Worklist Application

To navigate to the Worklist page:

1. Log in to Worklist application.
2. All the exception records with action as ASSIGNED will appear as a task in the worklist. Select the task to act on.

Figure 4–24 Exception record in Worklist application



3. Click Claim to claim the task.
4. Check the Stack Trace, Batch Data, Incident Number and Comments.
5. Perform appropriate actions using application screens, data patches.
6. Capture comments in Comments section. These comments will be used by the support user to further act on the exception record.
7. Select the mandatory Recommended Action for the support or operator user.
8. Click Done to complete the task. The exception record moves into PENDING state and will be visible to the support user to take further action.

> **Note**
>
> It is mandatory to capture valid meaningful Comments while performing any action on the exception record.

5 Setting Up The Bank And Branch

This chapter provides the process of setting up the bank and the branch commonly referred to as the Day 0 setups.

5.1 Common Services Day 0 Setup

The Common Services setup includes the following sections.

5.1.1 Core Maintenances

Core Entity Services seek to define the broad parameters within which the rest of the application functions. The service defines the bank, the various modules of the application that the bank may want to introduce, the languages and the time zones it operates in, the core parameters and structures of its various branches. The core entity services are also used by each of the different modules, and provide a variety of support functions to them.

The following Core Maintenances must be completed as a part of bank and branch setup:

- Bank Codes (Fast path: CS01)
- Business Group (Fast path: CS02)
- Bank Parameters (Fast path: CS03)
- Branch Parameters (Fast path: CS06)
- Country Codes (Fast path: CS09)
- Financial Cycle (Fast path: CS10)
- Reason Codes (Fast path: CS16)
- State Codes (Fast path: CS17)
- Bank Policy (Fast path: CS26)
- Bank Policy Deviation Definition (Fast path: CS39)
- Questionnaire Maintenance (Fast Path: CS103)
- Section Maintenance (Fast Path: 104)

Note

To view the detailed procedure for each application page, see its context sensitive help in the application.

5.1.1.1 Head Office Setup

The Head Office branch creation is currently being done via seed data where the Branch Type is HO. Branch Type is a seed table with fixed values for all applicable branch types, that is uploaded to the application from the backend. After the creation of Head Office branch through seed data, you can proceed to create other branches from the application where the Branch Type is shown as a LOV (excluding HO).

The process to set up a head office branch is as follows:

1. Create a new bank code in the application through the page **Bank Codes (Fast path: CS01).**
2. Set up the new bank parameters through the page **Bank Parameters (Fast path: CS03).**
3. Modify the seed data for Branch Type to include the new bank code as HO and run the seed. Currently the seed will be for Bank Code 08. The head office branch is created via this seed data.
4. Proceed to create the other branches through the application using the page **Branch Parameters (Fast Path: CS06)**, that includes all branch types other than HO.

> **Note**
>
> To view the detailed procedure for each application page, see its context-sensitive help in the application.

5.1.2 Currency Maintenances

The Currency Services are a part of the common services of Oracle Banking Platform and serve to record and retrieve the various currency related information.

The following Currency Maintenances must be completed as a part of bank and branch setup:

- Currency Codes (Fast path: CY01)
- Amount Text (Fast path: CY02)
- Currency Pairs (Fast path: CY03)
- Currency Branch Parameters (Fast path: CY04)
- Currency Denomination (Fast path: CY05)
- Currency Rate Types (Fast path: CY06)
- Exchange Rates (Fast path: CY07)

> **Note**
>
> To view the detailed procedure for each application page, see its context-sensitive help in the application.

5.1.3 Calendar Maintenances

The calendar services are embedded in the common services and serve to record and retrieve the various holidays of the bank in a calendar year.

The following Calendar Maintenances must be completed as a part of bank and branch setup:

- Holiday Rule Maintenance (Fast Path: CAL01)
- Calendar Type Maintenance (Fast Path: CAL02)
- Adhoc Calendar Maintenance (Fast path: CAL03)

> **Note**
>
> To view the detailed procedure for each application page, see its context-sensitive help in the application.

5.2 Other Setups

Following are the required setups:

- Alert Subscription (Fast Path: AL05)
- Alert Support (Fast Path: AL10)
- Message Template (Fast Path: AL03)
- Alert Maintenance (Fast Path: AL04)
- Risk Indicators Impact Cross-Reference (Fast Path: ACCT010)
- Document Inserts(Fast Path: CNM11)
- Document Type Definition (Fast Path: CNM01)
- Document Template Resolution Policy (Fast Path: CNM09)
- Document Search And Upload (Fast Path: CNM06)
- Document Category Definition (Fast Path: CNM02)
- Document Policy Definition (Fast Path: CNM03)
- Work Item Inquiry (Fast Path: WL001)
- Artifact Dependency Map (Fast Path: SM500)
- Policy Management (Fast Path: SM502)
- Rule Author (Fast Path: RL001)
- RuleSet (Fast Path: RULE01)
- Rule Search (Fast Path : RL003)
- Filter Definition (Fast Path: RL005)
- Expression Builder (Fast Path: RL004)
- Rule Simulator (Fast Path: RL002)

Note

To view the detailed procedure for each application page, see its context-sensitive help in the application.

6 Application Monitoring Using Administration Application

This chapter provides an overview on the various monitoring operations performed as an administrator using Administration application.

6.1 Dynamic Monitoring Service (DMS)

The aim is to monitor different channels involved in performing transactions with OBPM. The monitoring parameters consists of channels, services, trends (current behavior of execution), and time metrices. The monitoring is performed by DMS (Dynamic Monitoring Service).

What is DMS?

The Oracle Dynamic Monitoring Service (DMS) provides a set of Java APIs that measure and report performance metrics, trace performance and provide a context correlation service for Fusion Middleware and other Oracle products. Along with the APIs, DMS provides interfaces to enable application developers, support analysts, system administrators, and others to measure application-specific performance information.

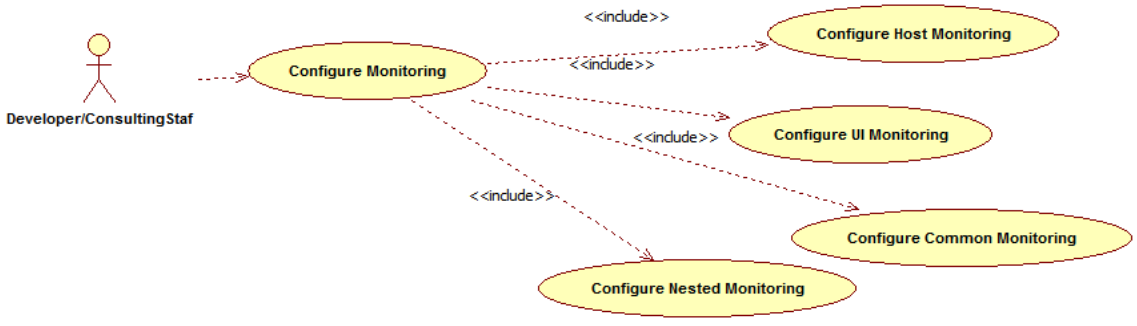
6.1.1 Usage

The usage of DMS is defined by the role of the user. Based on their roles, users can either take part in configuration of services for DMS or monitor the statistics collected via DMS.

Developers

These are the set of people who configure the monitoring services that are the part of OBPM system. The configuration can be made either for available services or for new services.

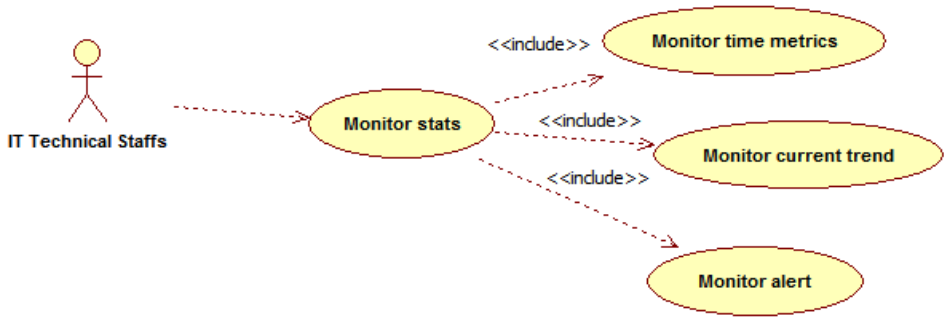
Figure 6–1 Developers



IT Technical Staff

This consists of set of people who monitor the DMS statistics generated for the service. With the help of various metrics generated they can analyze the behaviour of the target service. For example, 'time taken to execute' service could indicate need of optimization of the service.

Figure 6–2 IT Technical Staff



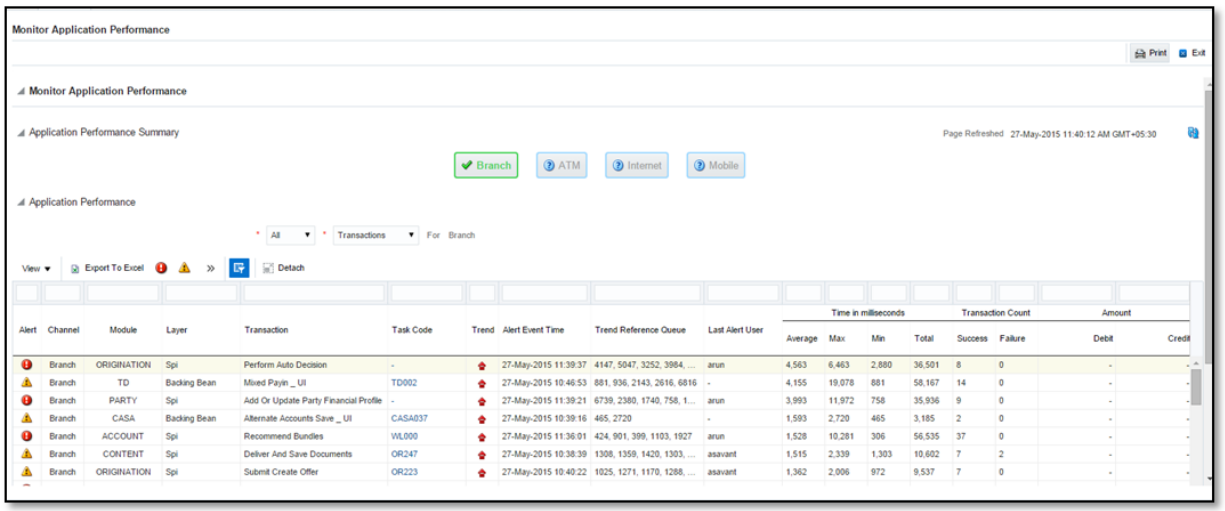
6.1.2 Monitoring Application using the OPA001 page

Once DMS statistics are captured for a particular channel and transactions involving it, it requires a UI representation to understand the statistics in a readable form so that one can analyse the behaviour. The monitoring activities are mainly carried out by IT Technical staff.

6.1.2.1 Monitoring Application Performance (Fast path: OPA001)

This page gives the monitoring statistics of different channels and the transactions occurring through it. It gives the time metric of the transactions, trend of the current transactions, and alert for the channel.

Figure 6–3 Monitoring Application Performance



The overall page can be subdivided in to 3 sub parts on the basis of information they provide:

6.1.2.1.1 Application Performance Summary

This section gives the information about the different channels of OBPM through which transactions are taking place. The information is about the health and active channels. The Refresh Button on top of this section gets the latest (refreshed) metrics.

Figure 6–4 Application Performance Summary

Following are the few notification about the channels:

- Denotes transactions not present for the channel
- Denotes normal status that is, the number of alerts are less than the specified limit
- Denotes warning status that is, the number of alerts are in the warning range
- Denotes critical status that is, number of alerts exceeds the limit

6.1.2.1.2 Log Level

This section gives logger level information for the host and UI server.

Figure 6–5 Log Level

6.1.2.1.3 Application Performance

This section gives the metrics for the transaction. Metrics include timing, alert, trending information. Certain filters can be applied over the metric table. Initially only 100 (Initial page size which is configurable) transactions are displayed. To display all the transactions, click the ALL button.

Trend

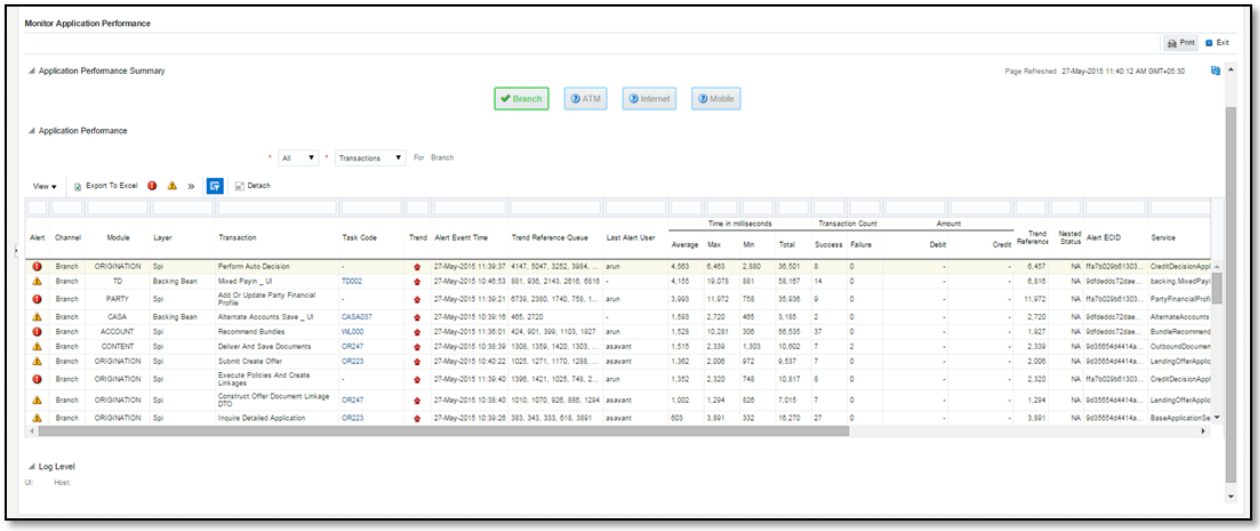
Indicates trending of execution timings of transaction. It is calculated by algorithm namely, Exponential Moving Average where if the execution time goes above the specified limit which is calculated by adding average execution time of the transaction and allowed limit (varies logarithmically to execution time); the transaction is considered as trending upwards and vice-versa for downwards trend.

However, if the execution time is with the range, trend is considered as neutral.

Alert

Indicates alerting state of the transaction. A transaction is given weight based on its properties namely, transaction type, timing category and module. The weight gives the offset allowed for transaction execution time. If the current transaction time is greater than average transaction time + offset, it is marked as alert. Initially it is marked as 'Critical' and after sometime the state is marked as 'Warning'.

Figure 6–6 Alert State



The table below explains each column of the table present in the given snapshot.

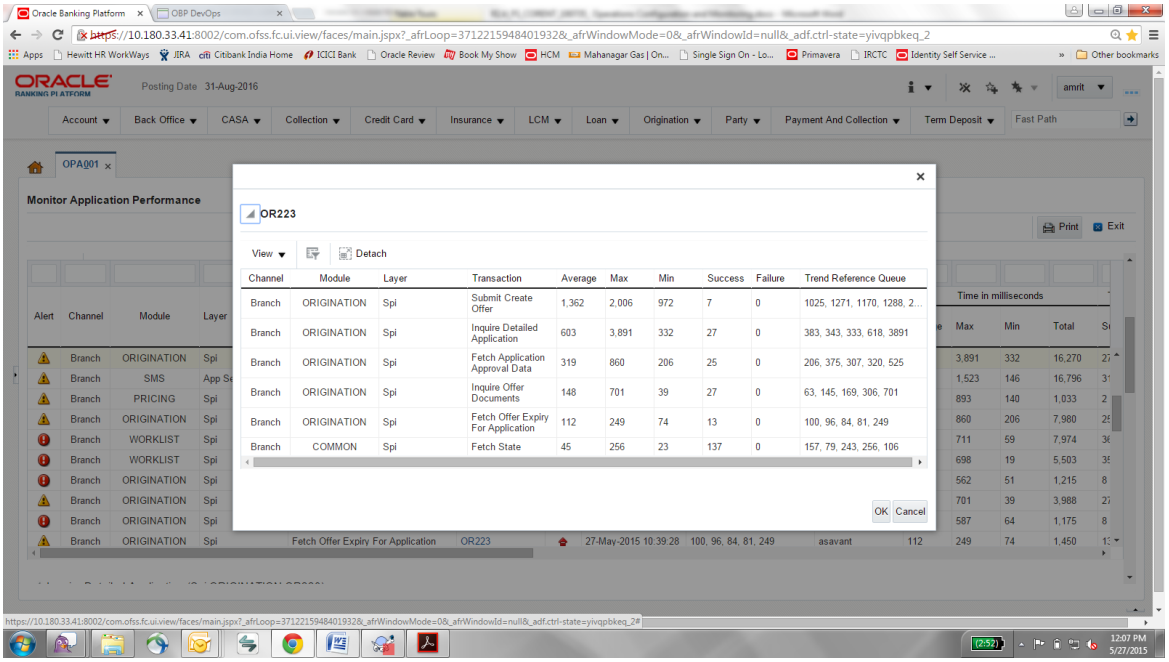
Table 6–1 Alert State

Sr. no.	Column Name	Description
1	Alert	Alert state of the transaction Valid Values: BLANK: No alert, Warning: Alert in past (default 5 minutes), Critical: Alerted Transaction
2	Channel	Channel through which the transaction occurred Valid Values: Branch, ATM, and POS.
3	Module	Application module of which transaction is a part
4	Layer	Configured Noun generation layer. Backing Bean for UI and Spi and App Service for Host.
5	Transaction	Name of the transaction
6	Task Code	Task code of the application page by which the transaction was triggered
7	Trend	Trending of transaction Valid Values: Upwards, Downwards, Neutral
8	Alert Event Time	Time at which last alert occurred for the transaction
9	Trend Reference Queue	Execution time of last n transactions (n=5)
10	Last Alert User	Teller who performed the last alerted transaction
11	Average Time	Average execution time
12	Max Time	Maximum time of execution of the transaction

Sr. no.	Column Name	Description
13	Min Time	Minimum time of execution of the transaction
14	Total Time	Total time of execution
15	Success Count	Number of times transaction executed successfully
16	Failure Count	Number of times transaction failed.
17	Debit Amount	Amount debited after transaction
18	Credit Amount	Amount credited after transaction
19	Trend Reference	Execution time of last transaction
20	Nested Status	Nested Status
21	Alert ECID	ECID of the last alerted transaction
22	Service	Service name of the transaction
23	Completed Operations	Number of completed transactions
24	Active Threads	Active Threads
25	Max Active Threads	Maximum active threads
26	Host	Host name
27	Process	Process Name
28	Server Name	Server name
29	App Root Type	Root type of noun
30	Failure Security Event	Failure due to security error
31	2FA Event	Authentication error
32	Failure Database Event	Failure due to database error
33	Failure Technical Event	Failure due to technical error
34	Failure Outbound Event	Failure due to outbound call (call outside OBPM)

One can select any of the task code which opens a popup with information about that task code only.

Figure 6–7 Select Task Code



Detailed Transaction View

This section gives the detailed view of a selected transaction. The desired transaction can be selected from the table (metric table). Click on any row to display a detailed view of the transaction.

Figure 6–8 Selection of Desired Transaction

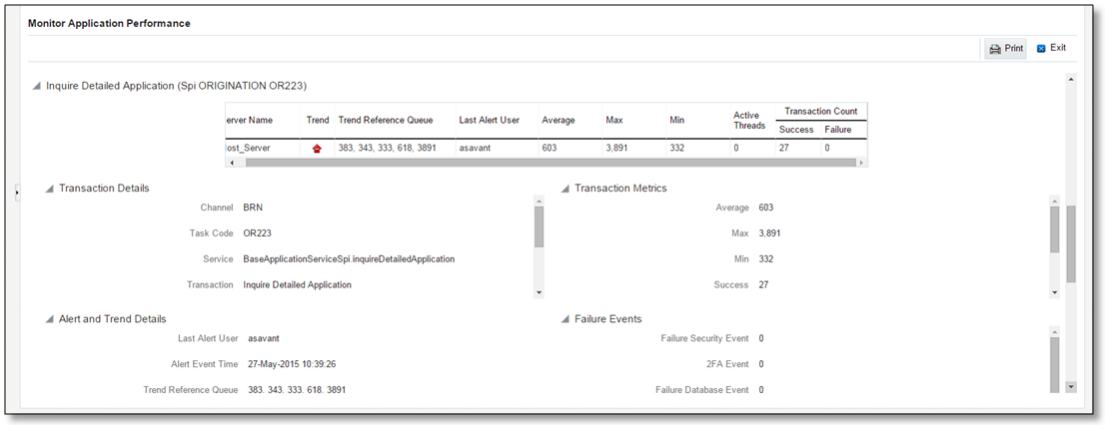


Figure 6–9 Transaction Details

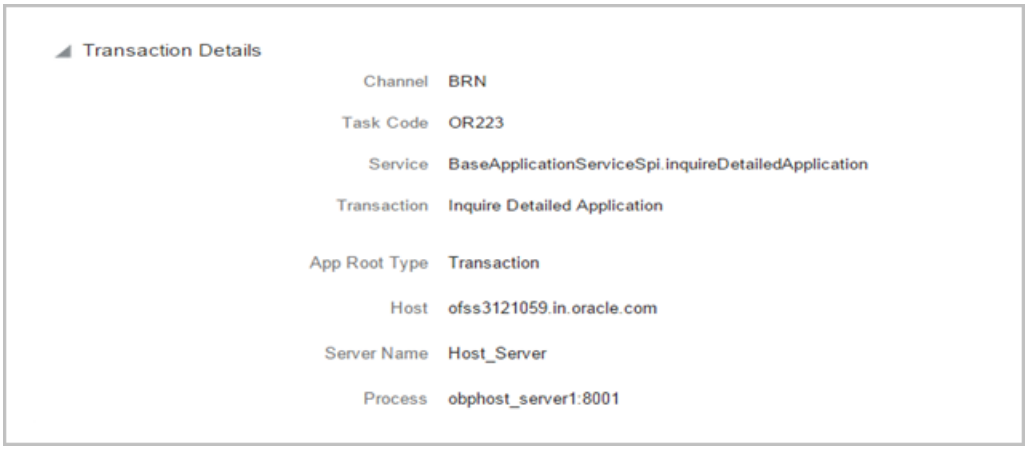


Figure 6–10 Transaction Metrics

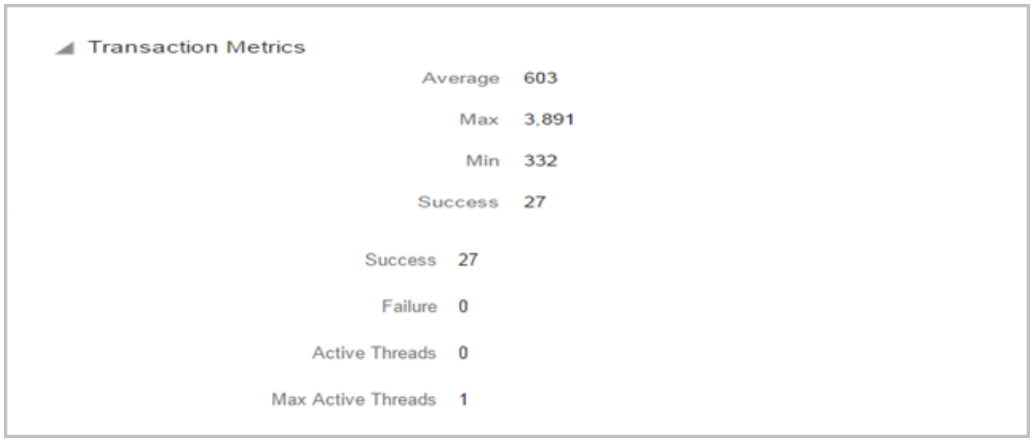


Figure 6–11 Alert and Trend Details

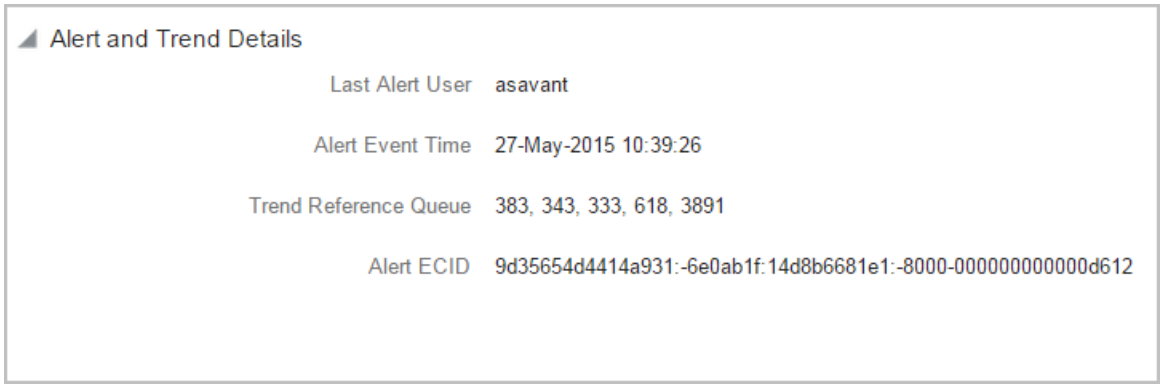


Figure 6–12 Failure Events

Configurations

The below mentioned configurations can be made in DMSConfig.properties:

- **Channel Status:** Number of alerts for which the channel shows 'Critical and 'Warning' status can be configured
- **Alert Status:** The time after which a 'Critical' alert changes to 'Warning' is configurable
- **Initial Page Size:** Every time host data is fetched only rows equal to page size are displayed. The page size is configurable

These configurations can be made in DMSConfig.properties.

6.2 Batch Performance Monitoring

Most of the enterprise applications would require bulk processing of records to perform business operations in real time environments. These business operations include complex processing of large volumes of information that is most efficiently processed with minimal or no user interaction. Such operations would typically include time based events (for example, month-end calculations, notices or correspondence), periodic application of complex business rules processed repetitively across very large data sets (for example, rate adjustments). Batch monitoring includes monitoring of all such batch processes. These batch processes generate huge statistics, which needs to be monitored in order to understand and improve its performance. OPA003 page is used to monitor these processes in detail along various metrices like duration, throughput, aborts, and so on.

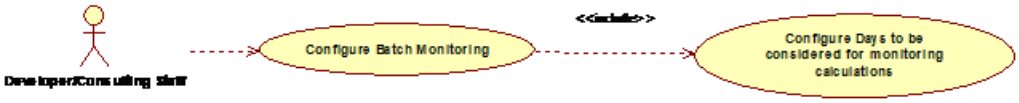
6.2.1 Use Cases

The overall use cases for the whole Batch monitoring operation are divided into two units on the basis of actor that works over batch monitoring operations. The different actors along with their use cases are as below:

Developers

These are the set of people who configure the monitoring services that are the part of OBPM system. The configuration can be made in the properties file BatchStatistics.properties. Configuration include the number of previous batch runs to be considered for calculation for monitored metrics.

Figure 6–13 Developers



IT Technical Staff

This consists of set of people who monitor the Batch statistics generated during the batch run.

Figure 6–14 IT Technical Staff

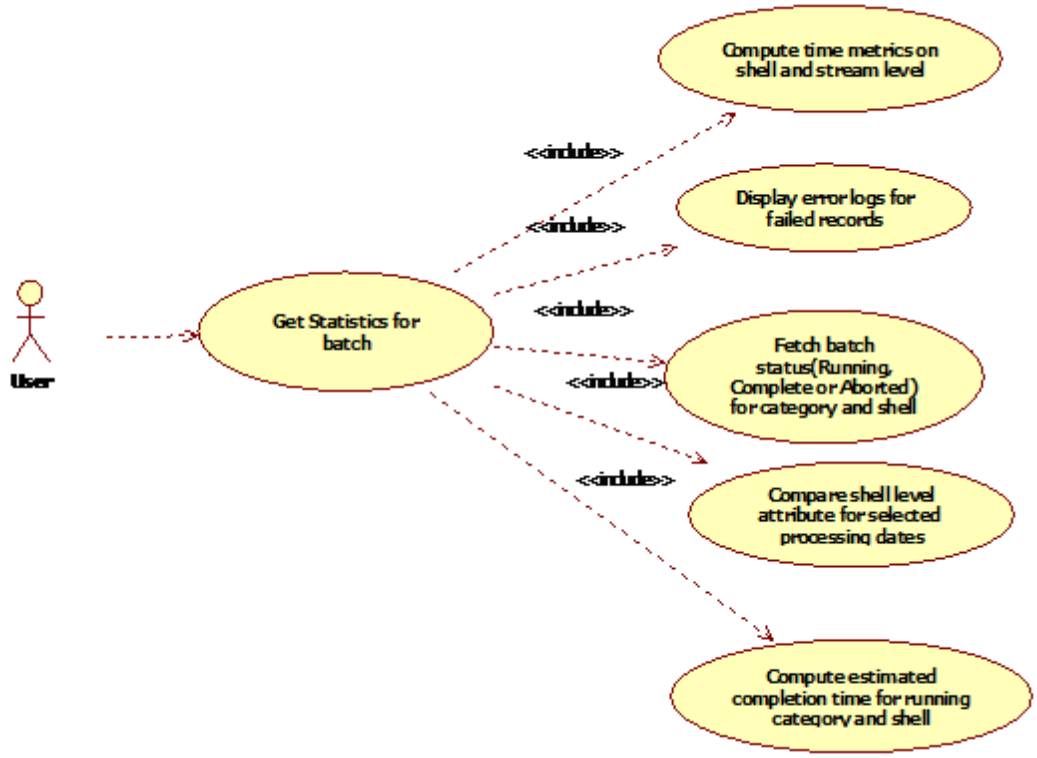


Figure 6–15 IT Technical Staff - Monitor Batch Stats

6.2.2 Monitoring Batch Performance Using OPA003 page

Once batch starts it needs UI representation to easily understand and interpret the batch stats. By monitoring these stats, one can understand the bottle necks of the batch process and hence can work in a way to improve batch performance.

6.2.2.1 Monitor Batch Performance (Fast path: OPA003)

This page takes category, job code, job type, and processing date as input and provides monitoring stats for shells running for selected category.

Figure 6–16 Batch Performance Monitoring

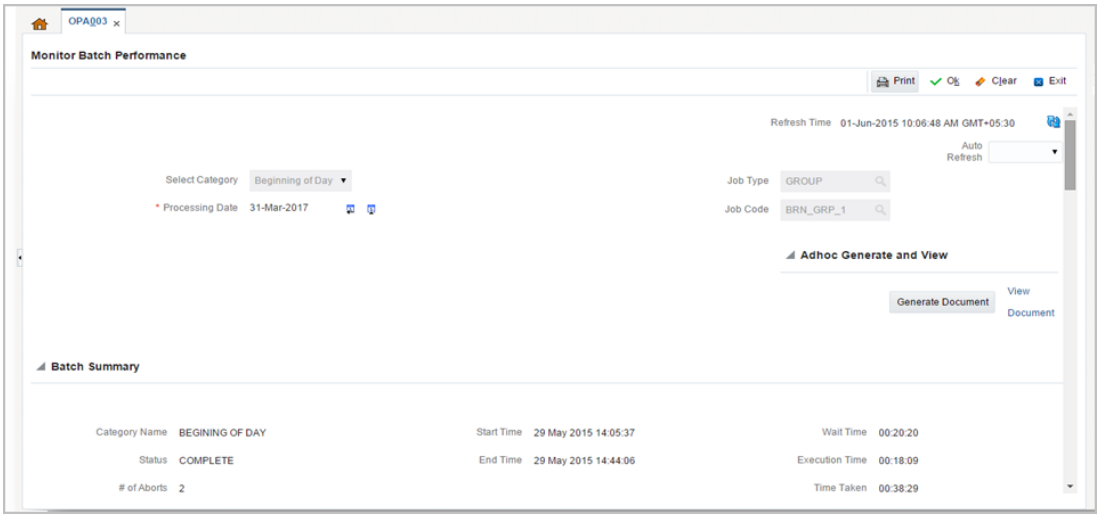
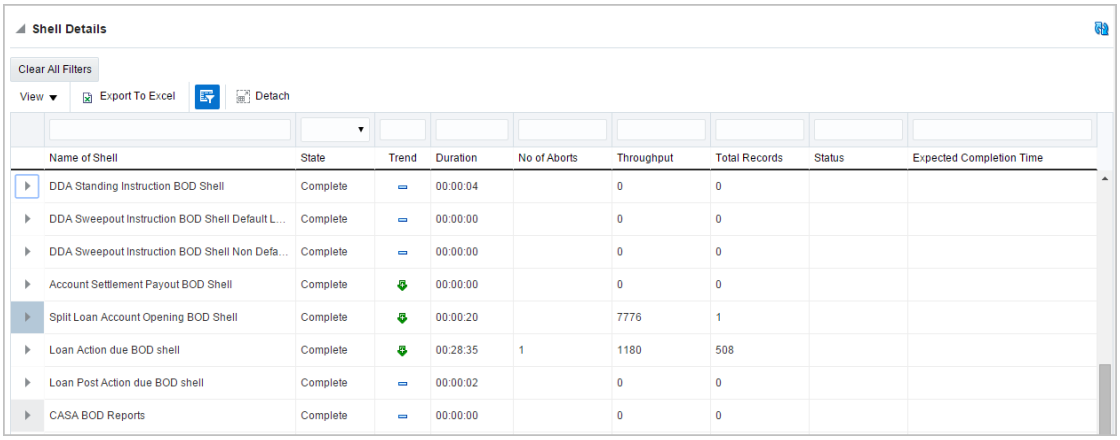
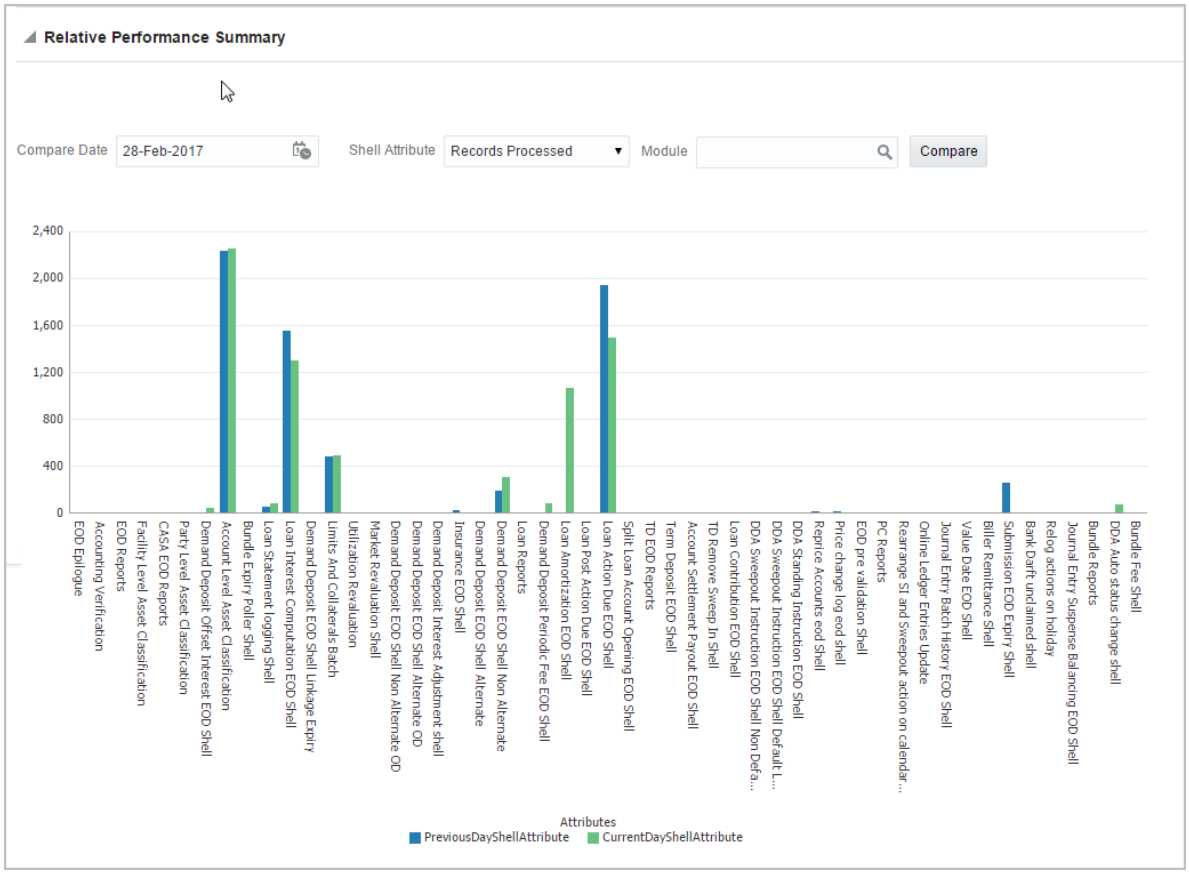


Figure 6–17 Batch Performance Monitoring - Shell Details



Name of Shell	State	Trend	Duration	No of Aborts	Throughput	Total Records	Status	Expected Completion Time
DDA Standing Instruction BOD Shell	Complete		00:00:04		0	0		
DDA Sweepout Instruction BOD Shell Default L...	Complete		00:00:00		0	0		
DDA Sweepout Instruction BOD Shell Non Defa...	Complete		00:00:00		0	0		
Account Settlement Payout BOD Shell	Complete		00:00:00		0	0		
Split Loan Account Opening BOD Shell	Complete		00:00:20		7776	1		
Loan Action due BOD shell	Complete		00:28:35	1	1180	508		
Loan Post Action due BOD shell	Complete		00:00:02		0	0		
CASA BOD Reports	Complete		00:00:00		0	0		

Figure 6–18 Batch Performance Monitoring - Relative Performance Summary



The batch statistics are monitored at different levels as follows:

- Category Level
- Shell Level
- Record level

Category Details

The parameter monitored at category level as given as below:

Table 6–2 Category Details

Attribute Name	Description
Category Name	Name of category for example, EOD (End Of Day)
Status	Indicates status of selected category. Valid Values: COMPLETE, RUNNING, ABORTED.
Start Time	Indicates start time of category. The time is represented in DD-MM-YYYY hh:mm:ss format.
End Time	Indicates end time of category. The time is represented in DD-MM-YYYY hh:mm:ss format.

Attribute Name	Description
Wait Time	It is the time for which category is in Aborted state. Wait time for category denotes the time for which batch was halted.
Execution Time	It the time for which category is in Running state.
Number Of Aborts	Indicates number of times category was aborted.
Total Time	It is the total time taken by category to complete. Time taken for category is summation of wait and execution time.
Estimated Completion Time	It is the predicted time for category completion. This time is calculated based on number of incomplete and pending shells for the current running category. The averaged value of previous run duration is considered for calculating estimated time. Similar calculation is done for estimating completion time for shell. This attribute is displayed only during batch run. It is not displayed once batch is complete
Time Status	The status of category (that is, delayed or early) denotes whether category is running slow or fast. This value is calculated based on average of historical data. This attribute is displayed only during batch run. It is not displayed once batch is complete

Shell Details

This level displays statistics of all shells corresponding to selected category. The parameters monitored at shell level are given below:

Table 6–3 Shell Details

Attribute Name	Description
Name of shell	Represents name of shell
Trend	Valid Values: UPWARD, DOWNWARD, NEUTRAL. It denotes the trend based on historical data for time required by shells to complete.
Status	Valid Values: Complete, Running, Aborted, Not Started. Indicates status of shell
Duration	It is the time required by shell to complete.
Start Time	Indicates start time of shell. The time is represented in DD-MM-YYYY hh:mm:ss format.
End Time	Indicates the time at which shell is completed. The time is represented in DD-MM-YYYY hh:mm:ss format.
Wait Time	It is the time for which shell is in aborted state.
Expected Completion Time	Indicates the estimated time for a shell to complete.
Failed Records	Number of failed records for a shell
Records Processed	Number of records processed in a shell
Number of Streams	Number of streams denote number of processes running in parallel for a shell. On proper analysis of historical data of stream count, number of records and duration for particular shell one can optimize throughput for it.

Attribute Name	Description
Throughput	It is the average processing time for one record. Throughput is denoted in millisecs.

Figure 6–19 Shell Details

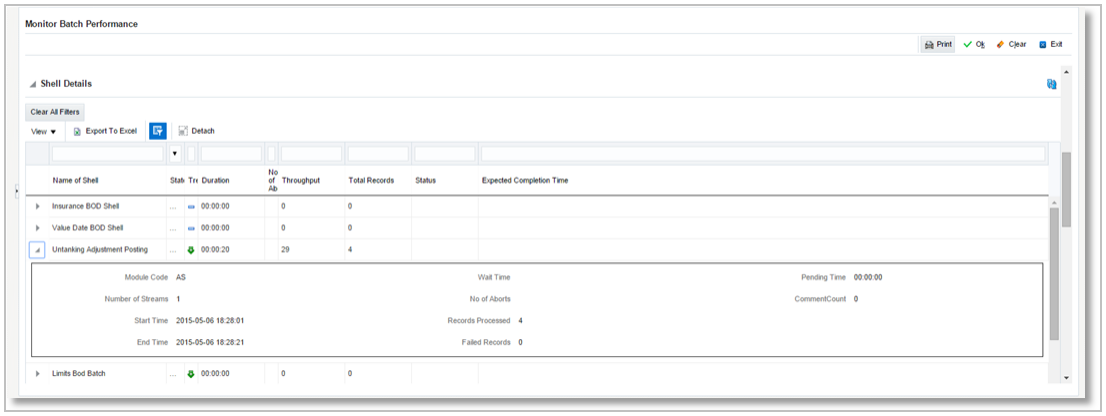
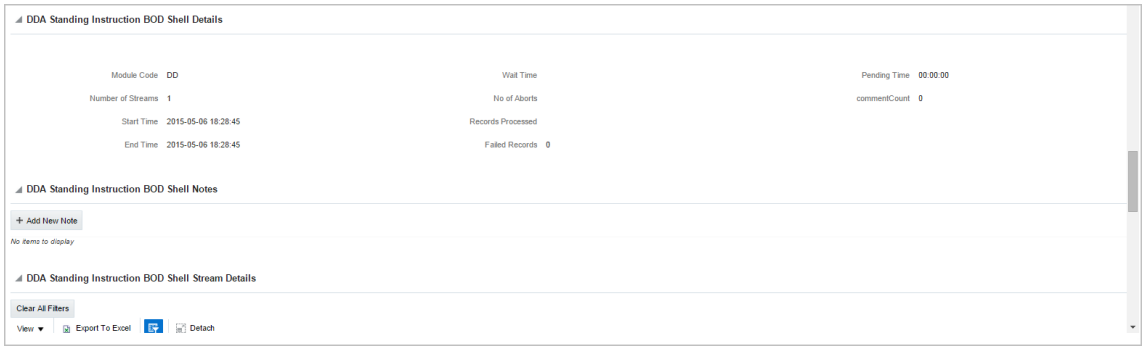


Figure 6–20 Shell Details - DDA Standing Instructions



Note

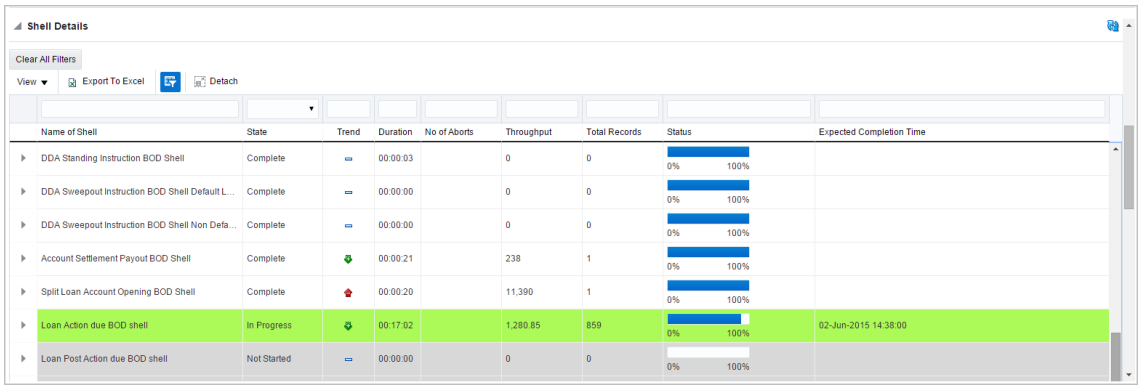
Note the following:

- Trend for a particular shell is decided based on comparison of time statistics (that is, current run time and historical data for previous batch runs). Number of previous batch run to be considered is configurable. It is configured in the property file that is, (BatchStatistics.properties). The trend and other estimated time seems more realistic if number of previous run days configured in property file are more.

- Trend gives an idea whether a particular shell is running fast or slow compared to previous runs though it is important to consider number of records being processed in that shell.

The following figure shows the view displayed during batch run. Few extra parameters like estimated completion time for shell and category are monitored during batch run.

Figure 6–21 View of Batch Run



Comments Table

Comments Table is rendered based on row click of shell details table.

- In case of batch abort, it is important to know the reason behind abort and how that is fixed. Comments table serves this purpose as one can log the details regarding fix and reason behind shell abort. Multiple comments can be captured for particular shell.
- Also one can query historical data for comments. The historical data of comments can be used to analyse the reason behind failure of particular shell.

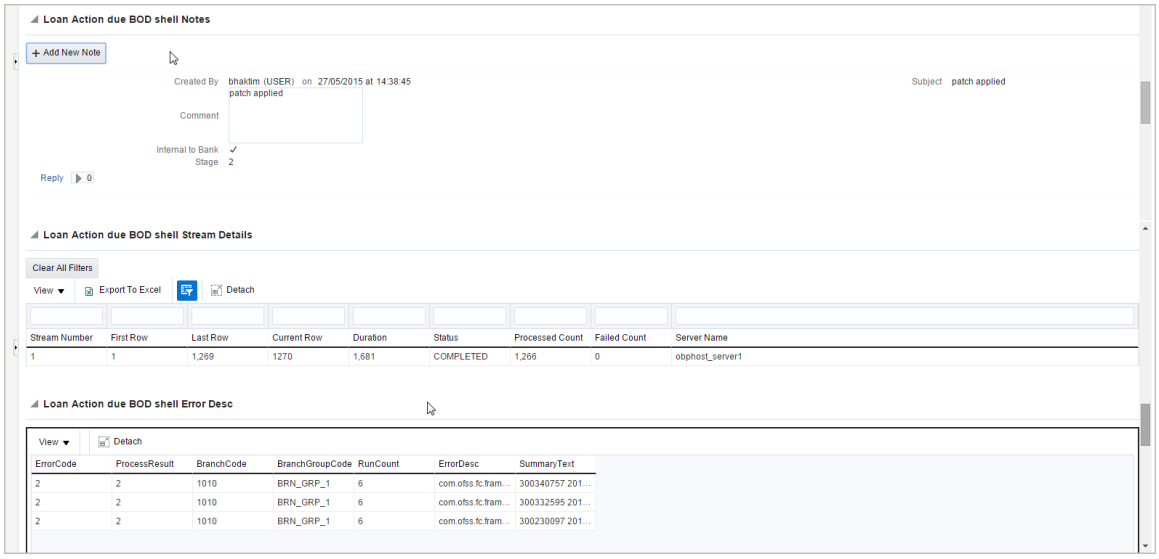
Stream Details Table

Stream Details table is rendered based on row click of shell details table.

Table 6–4 Stream Details

Attribute Name	Description
Stream Number	Indicates the number of a stream in which the record is being processed
First Row	Indicates the start sequence number of a record, processing in a particular stream.
Last Row	Indicates the end sequence number of a record, processing in a particular stream.
Duration	It is the time required for stream to complete.
Status	Valid Values: COMPLETED, RUNNING. It indicates the status of selected stream
Processed Count	Number of records processed in a stream
Server Name	Name of a server running the stream

Figure 6–22 Stream Based Shells



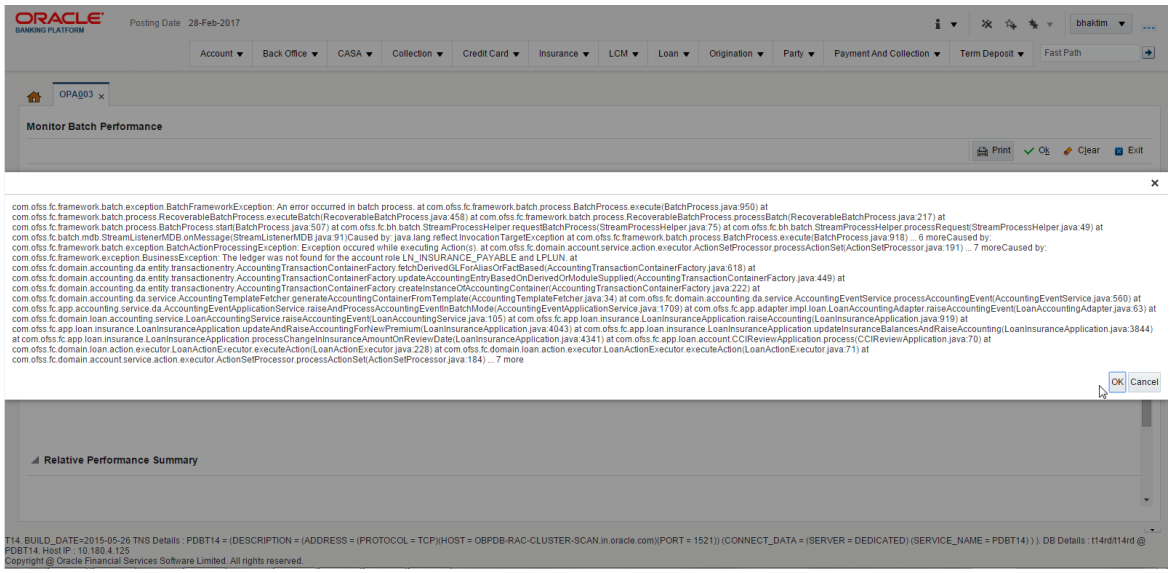
Note

Shells are categorized into two types that is, Stream based shells and Report based shells. Figure 6–22 displays the view for stream based shells.

Exception Log

On row click of the driver level details, it pops up a window showing the stack trace of failed records if present. One can analyze and know the reason behind the failure of that particular record.

Figure 6–23 Exception Log



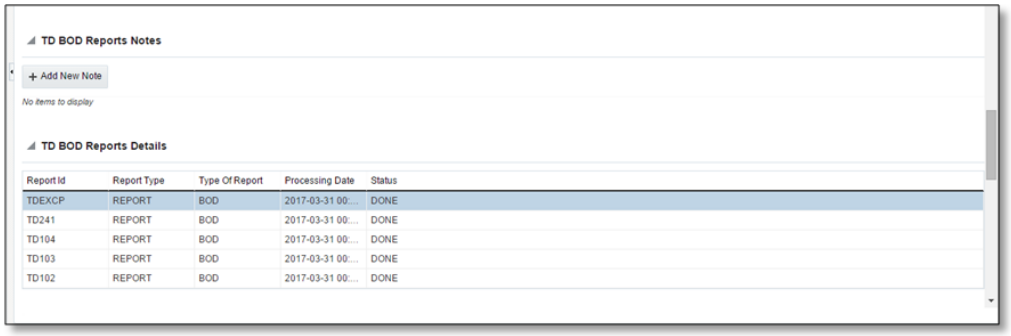
Reports Table

For Report based shells different parameters related to report processing are monitored. The monitored parameters are given below:

Table 6–5 Reports Table

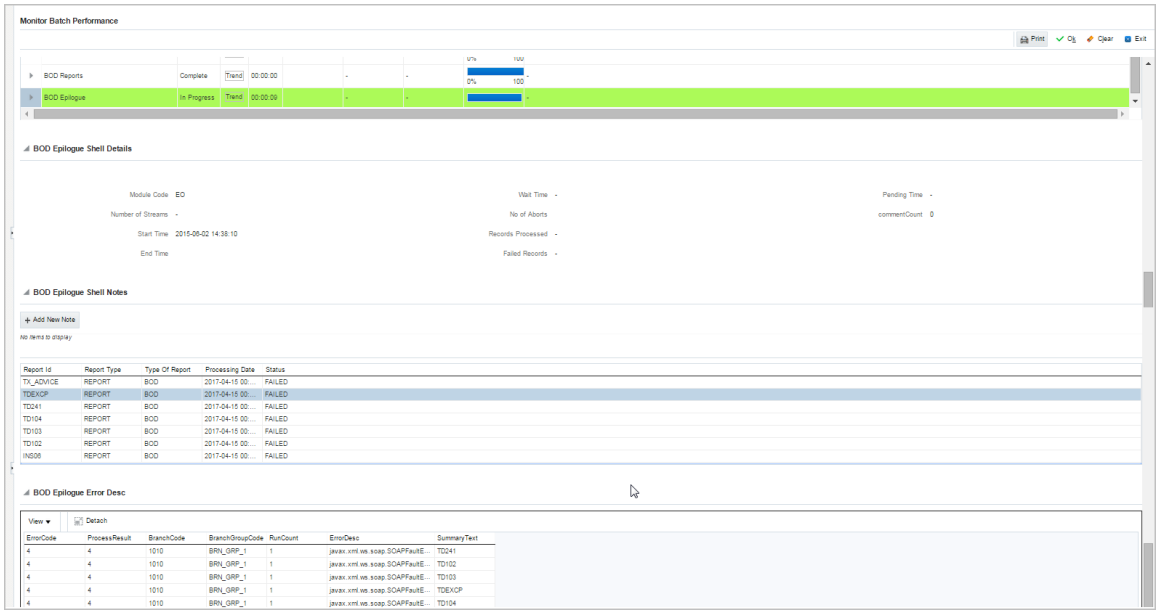
Attribute Name	Description
Report Id	ID to uniquely identify report
Report Type	Report
Type Of Report	Indicates type of reports. Reports are classified based on category.
Processing Date	Indicates processing date of report.
Status	Indicates the status of the report. Valid Values: DONE, PENDING, RUNNING, ABORTED.
Error Message	Error message represents the reason for report failure. No message is displayed in case of successful run.

Figure 6–24 Report Based Shells



The status of report based shell during batch run is shown in Figure 6–25:

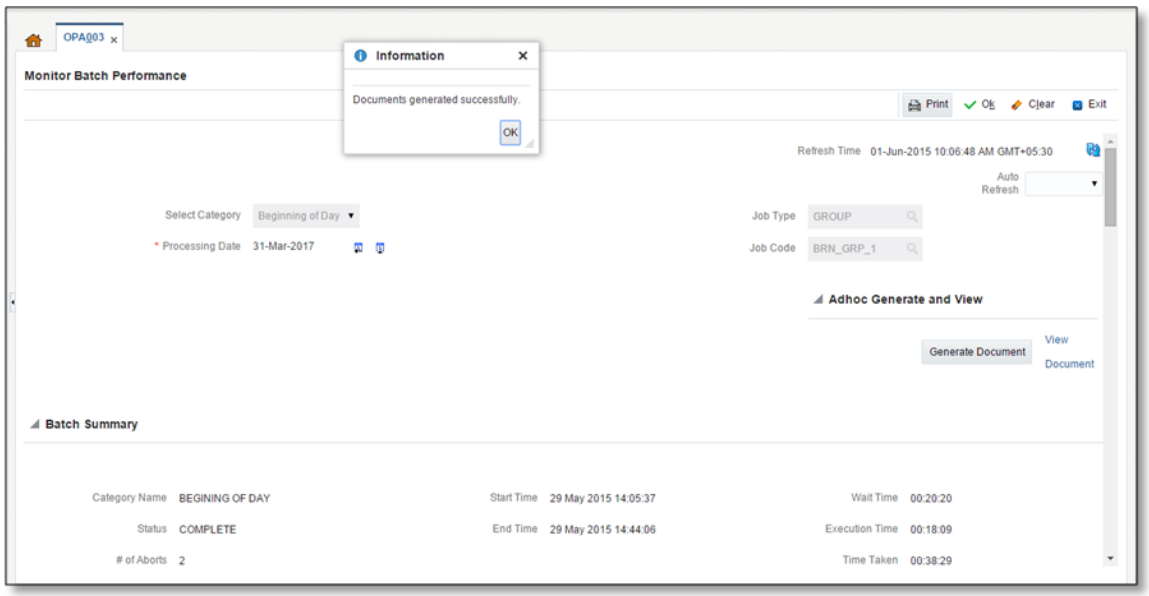
Figure 6–25 Status of Report Based Shell



Exception Report

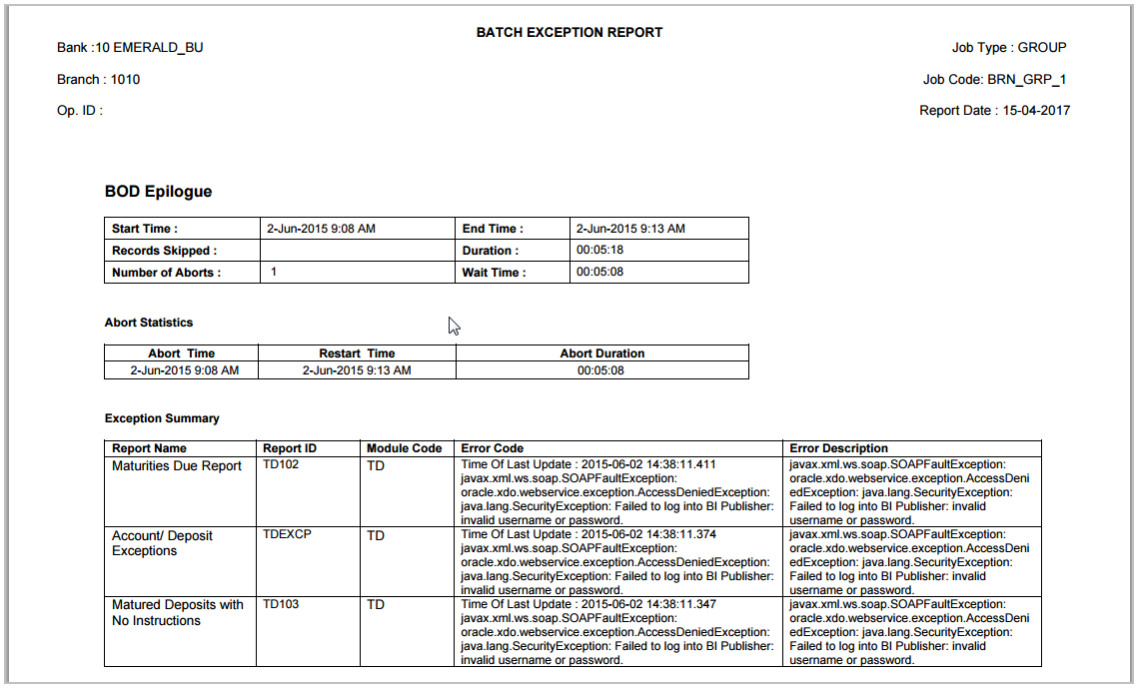
On click of Generate Document, it generates a report for aborted shells with information like Abort count and exception log.

Figure 6–26 Exception Report



The different parameter monitored at shell level and exception logs for all aborted shells are part of exception report. Figure 6–27 displays sample report for a particular shell.

Figure 6–27 Sample Report



BATCH EXCEPTION REPORT

Bank :10 EMERALD_BU

Branch : 1010

Op. ID :

Job Type : GROUP

Job Code: BRN_GRP_1

Report Date : 15-04-2017

BOD Epilogue

Start Time :	2-Jun-2015 9:08 AM	**End Time :**	2-Jun-2015 9:13 AM
Records Skipped :		**Duration :**	00:05:18
Number of Aborts :	1	**Wait Time :**	00:05:08

Abort Statistics

Abort Time	Restart Time	Abort Duration
2-Jun-2015 9:08 AM	2-Jun-2015 9:13 AM	00:05:08

Exception Summary

Report Name	Report ID	Module Code	Error Code	Error Description
Maturities Due Report	TD102	TD	Time Of Last Update : 2015-06-02 14:38:11.411 javax.xml.ws.soap.SOAPFaultException: oracle.xdo.webservice.exception.AccessDeniedException: java.lang.SecurityException: Failed to log into BI Publisher: invalid username or password.	javax.xml.ws.soap.SOAPFaultException: oracle.xdo.webservice.exception.AccessDeniedException: java.lang.SecurityException: Failed to log into BI Publisher: invalid username or password.
Account/ Deposit Exceptions	TDEXCP	TD	Time Of Last Update : 2015-06-02 14:38:11.374 javax.xml.ws.soap.SOAPFaultException: oracle.xdo.webservice.exception.AccessDeniedException: java.lang.SecurityException: Failed to log into BI Publisher: invalid username or password.	javax.xml.ws.soap.SOAPFaultException: oracle.xdo.webservice.exception.AccessDeniedException: java.lang.SecurityException: Failed to log into BI Publisher: invalid username or password.
Matured Deposits with No Instructions	TD103	TD	Time Of Last Update : 2015-06-02 14:38:11.347 javax.xml.ws.soap.SOAPFaultException: oracle.xdo.webservice.exception.AccessDeniedException: java.lang.SecurityException: Failed to log into BI Publisher: invalid username or password.	javax.xml.ws.soap.SOAPFaultException: oracle.xdo.webservice.exception.AccessDeniedException: java.lang.SecurityException: Failed to log into BI Publisher: invalid username or password.

Exception Log Table

The figure below provides the details of the exception log.

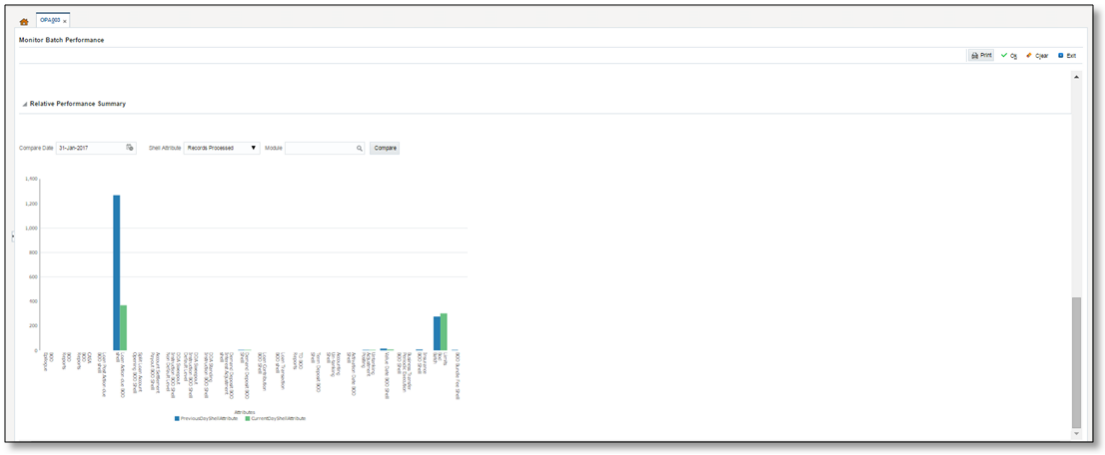
Figure 6–28 Exception Log Table

ERROR CODE	ERROR DESCRIPTION	SUMMARY
kException: An error occurred in batch process.862-8203-0444-6360		
Time Of Last Update : 2013-09-11 11:28:51.438 Error reference Number :862-8182-2552-7227Error msg :An error occurred in batch process.Error cause :com.ofss.fc.framework.batch.exception.BatchFrameworkException: An error occurred in batch process.862-8182-2552-7227	com.ofss.fc.framework.batch.exception.BatchFrameworkException: An error occurred in batch process. at com.ofss.fc.framework.batch.process.BatchProcess.execute(BatchProcess.java:910) at com.ofss.fc.framework.batch.process.RecoverableBatchProcess.executeBatch(RecoverableBatchProcess.java:432) at com.ofss.fc.framework.batch.process.RecoverableBatchProcess.processBatch(RecoverableBatchProcess.java:	
Time Of Last Update : 2013-09-11 11:28:53.868 Error reference Number :862-8206-5767-2044Error msg :An error occurred in batch process.Error cause :com.ofss.fc.framework.batch.exception.BatchFrameworkException: An error occurred in batch process.862-8206-5767-2044	com.ofss.fc.framework.batch.exception.BatchFrameworkException: An error occurred in batch process. at com.ofss.fc.framework.batch.process.BatchProcess.execute(BatchProcess.java:910) at com.ofss.fc.framework.batch.process.RecoverableBatchProcess.executeBatch(RecoverableBatchProcess.java:432) at com.ofss.fc.framework.batch.process.RecoverableBatchProcess.processBatch(RecoverableBatchProcess.java:	

6.2.3 Histogram of Shell Attribute Comparison

This section provides graphical view for comparing shell attributes for any two selected dates. The processing date, shell attribute and module name are taken as input to this table. The output is displayed as bar graph in which X axis represents the name of the shell and Y axis denotes value of shell attribute. For each shell two records are displayed, these records corresponds to the dates for which the data is being compared.

Figure 6–29 Graphs

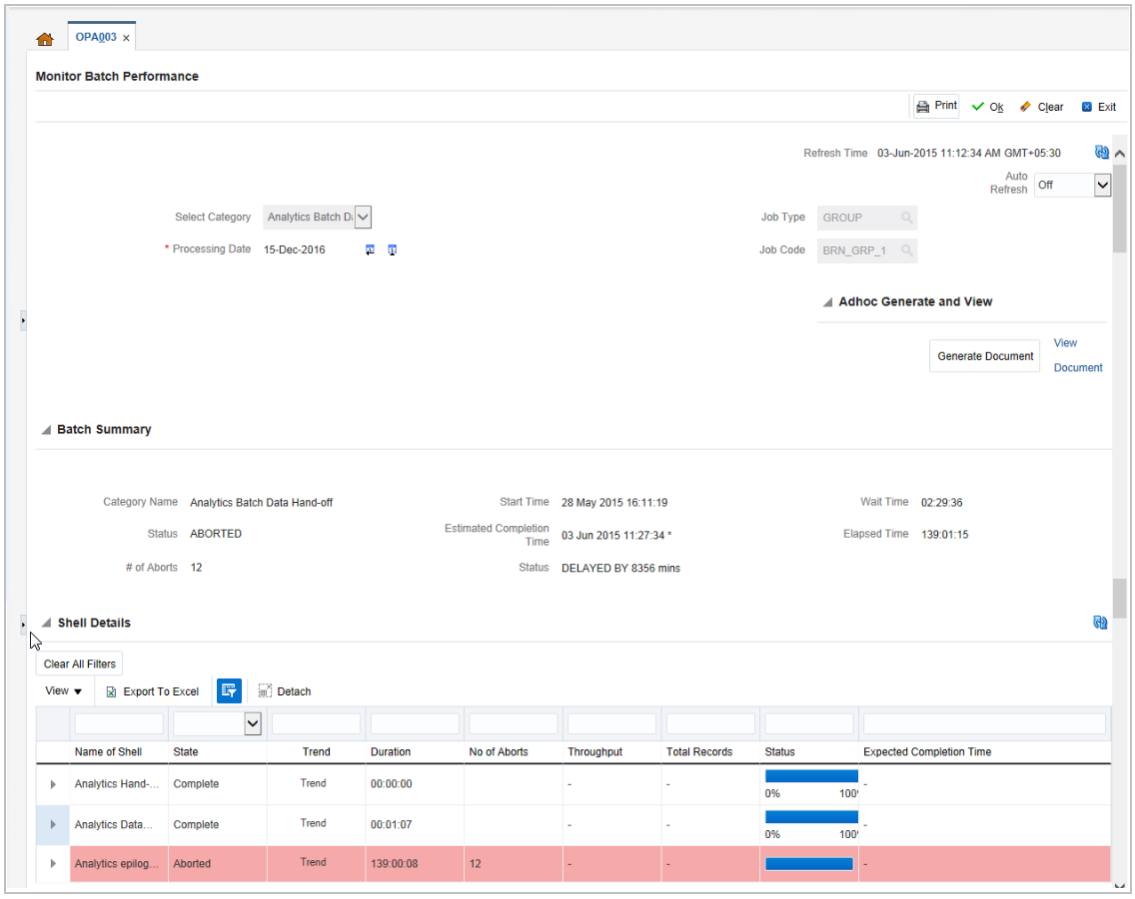


6.3 ODI Batch Handoff Monitoring

This section provides, top package level execution details for monitoring of ODI handoff. The input for these details are category ID, branch group code, branch group type, and processing date.

Figure 6–30 describes the input parameters for the batch handoff monitoring and the shell details like name of shell, start time, end time, number of aborts.

Figure 6–30 Input Parameters for Batch Handoff



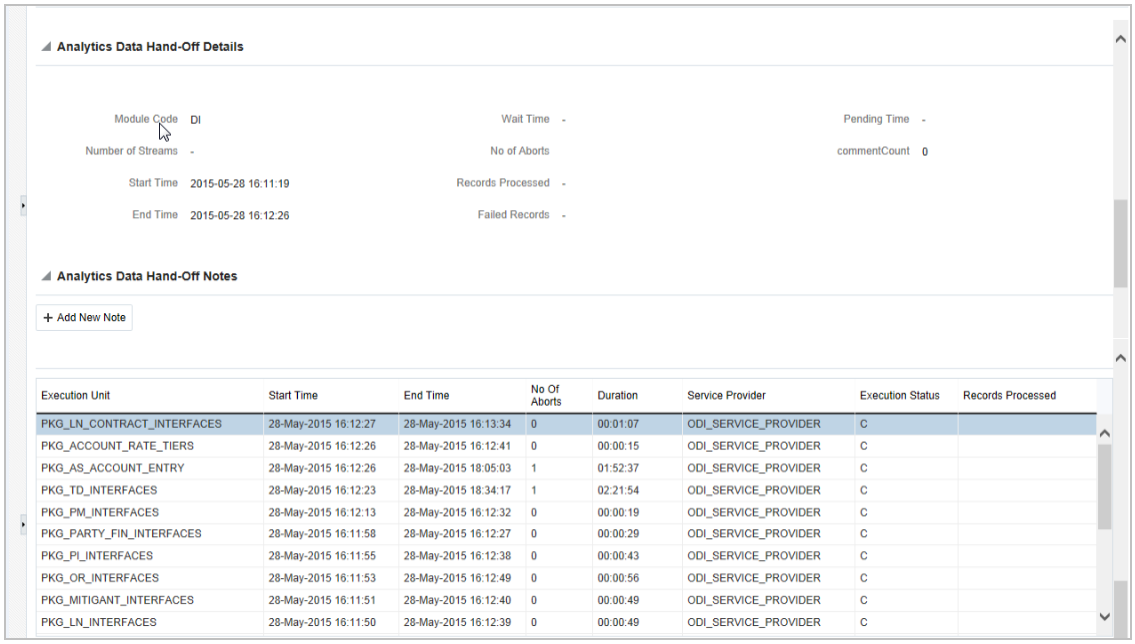
Execution Unit

On click of the analytics data Handoff shell, the below table is shown with the execution unit (top level package) level details:

This table contains the following attributes:

- Execution unit name
- Start time of execution of the execution unit
- End time of execution of the execution unit
- Number of aborts of the execution unit
- Duration of execution of the execution unit
- Service provider for ETL process (ODI)
- Execution status of the execution unit, that is, complete, running, and aborted
- Records processed at the stage level in ETL process

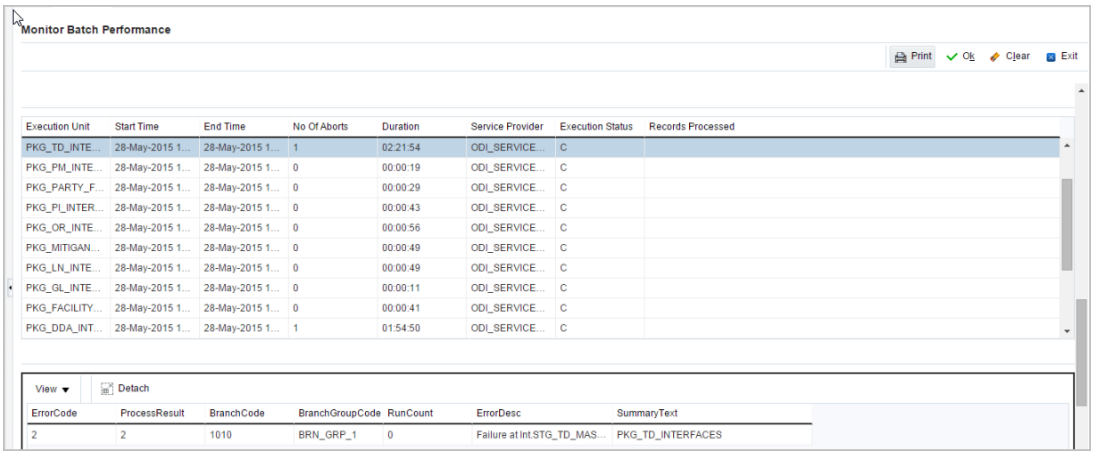
Figure 6–31 Execution Unit



Abort Statistics

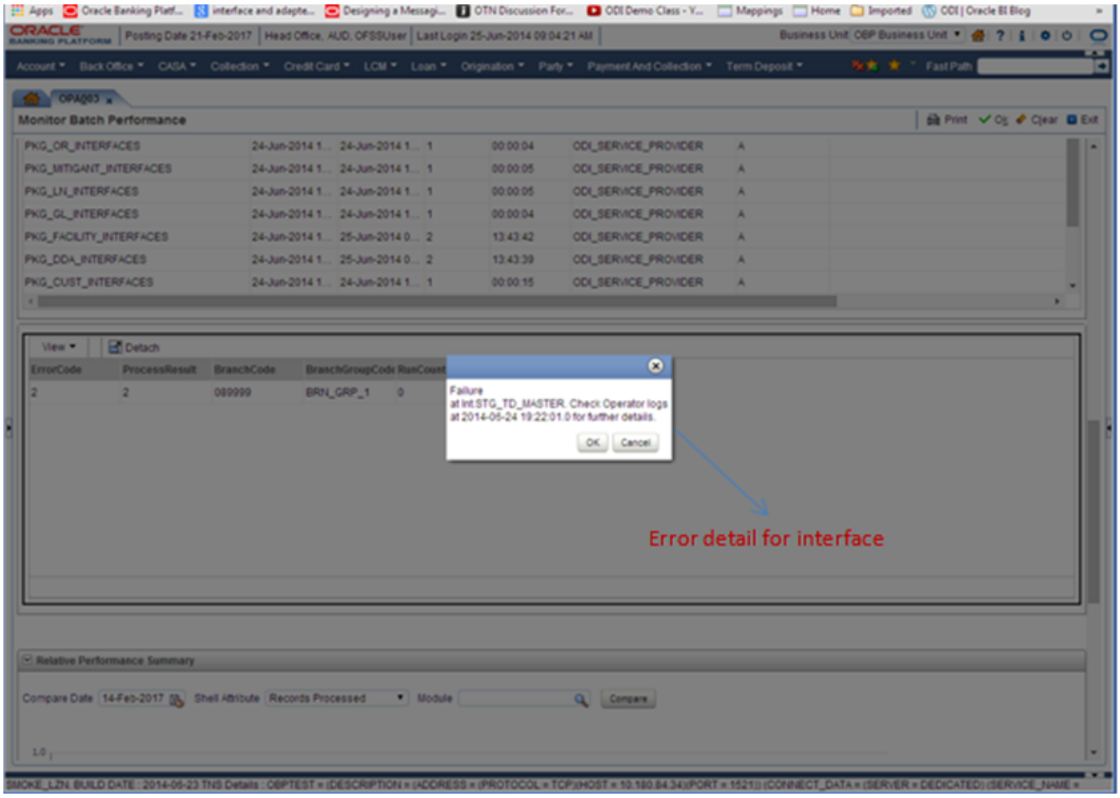
On click of aborted execution unit, the below table is shown with the abort details like run count, the actual error description, and summary of the exception containing the interface name for which the exception occurred.

Figure 6–32 Abort Statistics



On click of the error description table, the error description message appears as shown in Figure 6–33:

Figure 6–33 Failure Error Description



7 Application Monitoring Using EM Plugin

This chapter provides an overview on the various monitoring operations performed as an administrator, using Enterprise Manger (EM) Plugin.

7.1 Monitoring Application Using EM Plugin

Once DMS statistics are captured for a particular channel and transactions involving in it, it requires a UI representation to understand the stats in a readable form so that one can analyze the behavior. The monitoring activities are mainly carried out by IT Technical staff.

7.1.1 Oracle Enterprise Manager (EM)

Oracle Enterprise Manager is the application where all the monitoring data exists. It includes server and machines status and performance and also OBP monitoring statistics.

All the servers are monitored by EM including Host, UI, SOA, and so on.

We have a view corresponding to every environment containing all the components which include outbound components.

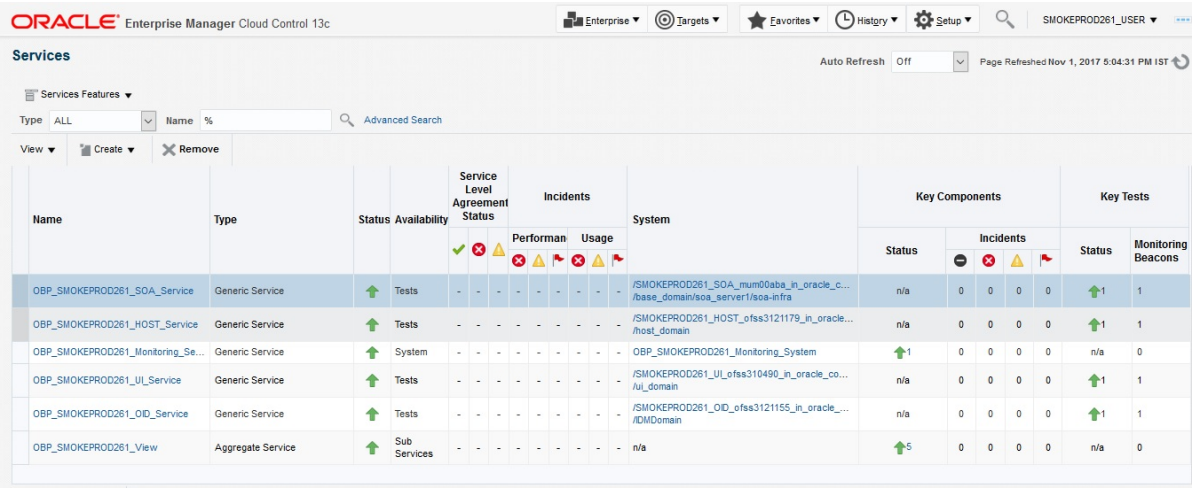
Some notations in EM are provided below:

Table 7–1 Notations in EM

	Indicates component is down
	Indicates component is up and running
	Indicates alerts
	Indicates warnings
	Indicates metric collection error
	Indicates healthy status

The following figure shows the environment view in Oracle Enterprise Manager:

Figure 7–1 Oracle Enterprise Manager



The views in the above figure include UI, Host, and SOA servers.

Security Stacks components such as OAAM, OID, OES, outbound components such as BIP, IPM, Documaker, ATM and POS channels are also part of the environment view.

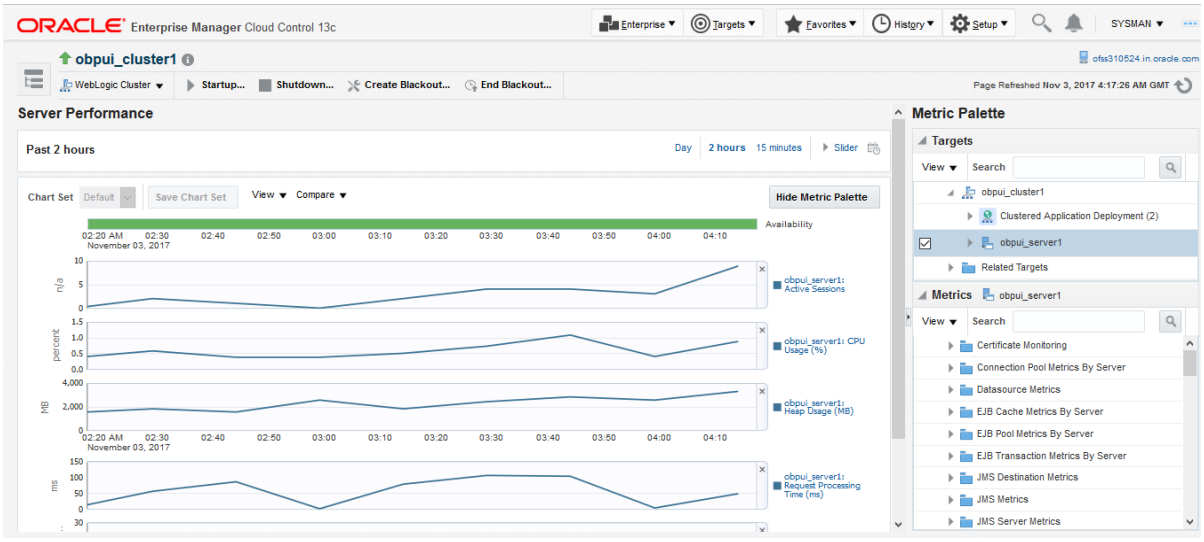
Each component can be further explored for details by clicking on the links provided for them.

7.1.2 UI

For UI, all the managed servers created under Weblogic cluster can be monitored. EM provides the following information for UI Cluster:

- Active Session about all Managed Servers
- CPU Usage
- Heap Usage
- Request Processing Time

Figure 7–2 UI Cluster in EM

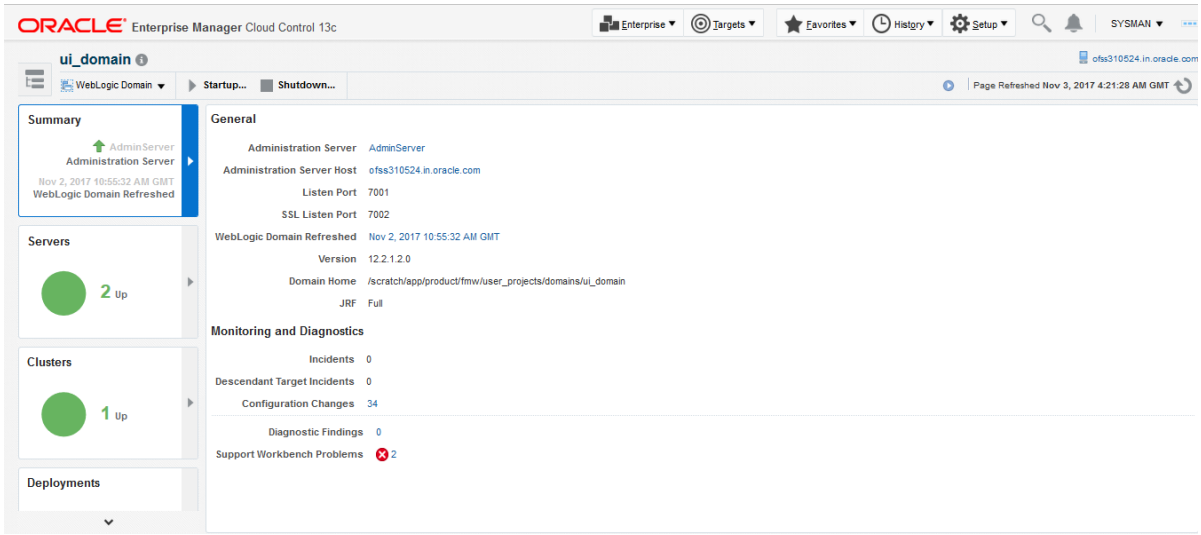


UI is hosted on WebLogic domain, so the EM target of UI machine is WebLogic domain. EM gives the following information for UI:

- Server Performance Statistics
- Up/Down Status
- List of deployed applications
- Incidents or Alerts; if any

The following figure displays the WebLogic domain for UI.

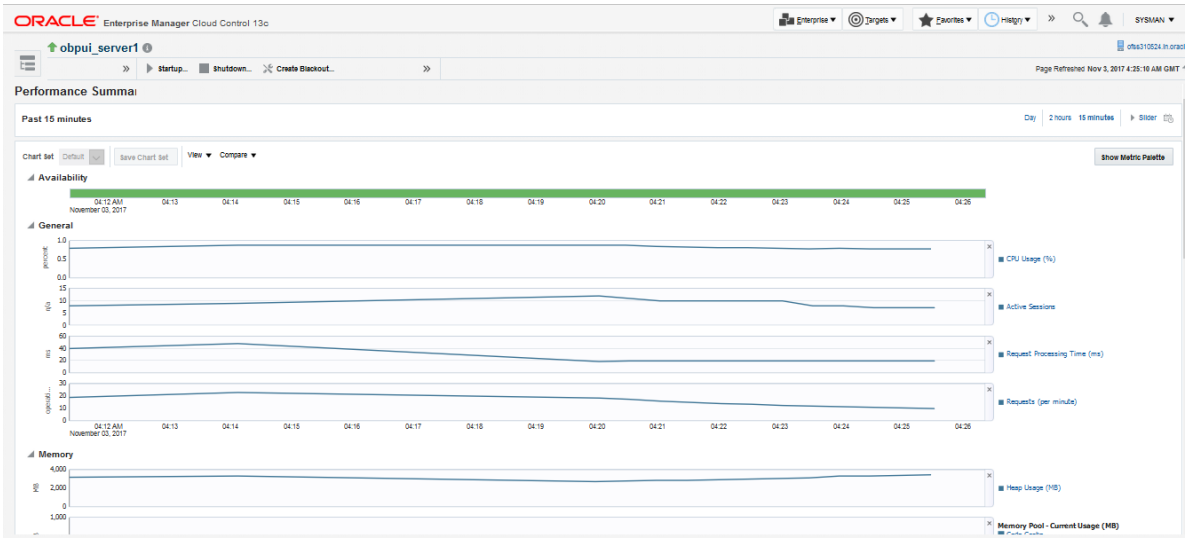
Figure 7–3 WebLogic Domain for UI



The performance metric includes metrics like CPU Utilization, Memory Utilization, Active Sessions and are default metrics provided by EM.

The following figure displays the metrics chart.

Figure 7–4 Metrics Chart

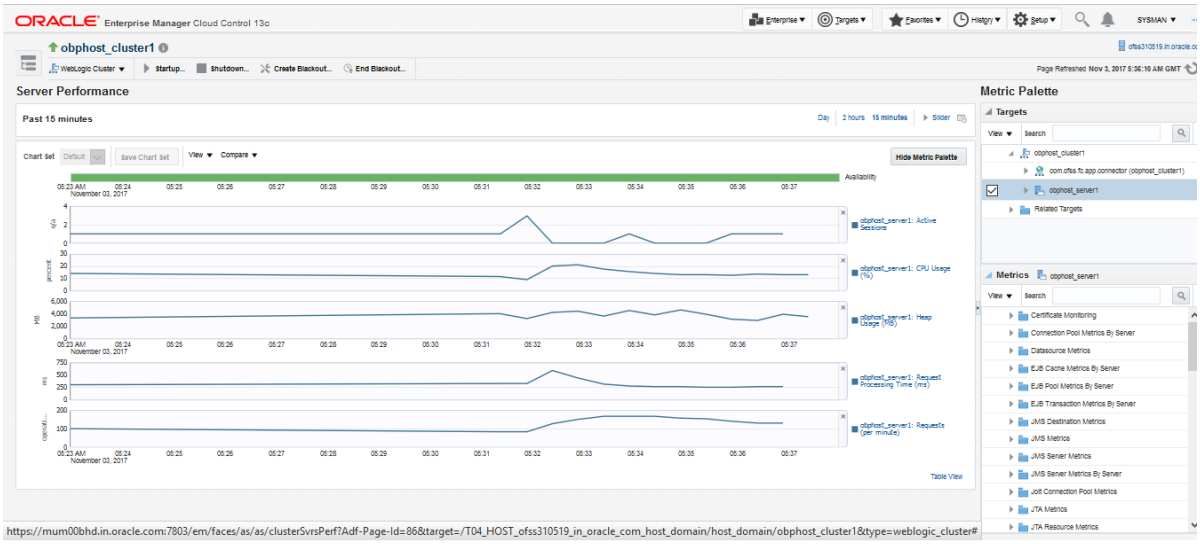


7.1.3 Host

For Host, all the managed servers created under Weblogic cluster can be monitored. EM provides the following information for Host Cluster:

- Active Session about all Managed Servers
- CPU Usage
- Heap Usage
- Request Processing Time

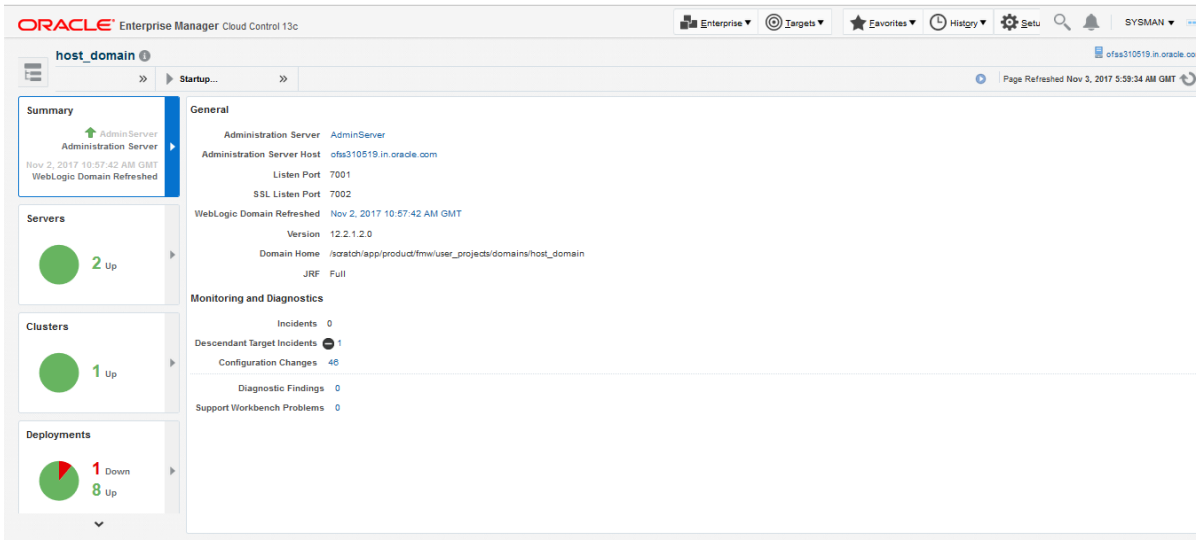
Figure 7–5 Host Cluster in EM



Similar to UI, Host is also deployed on WebLogic domain and has similiar metrics like UI.

The following figure displays the host target in EM.

Figure 7–6 Host Target in EM



The following figure displays the metric charts.

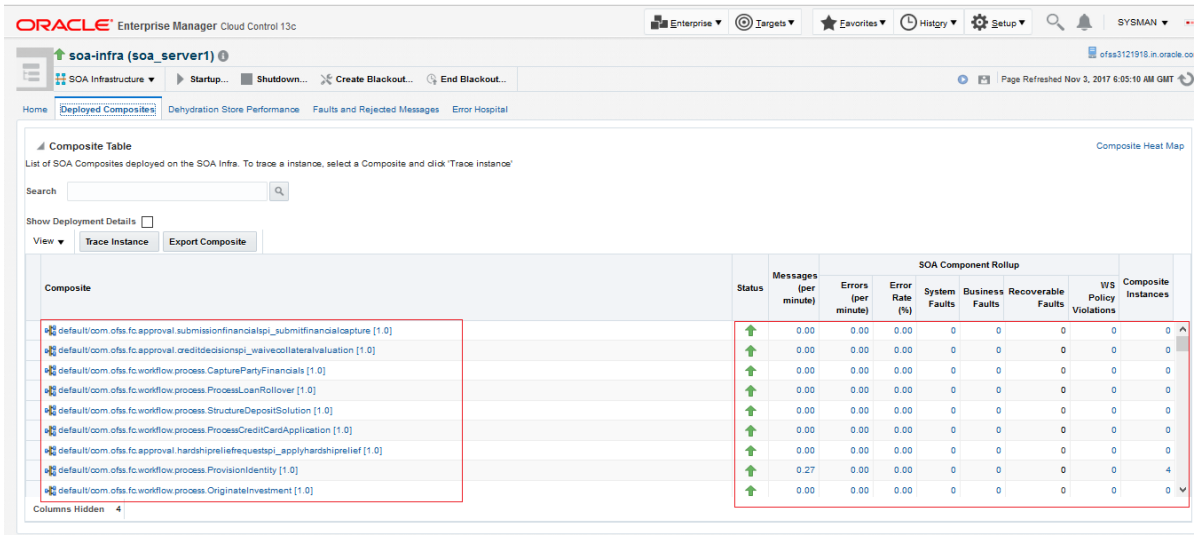
Figure 7–7 Metrics Chart

7.1.4 SOA

SOA server is deployed on WebLogic domain where the SOA processes are deployed.

The process list can be seen in the list of deployed applications. The other metrics remain same as for WebLogic domain in EM. The following figure displays the process list.

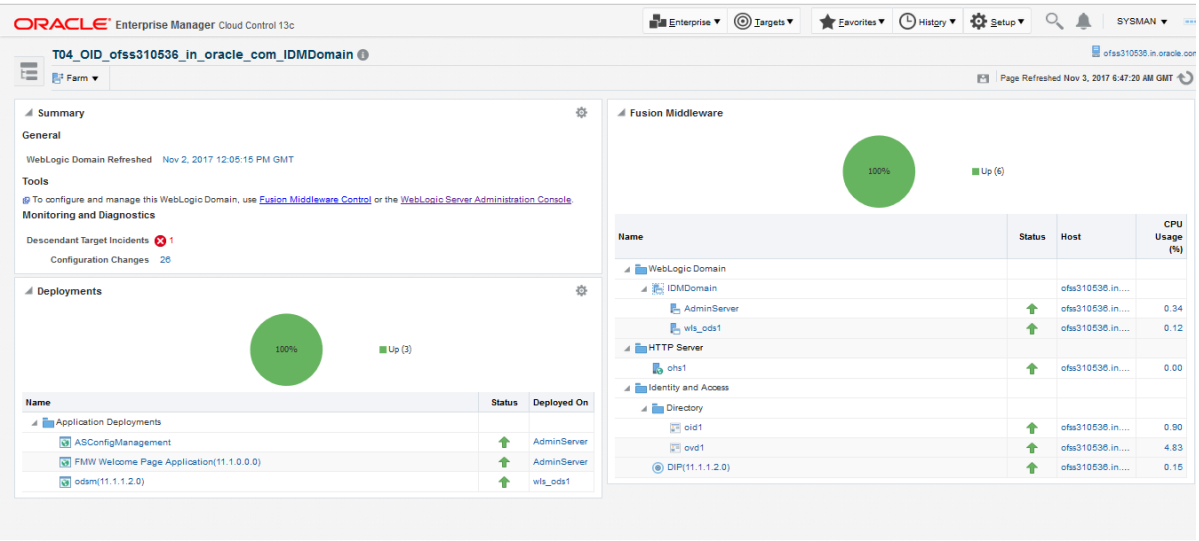
Figure 7–8 Viewing Process List



7.1.5 Security Stack (OID and OAM)

OID and OAM are also deployed as WebLogic domain.

Figure 7–9 OID WebLogic Domain



7.1.6 Document Generation Outbound Components (Documaker, BIP, IPM)

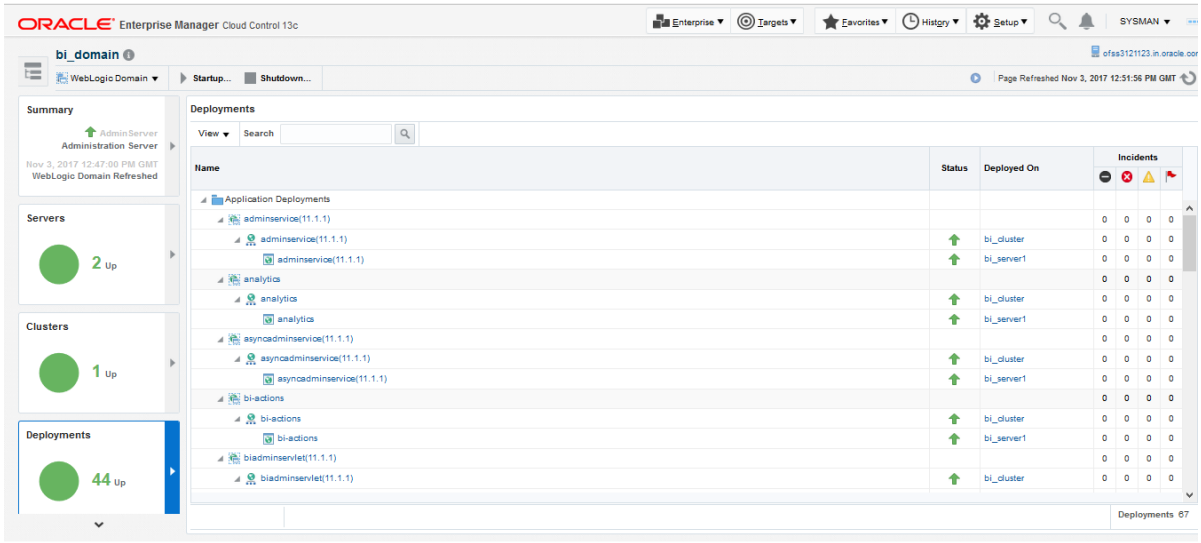
These are not part of the application, but we monitor these so as to detect the cause of failure in case the document generation fails at any point of time.

A webservice is invoked for generating the documents from the application with BIP as well as documaker. From EM, we check whether that webservice is up and running or not, which gives the status of these components. The following figure displays the status from EM.

Figure 7–10 Document Generation Status

BIP is also deployed on WebLogic domain.

Figure 7–11 BIP Deployment



7.1.7 ATM and POS (Point Of Sales) Channels

ATM and POS work on socket listener mechanism.

So, for them to be up and running the port on which they listen should be up. In EM, to monitor these channels, check if the port is listening.

The following figure displays the status from EM.

Figure 7–12 EM Monitoring

7.1.8 Outbound OFSAA call

The application calls OFSAA for calculation of economic cost. This is done through a webservice.

To monitor this, check if the webservice is up and running.

Figure 7–13 Web Monitoring

7.1.9 Monitoring Views

Monitoring views show the batch and application performance statistics along with server performance history. It consists of Batch Monitoring and Application Monitoring tabs, which show detailed view of batch performance and application performance statistics along with the server performance statistics on which they are running.

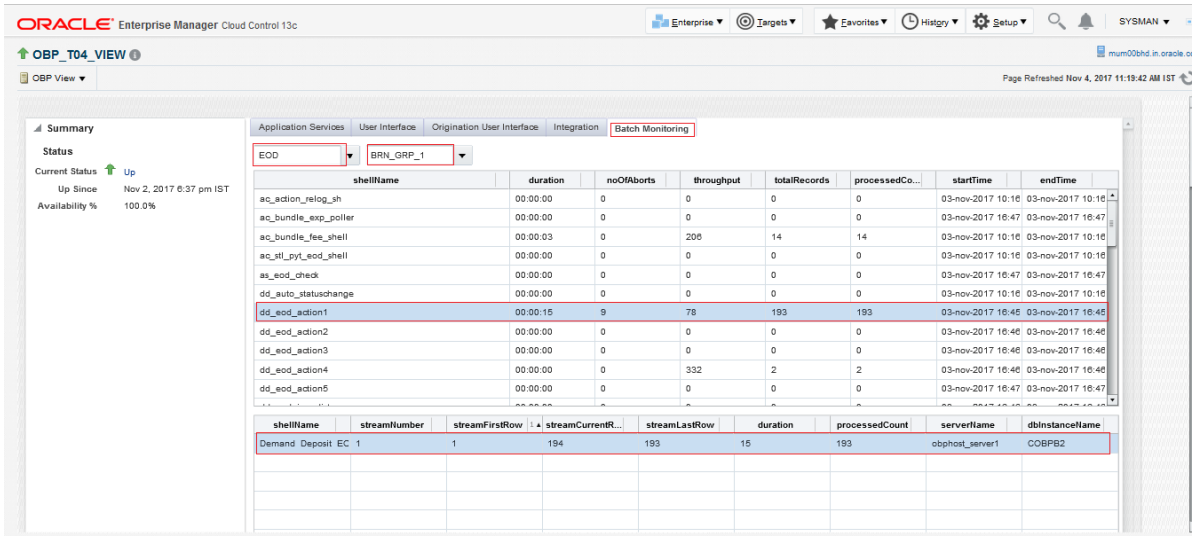
7.1.9.1 Batch Monitoring

Batch Monitoring shows detailed view for host and database server performance charts along with batch performance statistics.

The batch performance statistics are the details of the categories run in the application. The date for which category details are shown is the last run date. The categories include EOD, CutOff, Internal System EOD and BOD.

To get the details of a particular category, select it from the combo box. This will display the list of shells in the category in the table below. From the table, select the desired shell, the shell details provides the stream details of the selected shell.

Figure 7–14 Database Server Info



The streams can run in different servers. To get the details of the performance of the server in which the stream is executed, select the stream. The charts below gives the performance summary of the server in which the stream is executed and the database performance.

The following figure displays the status from EM.

Figure 7–15 Batch Monitoring Status

Figure 7–16 Batch Configuration

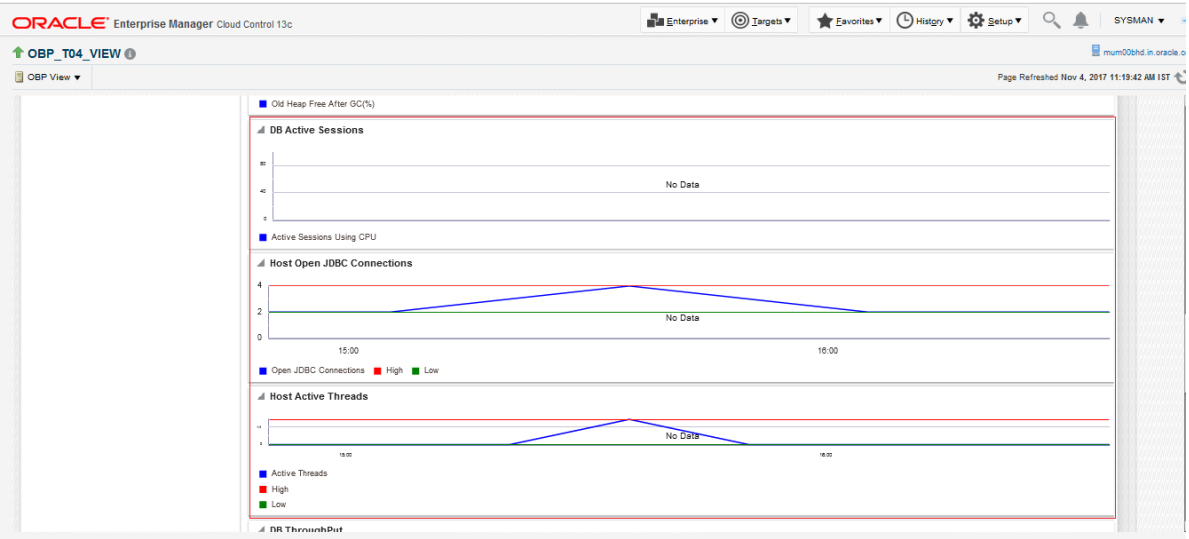
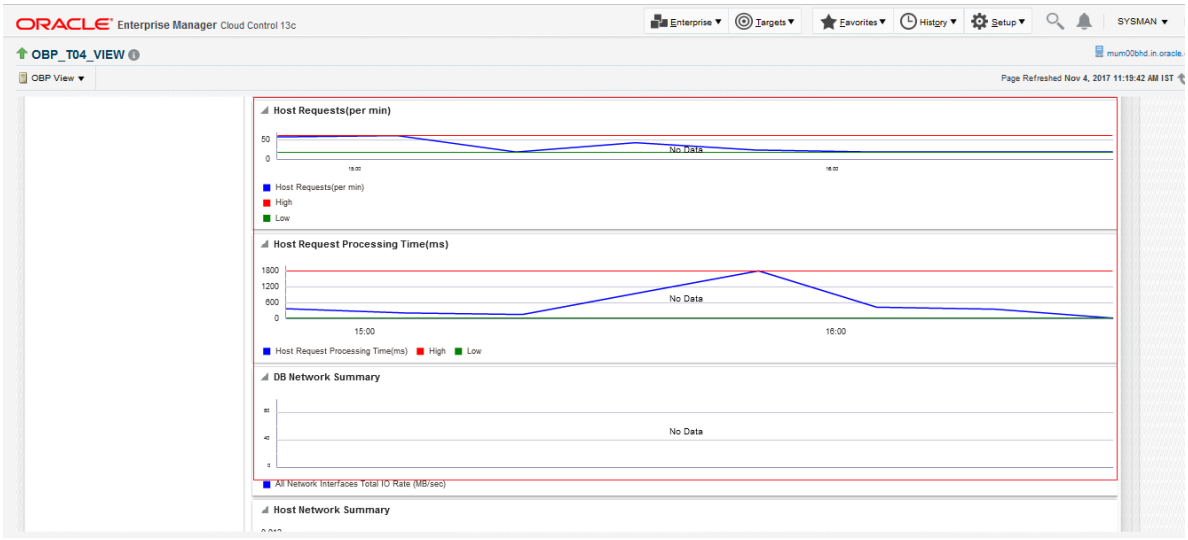


Figure 7–17 WebLogic Service Info



7.1.9.2 Application Monitoring

Application Monitoring shows detailed view of UI and host clusters and servers.

There are four separate tabs, namely Application Services, User Interface, Origination User Interface, and Integration.

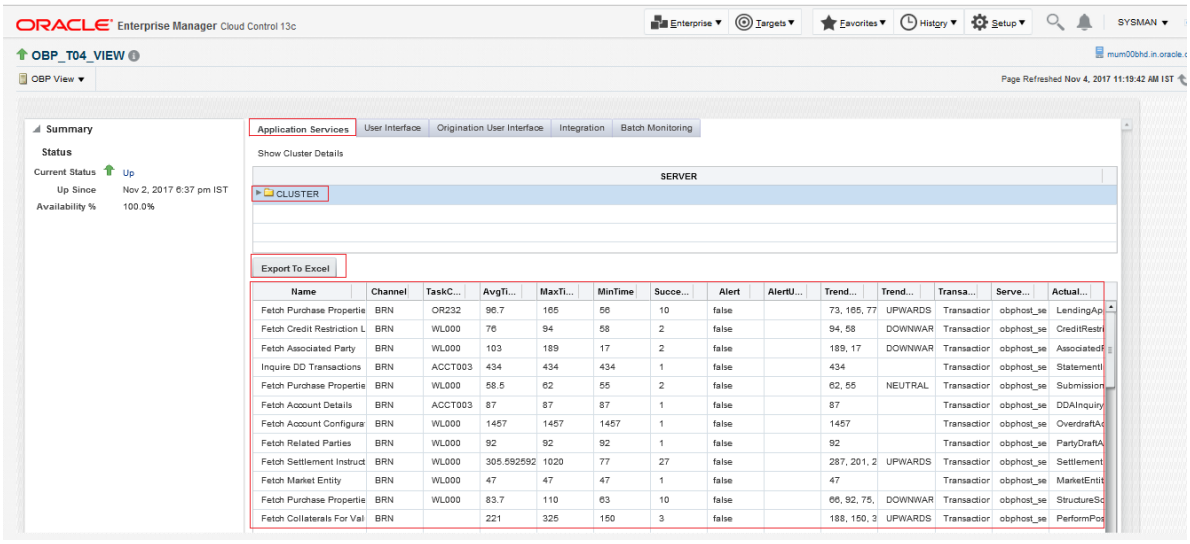
7.1.9.2.1 Application Services

This section provides performance metrics for all application services executed on Host Server. Metrics include timing, alert, trending information, and so on.

For cluster details, click the Show Cluster Details link.

Click CLUSTER to view application metrics for the servers present in the HOST cluster. User can export the application metric data by clicking the Export To Excel button.

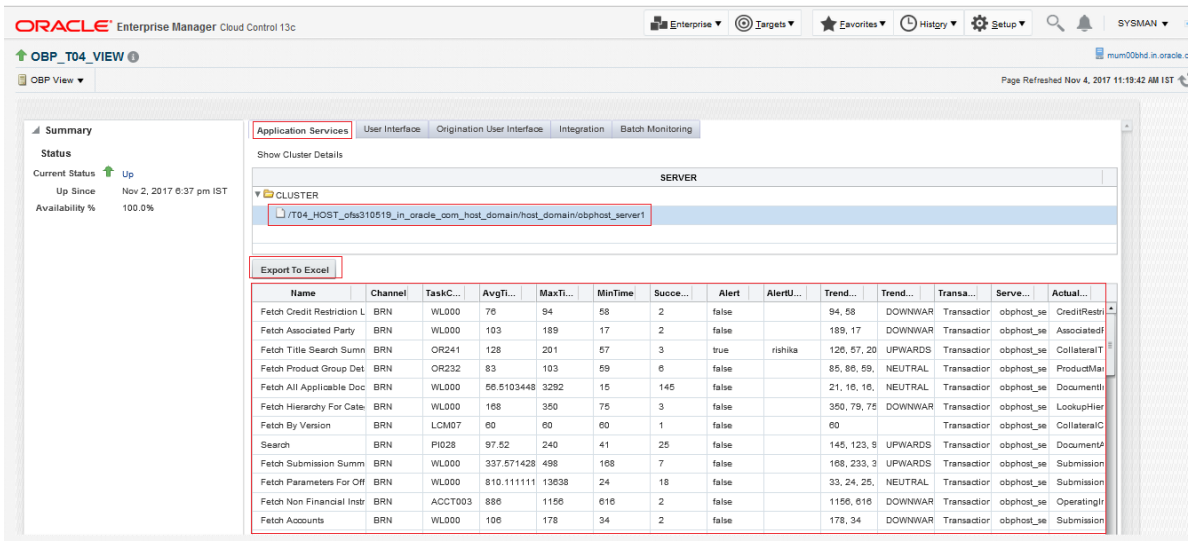
Figure 7–18 Application Metrics of Application Services for all servers in cluster



Expand CLUSTER to get a list of all the servers.

Each server can be further selected to get the details at the server level. On clicking the server, the application metrics are displayed in the table as shown in the following figure.

Figure 7–19 Application Metrics of Application Services for selected server



The following table explains each column of the table present in the given snapshot:

Table 7–2 Details of the Application Metrics table of Application Services

Sr. No.	Column Name	Description
1	Name	Logical name of the application services

Sr. No.	Column Name	Description
2	Channel	Channel through which the transaction occurred Valid Values: Branch, ATM, and POS.
3	Task Code	Task code of the application page by which the transaction was triggered. Application module of which transaction is a part
4	Average Time	Average execution time of the application service
5	Max Time	Maximum time of execution of the application service
6	Min Time	Minimum time of execution of the application service
7	Success Count	Number of times application service executed successfully
8	Alert	Alert state of the application service
9	Alert User	Teller who performed the last alerted transaction
10	Trend Reference Queue	Execution time of last n transactions (n=5)
11	Trend	Trending of transaction Valid Values: Upwards, Downwards, Neutral
12	Transaction Type	Maximum time of execution of the transaction
13	Server Name	Server name
14	Actual Service Name	Service name of the transaction

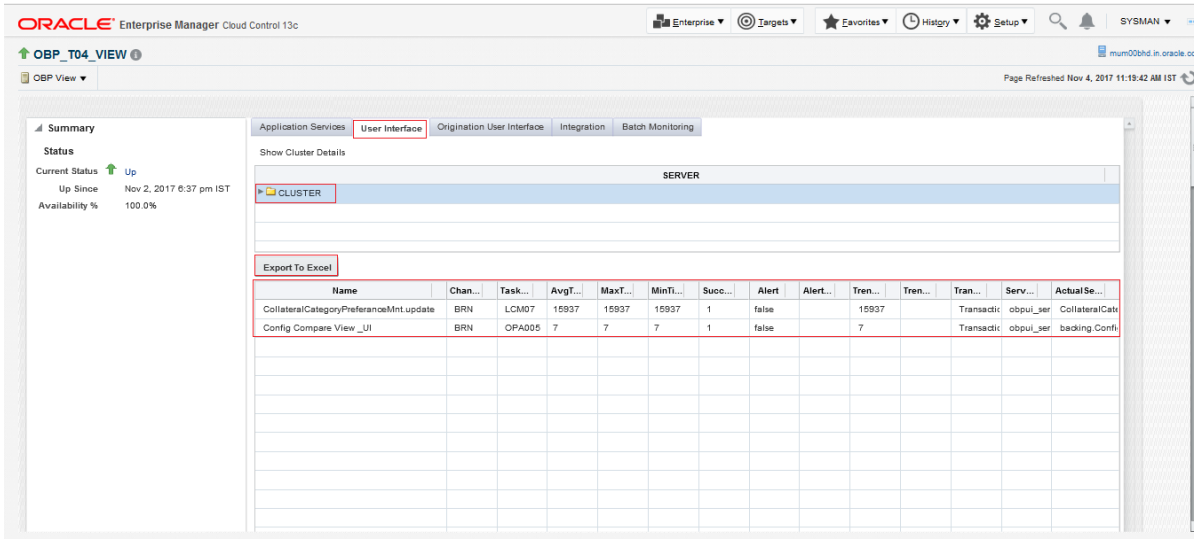
7.1.9.2.2 User Interface

This section provides performance metrics for all major UI components executed on UI Server. Metrics include timing, alert, trending information, and so on.

For cluster details, click the Show Cluster Details link.

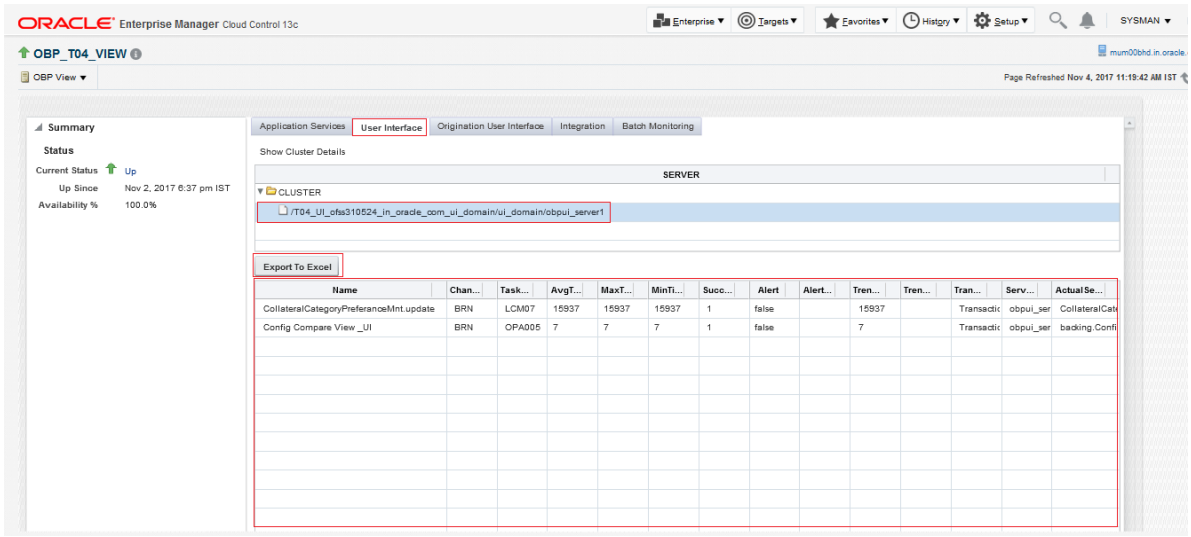
Click CLUSTER to view application metrics for the servers present in the UI cluster. User can export the application metric data by clicking the Export To Excel button.

Figure 7–20 Application Metric for all UI servers in cluster



Expand CLUSTER to get a list of all the servers. Each server can be further selected to get the details at the server level. On clicking the server, the application metrics are displayed in the table as shown in the following figure.

Figure 7–21 Application Metrics of UI components for selected server



The following table explains each column of the table present in the given snapshot:

Table 7–3 Details of the Application Metrics table of UI Components

Sr. No.	Column Name	Description
1	Name	Logical name of the UI component
2	Channel	Channel through which the transaction occurred

Sr. No.	Column Name	Description
		Valid Values: Branch, ATM, and POS
3	Task Code	Task code of the application page by which the transaction was triggered. Application module of which transaction is a part
4	Average Time	Average execution time of UI component
5	Max Time	Maximum time of execution of the UI component
6	Min Time	Minimum time of execution of the UI component
7	Success Count	Number of times UI component executed successfully
8	Alert	Alert state of the UI component
9	Alert User	Teller who performed the last alerted transaction
10	Trend Reference Queue	Execution time of last n transactions (n=5)
11	Trend	Trending of transaction Valid Values: Upwards, Downwards, Neutral
12	Transaction Type	Type of transaction
13	Server Name	UI Server name
14	Actual Service Name	Actual name of UI component

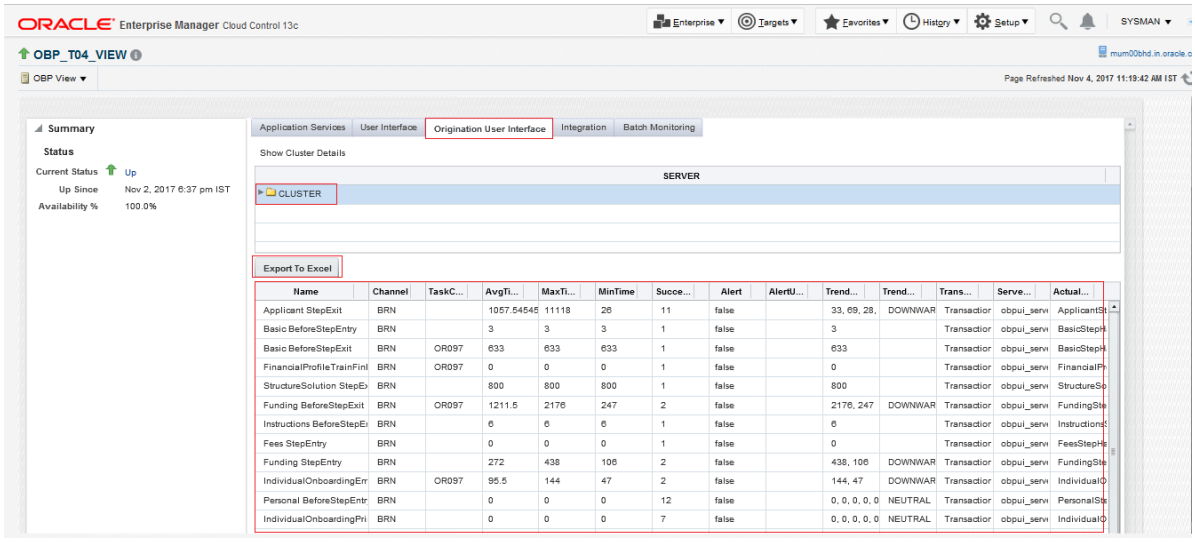
7.1.9.2.3 Origination User Interface

This section provides performance metrics for rendering all multistep train taskflows. The metrics capture the time taken for entering and exiting a particular step of the application form. If there are any host calls made to persist data before leaving a step or to fetch data from host server before entering a step, these metrics encapsulate those timings. Metrics include timing, alert, trending information, and so on.

For cluster details, click the Show Cluster Details link.

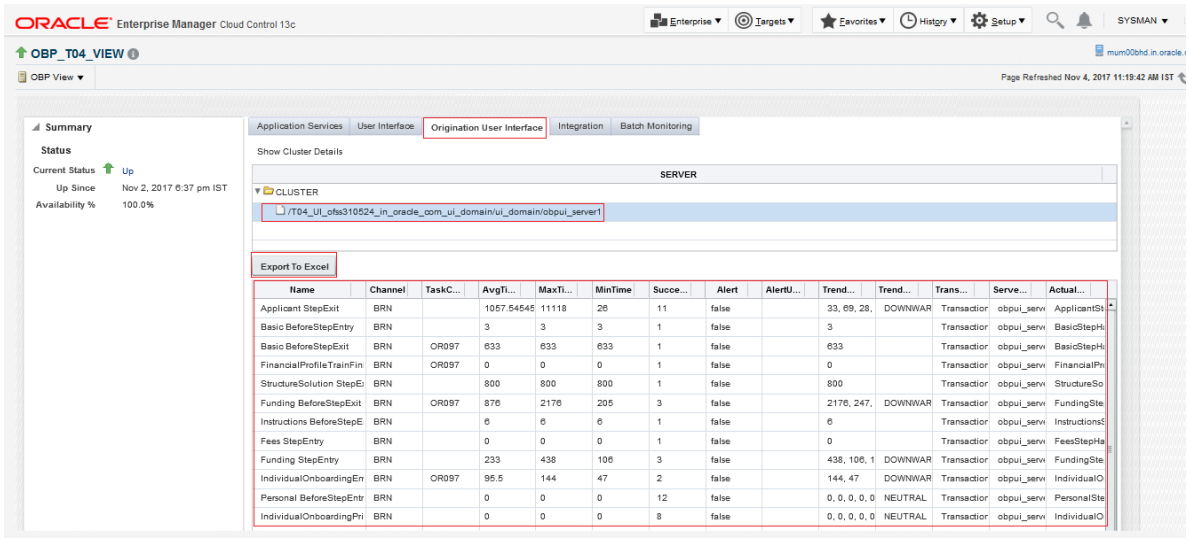
Click CLUSTER to view application metrics for the servers present in the UI cluster. User can export the application metric data by clicking the Export To Excel button.

Figure 7–22 Application Metrics of Origination UI Components for all UI servers in cluster



Expand 'CLUSTER' to get a list of all the servers. Each server can be further selected to get the details at the server level. On clicking the server, the application metrics are displayed in the table as shown in the following figure.

Figure 7–23 Application Metrics of Origination UI components for selected server



The following table explains each column of the table present in the given snapshot:

Table 7–4 Details of the Application Metrics table of Origination UI Components

Sr. No.	Column Name	Description
1	Name	Logical name of the multistep train taskflow component

Sr. No.	Column Name	Description
2	Channel	Channel through which the transaction occurred Valid Values: Branch, ATM, and POS.
3	Task Code	Task code of the application page by which the transaction was triggered. Application module of which transaction is a part
4	Average Time	Average execution time
5	Max Time	Maximum time of execution of the multistep train taskflow component
6	Min Time	Minimum time of execution of the multistep train taskflow component
7	Success Count	Number of times multistep train taskflow component executed successfully
8	Alert	Alert state of the multistep train taskflow component
9	Alert User	Teller who performed the last alerted transaction
10	Trend Reference Queue	Execution time of last n transactions (n=5)
11	Trend	Trending of transaction Valid Values: Upwards, Downwards, Neutral
12	Transaction Type	Type of transaction
13	Server Name	UI Server name
14	Actual Service Name	Actual name of multistep train taskflow component

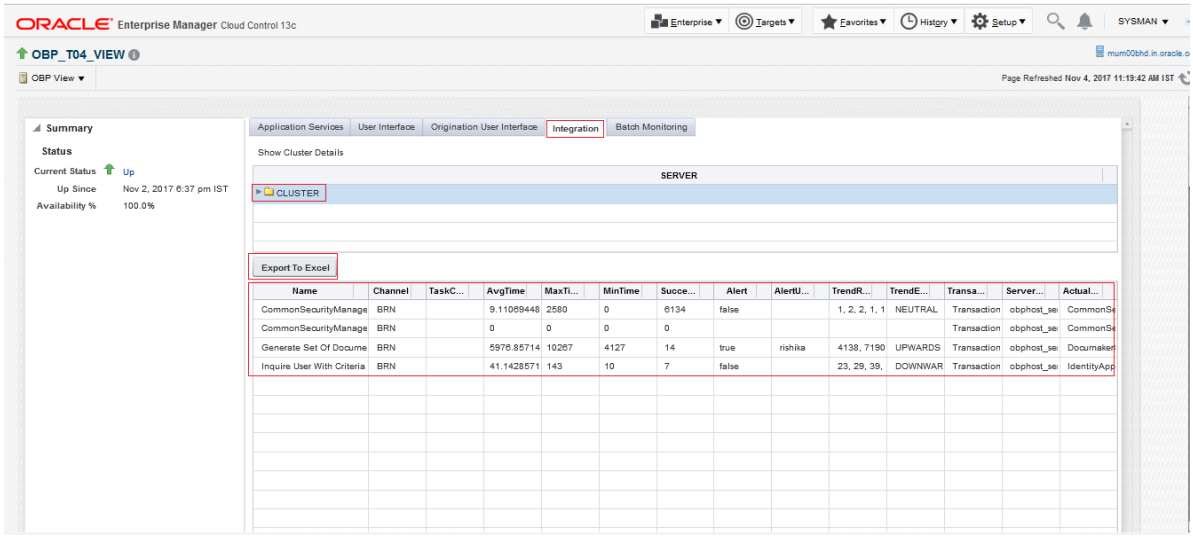
7.1.9.2.4 Integration

This section provides performance metric for all outbound services called from Host Server. Metrics include timing, alert, trending information, and so on.

For cluster details, click the Show Cluster Details link.

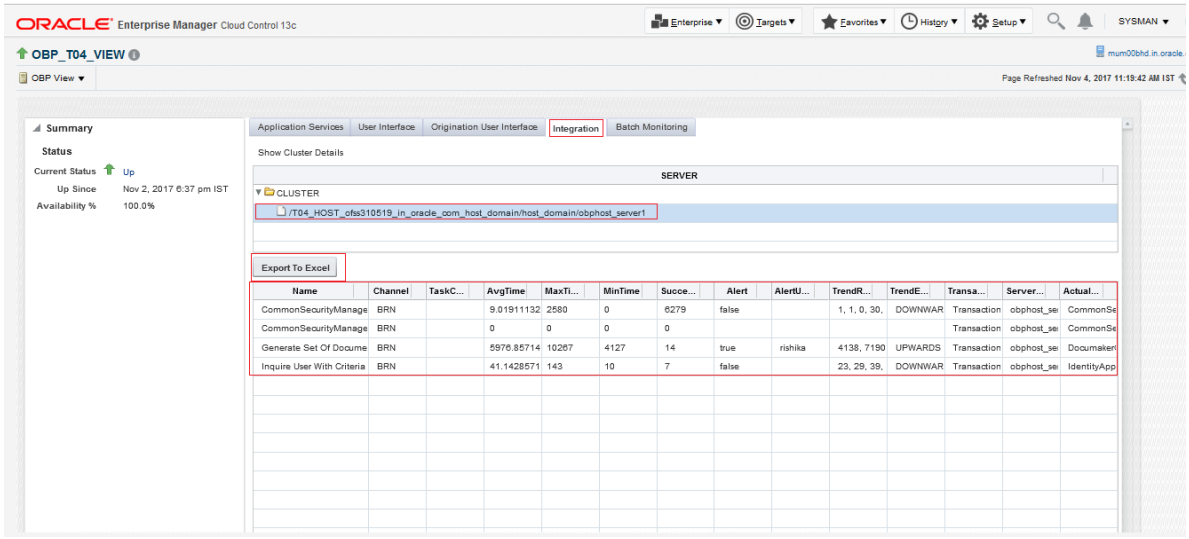
Click CLUSTER to view application metrics for the servers present in the HOST cluster. User can export the application metric data by clicking the Export To Excel button.

Figure 7–24 Application Metrics of all outbound services called from all host servers in cluster



Expand it to get a list of all the servers. Each server can be further selected to get the details at the server level. On clicking the server, the application metrics are displayed in the table as shown in the following figure.

Figure 7–25 Application Metrics of all outbound services called from selected server



The following table explains each column of the table present in the given snapshot:

Table 7–5 Details of the Application Metrics table of all Outbound Services

Sr. No.	Column Name	Description
1	Name	Logical name of the Outbound services
2	Channel	Channel through which the transaction occurred Valid Values: Branch, ATM, and POS.

Sr. No.	Column Name	Description
3	Task Code	Task code of the application page by which the transaction was triggered. Application module of which transaction is a part
4	Average Time	Average execution time
5	Max Time	Maximum time of execution of the outbound service
6	Min Time	Minimum time of execution of the outbound service
7	Success Count	Number of times outbound service executed successfully
8	Alert	Alert state of the outbound service
9	Alert User	Teller who performed the last alerted transaction
10	Trend Reference Queue	Execution time of last n transactions (n=5)
11	Trend	Trending of transaction Valid Values: Upwards, Downwards, Neutral
12	Transaction Type	Maximum time of execution of the transaction
13	Server Name	Server name
14	Actual Service Name	Service name of the transaction

8 Configuration Export-Import Operations

This chapter gives an insight to the Configuration Export-Import operations.

8.1 Objective

Config operations include exporting business configurations, from one environment, to DB or file and importing these configurations in another environment, thus replicating the entire data with the golden copy.

Compare Business Configurations (OPA005) page provides the UI to compare the entities present in two environments on the basis of the data attributes.

Suppose R1 is an environment where the teller has maintained an entity, say currency AUD and in R2 environment the teller wants the exact copy of R1. The Import Export operations allows the user to export a single entity or all entities of a taskcode and can replicate the working environment with the exported version of data very effectively.

The overall Config operations are divided into five parts, each part representing an operation with its specific functionality. The user has the option to invoke any of the operation to get the required work done.

8.2 Export

This operation aims at exporting a business configuration of a taskcode to the configured location. It stores the serializable response of the entity. When export operation is invoked, data gets exported to the database or file as per configuration.

This operation can be carried out as a webservice call for the Export operation of the specific taskcode whose page level configuration has not been done.

8.3 Import

This operation aims at replicating the entity of target environment with exported data from a source environment. It retrieves the serializable response of the entity from database or file as per configuration and de-serializes the response to replicate the entity in target environment. When import operation is performed, it fetches the response from the source environment database and inserts/updates in the target environment.

8.4 Export All

This operation aims at exporting all the entities of a given taskcode. So that the same can be replicated in other environment. It is carried out through a web service call, by invoking the fetchAllAndExport method of ExportImportApplicationService. The request parameters are sessionContext, taskCode.

Export Request

Export request xml is provided below:

```
- <soapenv:Envelope xmlns:soapenv="http://
schemas.xmlsoap.org/soap/envelope/"
xmlns:exp="http://eximp.service.ops.app.fc.ofss.com/ExportImportAp
plicationService" xmlns:con="http:// context.app.fc.ofss.com"
xmlns:exc="http://exception.infra.fc.ofss.com">
```

```
<soapenv:Header />
- <soapenv:Body>
- <exp:fetchAllAndExportExportImport>
- <exp:sessionContext>
<con:bankCode>48</con:bankCode>
<con:businessUnit>MODELBANK</con:businessUnit>
<con:channel>BRN</con:channel>
<con:marketEntity>MODEL01</con:marketEntity>
<con:postingDateText>20130228000000</con:postingDateText>
<con:targetUnit>MODELBANK</con:targetUnit>
<con:transactionBranch>8542</con:transactionBranch>
<con:userId>OFSSUser</con:userId>
</exp:sessionContext>
<exp:taskCode>PM031</exp:taskCode>
</exp:fetchAllAndExportExportImport>
</soapenv:Body>
</soapenv:Envelope>
```

Export Response

Once this service is invoked with the above request, it fetches the configVersionNo of the exported data in response which is the version number with which Export All was performed.

```
- <S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
- <S:Body>
- <ns11:fetchAllAndExportExportImportResponse
xmlns:ns11="http://eximp.service.ops.app.fc.ofss.com/ExportImportA
pplicationService" xmlns:ns10="http://fact.enumeration.fc.ofss.com"
xmlns:ns9="http://enumeration.fc.ofss.com"
xmlns:exceptioninfra="http://exception.infra.fc.ofss.com"
xmlns:datatype="http://datatype.fc.ofss.com"
xmlns:contextapp="http://context.app.fc.ofss.com"
xmlns:dtocoreseedopsapp="http://dto.core.seed.ops.app.fc.ofss.com"
xmlns:dtocommondomainframework="http://dto.common.domain.framework
.fc.ofss.com"
xmlns:errorvalidationinfra="http://error.validation.infra.fc.ofss.
com" xmlns:opsapp="http://ops.app.fc.ofss.com"
xmlns:responseservice="http://response.service.fc.ofss.com">
- <ns11:return>

<responseservice:configVersionId>104</responseservice:configVersio
nId>
- <responseservice:status>
<responseservice:errorCode>0</responseservice:errorCode>
<responseservice:extendedReply />

<responseservice:internalReferenceNumber>2016075018875027</respons
eservice:internalReferenceNumber>
<responseservice:isOverriden>false</responseservice:isOverriden>
```

```
<responseservice:isServiceChargeApplied>false</responseservice:isS
erviceChargeApplied>
- <responseservice:postingDate>
<datatype:dateString>20130228000000</datatype:dateString>
<datatype:month>2</datatype:month>
<datatype:monthDate>228</datatype:monthDate>
<datatype:monthDateTime>228000000</datatype:monthDateTime>
<datatype:timestamp>2013-02-28T00:00:00+05:30</datatype:timestamp>
<datatype:year>2013</datatype:year>
</responseservice:postingDate>
<responseservice:replyCode>0</responseservice:replyCode>
<responseservice:replyText>Operation completed
successfully.</responseservice:replyText>
<responseservice:spReturnValue>0</responseservice:spReturnValue>
</responseservice:status>
</ns11:return>
</ns11:fetchAllAndExportExportImportResponse>
</S:Body>
</S:Envelope>
```

The user can import the required data based on this version number.

8.5 Import All

This operation aims at importing the record for the given taskCode and configVersionNo in the target environment. This method fetches the exported record based on versionNo and taskCode and tries to update if the records exist, else create the new records.

This is carried out by making a web service call to importAll method of ExportImportApplicationService with taskCode, versionNo as input. The request and response xml are as attached.

Import Request

```
- <soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:exp="http://eximp.service.ops.app.fc.ofss.com/ExportImportAp
plicationService" xmlns:con="http://context.app.fc.ofss.com"
xmlns:exc="http://exception.infra.fc.ofss.com">
<soapenv:Header />
- <soapenv:Body>
- <exp:importAllExportImport>
- <exp:sessionContext>
<con:bankCode>48</con:bankCode>
<con:businessUnit>MODELBANK</con:businessUnit>
<con:channel>BRN</con:channel>
<con:marketEntity>MODEL01</con:marketEntity>
<con:postingDateText>20130228000000</con:postingDateText>
<con:targetUnit>MODELBANK</con:targetUnit>
<con:transactionBranch>8542</con:transactionBranch>
<con:userId>OFSSUser</con:userId>
</exp:sessionContext>
```

```
<exp:taskCode>PM031</exp:taskCode>
<exp:versionNo>104</exp:versionNo>
</exp:importAllExportImport>
</soapenv:Body>
</soapenv:Envelope>
```

Import Respose

```
- <S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
- <S:Body>
- <ns5:importAllExportImportResponse
xmlns="http://enumeration.fc.ofss.com"
xmlns:ns2="http://fact.enumeration.fc.ofss.com"
xmlns:ns3="http://context.app.fc.ofss.com"
xmlns:ns4="http://exception.infra.fc.ofss.com"
xmlns:ns5="http://eximp.service.ops.app.fc.ofss.com/ExportImportAp
plicationService"
xmlns:ns6="http://dto.common.domain.framework.fc.ofss.com"
xmlns:ns7="http://datatype.fc.ofss.com"
xmlns:ns8="http://ops.app.fc.ofss.com"
xmlns:ns9="http://response.service.fc.ofss.com"
xmlns:ns10="http://error.validation.infra.fc.ofss.com"
xmlns:ns11="http://dto.core.config.app.fc.ofss.com">
- <ns5:return>
<ns9:errorCode>0</ns9:errorCode>
<ns9:extendedReply />

<ns9:internalReferenceNumber>2012132010145535</ns9:internalReferen
ceNumber>
<ns9:isOverriden>false</ns9:isOverriden>
<ns9:isServiceChargeApplied>false</ns9:isServiceChargeApplied>
- <ns9:postingDate>
<ns7:month>4</ns7:month>
<ns7:monthDate>425</ns7:monthDate>
<ns7:monthDateTime>425000000</ns7:monthDateTime>
<ns7:timestamp>2012-04-25T00:00:00+05:30</ns7:timestamp>
<ns7:year>2012</ns7:year>
</ns9:postingDate>
<ns9:replyCode>0</ns9:replyCode>
<ns9:replyText>Operation completed successfully.</ns9:replyText>
<ns9:spReturnValue>0</ns9:spReturnValue>
</ns5:return>
</ns5:importAllExportImportResponse>
</S:Body>
</S:Envelope>
```

8.6 Config Compare

This operation is used to compare Domain Objects, with same key, for a given taskCode. It aims at comparing the entities from two databases which are termed as TO and FROM database. The comparison is

such as it contains following information:

Present only in TO database (presently working environment)

Present only in FROM database (configurable DB environment)

Present in both, but data is different

In the whole set of operations, Export and Import can be performed either by screen or by webservice. For performing import using DB datastore, the reference DataSource needs to be configured in the target environment (the reference datasource is initially configured at the time of installation), which points to the data base where export has been performed. For ExportAll and ImportAll there is a common service ExportImportApplicationService which have the operation to perform the duties.

8.7 Data Store Configuration

The Data Store for Config operations can be configured to either Database or File. The user has the option to choose any one of the two data store configurations. The exported response will be stored in database or file as per this configuration.

8.7.1 DB Data Store

This configuration stores the exported data to database. For using this configuration, following changes have to be made:

1. In **FLX_FW_CONFIG_ALL_B** table, maintain **DataSourceType=File**

   select ***from flx_fw_config_all_b** where **category_id='DataSourceDestination'** and **prop_id='DataSourceType'**

   Note

   No separate configuration is required for export and import in case of DB Data Store.

8.7.2 File Data Store

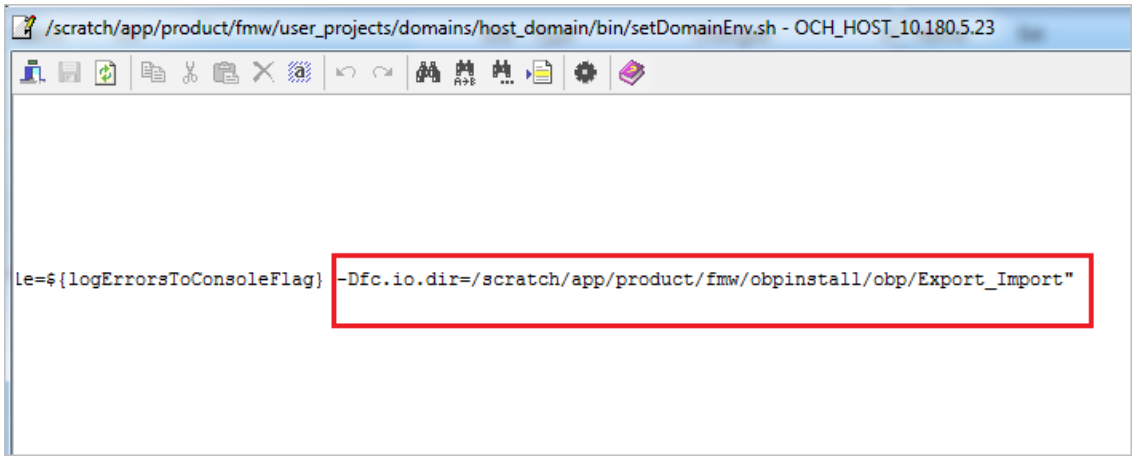
This configuration stores the exported data to file. For using this configuration, following changes have to be made:

1. In **FLX_FW_CONFIG_ALL_B table**, maintain **DataSourceType=FILE**.

   select ***from flx_fw_config_all_b** where **category_id='DataSourceDestination'** and **prop_id='DataSourceType'**

2. Set the location of system property **fc.io.dir** in **setDomainEnv.sh** to a valid directory.

Figure 8–1 File Data Store



8.7.2.1 Configuration for Export

The configuration for export requires the following changes:

1. Set **ExportLoggingPath** variable in **FLX_FW_CONFIG_ALL_B** to the directory where the exported files are needed to be stored. This is relative path with respect to **fc.io.dir/runarea/BusinessUnit**.

   select * **from flx_fw_config_all_b** where **category_id='DataSourceDestination'** and **prop_id='ExportLoggingPath'**.

   If **fc.io.dir** is set to /scratch/app/product/fmw/obpinstall/obp/Export_Import, **ExportLoggingPath** is set to **export** and **business unit** is **DEMO_BANK**, then the files will be stored at /scratch/app/product/fmw/obpinstall/obp/Export_Import/runarea/DEMO_BANK/export.

8.7.2.2 Configuration for Import

The configuration for import requires the following changes:

1. Set **ImportLoggingPath** variable in **FLX_FW_CONFIG_ALL_B** to the directory from where the exported response has to imported.

   select * **from flx_fw_config_all_b** where **category_id='DataSourceDestination'** and **prop_id='ImportLoggingPath'**

   /scratch/app/product/fmw/obpinstall/obp/Export_Import/runarea/DEMO_BANK/export will be set as **ImportLoggingPath** in our case.

8.8 How to Export Records

Export Operations can be performed using screen or webservice. This operation can be used to export either a single record or multiple records based on the requirement.

In case of DB Data Store, exported data is stored in **flx_ops_config_data_item** and for File Data Store, exported files are generated at the path specified for export configuration. For more information, see Chapter 8.7 Data Store Configuration.

8.8.1 Exporting Single Record

This operation is used to export single record of an entity.

Let us assume the configuration is done for **DATABASE**, so the data gets stored in **flx_ops_config_data_item** table of the source database. The entity inquiry response gets stored as a serialized byte into the database.

Export operation can be carried out as a webservice call for the export operation of the specific taskcode. A single record of business configuration can be exported using the service **<BusinessConfiguration>ApplicationService**, which provides a '**fetch<BusinessConfiguration>AndExport**' method.

The request parameters to this service are:

- SessionContext
- <BusinessConfiguration>DTO - Representing the key of the record to be exported.

  In response of the service call, it returns '**configVersionNo**'. This 'configVersionNo' will be used to import this record into the target environment.

Sample request and response are as follows:

Export Single Request

```
- <soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:act="http://action.service.ep.app.fc.ofss.com/ActivityEventA
ctionApplicationService" xmlns:con="http://context.app.fc.ofss.com"
xmlns:exc="http://exception.infra.fc.ofss.com"
xmlns:dto="http://dto.common.domain.framework.fc.ofss.com"
xmlns:dat="http://datatype.fc.ofss.com"
xmlns:act1="http://action.dto.ep.app.fc.ofss.com"
xmlns:rule="http://rule.action.dto.ep.app.fc.ofss.com"
xmlns:sub="http://subscriber.action.dto.ep.app.fc.ofss.com">
<soapenv:Header />
- <soapenv:Body>
- <act:fetchActivityEventActionAndExportActivityEventAction>
- <!-- Optional:
-->
- <act:sessionContext>
<con:bankCode>08</con:bankCode>
<con:businessUnit>OBP_BU</con:businessUnit>
<con:channel>BRN</con:channel>
<con:marketEntity>SUN01</con:marketEntity>
<con:postingDateText>20130228000000</con:postingDateText>
<con:targetUnit>OBP_BU</con:targetUnit>
<con:transactionBranch>089999</con:transactionBranch>
<con:userId>OFSSUser</con:userId>
</act:sessionContext>
- <!-- Optional:
-->
- <act:activityEventActionDTO>
```

```
- <act1:keyDTO>
- <!-- Optional:
-->
<act1:actionId>A</act1:actionId>
- <!-- Optional:
-->

<act1:activityId>com.ofss.fc.domain.lcm.batch.service.BatchCovenan
tService.processNotificationForCompliance</act1:activityId>
- <!-- Optional:
-->
<act1:eventId>LM_STATUS_COV</act1:eventId>
</act1:keyDTO>
</act:activityEventActionDTO>
</act:fetchActivityEventActionAndExportActivityEventAction>
</soapenv:Body>
</soapenv:Envelope>
```

Export Single Response

```
- <S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
- <S:Header>
<work:WorkContext
xmlns:work="http://oracle.com/weblogic/soap/workarea/">rO0ABXdWABx
3ZWJsb2dpYy5hcHAub2JwLXdlYnNlcnZpY2VzAAAA1gAAACN3ZWJsb2dpYy53b3JrY
XJlYS5TdHJpbmdXb3JrQ29udGV4dAAJMi4yLjAuMC4wAAA=</work:WorkContext>
</S:Header>
- <S:Body>
-
<ns14:fetchActivityEventActionAndExportActivityEventActionResponse
xmlns:ns14="http://action.service.ep.app.fc.ofss.com/ActivityEvent
ActionApplicationService"
xmlns:ns13="http://ep.enumeration.fc.ofss.com"
xmlns:ns12="http://enumeration.fc.ofss.com"
xmlns:exceptioninfra="http://exception.infra.fc.ofss.com"
xmlns:validationdtoapp="http://validation.dto.app.fc.ofss.com"
xmlns:ruleactiondtoepapp="http://rule.action.dto.ep.app.fc.ofss.co
m" xmlns:datatype="http://datatype.fc.ofss.com"
xmlns:contextapp="http://context.app.fc.ofss.com"
xmlns:dtocommondomainframework="http://dto.common.domain.framework
.fc.ofss.com"
xmlns:errorvalidationinfra="http://error.validation.infra.fc.ofss.
com" xmlns:actiondtoepapp="http://action.dto.ep.app.fc.ofss.com"
xmlns:responseservice="http://response.service.fc.ofss.com"
xmlns:userdtosmsapp="http://user.dto.sms.app.fc.ofss.com"
xmlns:subscriberactiondtoepapp="http://subscriber.action.dto.ep.ap
p.fc.ofss.com">
- <ns14:return>

<responseservice:configVersionId>187</responseservice:configVersio
nId>
```

```
- <responseservice:status>
<responseservice:errorCode>0</responseservice:errorCode>
<responseservice:extendedReply />

<responseservice:internalReferenceNumber>2016305031622005</respons
eservice:internalReferenceNumber>
<responseservice:isOverriden>false</responseservice:isOverriden>

<responseservice:isServiceChargeApplied>false</responseservice:isS
erviceChargeApplied>
- <responseservice:postingDate>
<datatype:dateString>20130228000000</datatype:dateString>
</responseservice:postingDate>
<responseservice:replyCode>0</responseservice:replyCode>
<responseservice:replyText>Operation completed
successfully.</responseservice:replyText>
<responseservice:spReturnValue>0</responseservice:spReturnValue>
</responseservice:status>
- <actiondtoepapp:activityEventActionDTO>

<dtocommondomainframework:auditSequence>1</dtocommondomainframewor
k:auditSequence>

<dtocommondomainframework:createdBy>ArvindKu</dtocommondomainframe
work:createdBy>
- <dtocommondomainframework:creationDate>
<datatype:dateString>20130809000000</datatype:dateString>
</dtocommondomainframework:creationDate>

<dtocommondomainframework:generatedPackageId>false</dtocommondomai
nframework:generatedPackageId>

<dtocommondomainframework:lastUpdatedBy>OFSSUser</dtocommondomainf
ramework:lastUpdatedBy>
- <dtocommondomainframework:lastUpdatedDate>
<datatype:dateString>20140721162124</datatype:dateString>
</dtocommondomainframework:lastUpdatedDate>

<dtocommondomainframework:version>2</dtocommondomainframework:vers
ion>
<actiondtoepapp:alertName>Covenant Status
Complied</actiondtoepapp:alertName>
- <actiondtoepapp:alertTemplate>

<dtocommondomainframework:auditSequence>1</dtocommondomainframewor
k:auditSequence>

<dtocommondomainframework:generatedPackageId>false</dtocommondomai
nframework:generatedPackageId>
```

```
<dtocommondomainframework:version>1</dtocommondomainframework:vers
ion>
- <actiondtoepapp:keyDTO>
<actiondtoepapp:id>1</actiondtoepapp:id>
</actiondtoepapp:keyDTO>
<actiondtoepapp:importance>CRITICAL</actiondtoepapp:importance>
<actiondtoepapp:language>ENG</actiondtoepapp:language>
<actiondtoepapp:name>Email Template</actiondtoepapp:name>
<actiondtoepapp:urgency>HIGH</actiondtoepapp:urgency>
</actiondtoepapp:alertTemplate>
<actiondtoepapp:alertType>MANDATORY</actiondtoepapp:alertType>
- <actiondtoepapp:decisionAgent>

<dtocommondomainframework:auditSequence>1</dtocommondomainframewor
k:auditSequence>

<dtocommondomainframework:generatedPackageId>false</dtocommondomai
nframework:generatedPackageId>

<dtocommondomainframework:version>1</dtocommondomainframework:vers
ion>
- <ruleactiondtoepapp:keyDTO>
<ruleactiondtoepapp:id>0</ruleactiondtoepapp:id>
</ruleactiondtoepapp:keyDTO>
- <ruleactiondtoepapp:rule>

<dtocommondomainframework:auditSequence>1</dtocommondomainframewor
k:auditSequence>

<dtocommondomainframework:generatedPackageId>false</dtocommondomai
nframework:generatedPackageId>

<dtocommondomainframework:version>1</dtocommondomainframework:vers
ion>
<ruleactiondtoepapp:description>Invokes the default
rule</ruleactiondtoepapp:description>
<ruleactiondtoepapp:keyDTO />
<ruleactiondtoepapp:name>defaultRule</ruleactiondtoepapp:name>

<ruleactiondtoepapp:ruleClass>com.ofss.fc.domain.ep.service.action
.rule.DefaultRuleHandler</ruleactiondtoepapp:ruleClass>

<ruleactiondtoepapp:ruleEngine>INTERNAL</ruleactiondtoepapp:ruleEn
gine>
</ruleactiondtoepapp:rule>
</actiondtoepapp:decisionAgent>
- <actiondtoepapp:expiryDate>
<datatype:dateString>20991231000000</datatype:dateString>
```

```
</actiondtoepapp:expiryDate>
<actiondtoepapp:isConditional>false</actiondtoepapp:isConditional>
<actiondtoepapp:isRetryAllowed>true</actiondtoepapp:isRetryAllowed>

<actiondtoepapp:isTransactional>false</actiondtoepapp:isTransactio
nal>
- <actiondtoepapp:keyDTO>
<actiondtoepapp:actionId>A</actiondtoepapp:actionId>

<actiondtoepapp:activityId>com.ofss.fc.domain.lcm.batch.service.Ba
tchCovenantService.processNotificationForCompliance</actiondtoepap
p:activityId>
<actiondtoepapp:eventId>LM_STATUS_COV</actiondtoepapp:eventId>
</actiondtoepapp:keyDTO>
<actiondtoepapp:maxRetryCount>2</actiondtoepapp:maxRetryCount>
- <actiondtoepapp:recipientMessageTemplates>

<dtocommondomainframework:auditSequence>1</dtocommondomainframewor
k:auditSequence>

<dtocommondomainframework:generatedPackageId>false</dtocommondomai
nframework:generatedPackageId>

<dtocommondomainframework:version>1</dtocommondomainframework:vers
ion>

<subscriberactiondtoepapp:amount>0</subscriberactiondtoepapp:amoun
t>

<subscriberactiondtoepapp:bankerType>NA</subscriberactiondtoepapp:
bankerType>

<subscriberactiondtoepapp:conditional>false</subscriberactiondtoep
app:conditional>
- <subscriberactiondtoepapp:decisionAgent>

<dtocommondomainframework:auditSequence>1</dtocommondomainframewor
k:auditSequence>

<dtocommondomainframework:generatedPackageId>false</dtocommondomai
nframework:generatedPackageId>

<dtocommondomainframework:version>1</dtocommondomainframework:vers
ion>
- <ruleactiondtoepapp:keyDTO>
<ruleactiondtoepapp:id>0</ruleactiondtoepapp:id>
</ruleactiondtoepapp:keyDTO>
- <ruleactiondtoepapp:rule>
```

```
<dtocommondomainframework:auditSequence>1</dtocommondomainframewor
k:auditSequence>

<dtocommondomainframework:generatedPackageId>false</dtocommondomai
nframework:generatedPackageId>

<dtocommondomainframework:version>1</dtocommondomainframework:vers
ion>
<ruleactiondtoepapp:description>Invokes the default
rule</ruleactiondtoepapp:description>
<ruleactiondtoepapp:keyDTO />
<ruleactiondtoepapp:name>defaultRule</ruleactiondtoepapp:name>

<ruleactiondtoepapp:ruleClass>com.ofss.fc.domain.ep.service.action
.rule.DefaultRuleHandler</ruleactiondtoepapp:ruleClass>

<ruleactiondtoepapp:ruleEngine>INTERNAL</ruleactiondtoepapp:ruleEn
gine>
</ruleactiondtoepapp:rule>
</subscriberactiondtoepapp:decisionAgent>
- <subscriberactiondtoepapp:keyDTO>

<subscriberactiondtoepapp:actionId>A</subscriberactiondtoepapp:act
ionId>

<subscriberactiondtoepapp:activityId>com.ofss.fc.domain.lcm.batch.
service.BatchCovenantService.processNotificationForCompliance</sub
scriberactiondtoepapp:activityId>

<subscriberactiondtoepapp:destinationType>EMAIL</subscriberactiond
toepapp:destinationType>
<subscriberactiondtoepapp:eventId>LM_STATUS_
COV</subscriberactiondtoepapp:eventId>
<subscriberactiondtoepapp:messageTemplateId>LCM_Covenant status is
Complied</subscriberactiondtoepapp:messageTemplateId>

<subscriberactiondtoepapp:subscriberType>PARTY</subscriberactiondt
oepapp:subscriberType>

<subscriberactiondtoepapp:subscriberValue>CUSTOMER</subscriberacti
ondtoepapp:subscriberValue>
</subscriberactiondtoepapp:keyDTO>

<subscriberactiondtoepapp:recipientType>INTERNAL</subscriberaction
dtoepapp:recipientType>
</actiondtoepapp:recipientMessageTemplates>
</actiondtoepapp:activityEventActionDTO>
</ns14:return>
```

```
</ns14:fetchActivityEventActionAndExportActivityEventActionRespons
e>
</S:Body>
</S:Envelope>
```

8.8.2 Exporting All Records

This operation is used to export all the entities of a given task code. The exported package can then be replicated into the target environment. All records of a Business configuration entity can be exported using the **FetchAllAndExport** method of **ExportImportApplicationService**.

The request parameters to this service are:

- **SessionContext**
- **TaskCode**

A '**configVersionNo**' is returned in the response. This '**configVersionNo**' will be used as an identifier to trigger an import into the target environment.

Sample request and response are as follows:

Export All Request

```
- <soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:exp="http://eximp.service.ops.app.fc.ofss.com/ExportImportAp
plicationService" xmlns:con="http://context.app.fc.ofss.com"
xmlns:exc="http://exception.infra.fc.ofss.com">
<soapenv:Header />
- <soapenv:Body>
- <exp:fetchAllAndExportExportImport>
- <!-- Optional:
-->
- <exp:sessionContext>
<con:bankCode>08</con:bankCode>
<con:businessUnit>OBP_BU</con:businessUnit>
<con:channel>BRN</con:channel>
<con:marketEntity>SUN01</con:marketEntity>
<con:postingDateText>20130228000000</con:postingDateText>
<con:targetUnit>OBP_BU</con:targetUnit>
<con:transactionBranch>089999</con:transactionBranch>
<con:userId>OFSSUser</con:userId>
</exp:sessionContext>
<exp:taskCode>AL04</exp:taskCode>
</exp:fetchAllAndExportExportImport>
</soapenv:Body>
</soapenv:Envelope>
```

Export All Response

```
- <S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
- <S:Header>
```

```
<work:WorkContext
xmlns:work="http://oracle.com/weblogic/soap/workarea/">rO0ABXdWABx
3ZWJsb2dpYy5hcHAub2JwLXdlYnNlcnZpY2VzAAAA1gAAACN3ZWJsb2dpYy53b3JrY
XJlYS5TdHJpbmdXb3JrQ29udGV4dAAJMi4yLjAuMC4wAAA=</work:WorkContext>
</S:Header>
- <S:Body>
- <ns13:fetchAllAndExportExportImportResponse
xmlns:ns13="http://eximp.service.ops.app.fc.ofss.com/ExportImportA
pplicationService" xmlns:ns12="http://ops.enumeration.fc.ofss.com"
xmlns:ns11="http://fact.enumeration.fc.ofss.com"
xmlns:ns10="http://enumeration.fc.ofss.com"
xmlns:exceptioninfra="http://exception.infra.fc.ofss.com"
xmlns:validationdtoapp="http://validation.dto.app.fc.ofss.com"
xmlns:datatype="http://datatype.fc.ofss.com"
xmlns:contextapp="http://context.app.fc.ofss.com"
xmlns:dtocoreseedopsapp="http://dto.core.seed.ops.app.fc.ofss.com"
xmlns:dtocommondomainframework="http://dto.common.domain.framework
.fc.ofss.com"
xmlns:errorvalidationinfra="http://error.validation.infra.fc.ofss.
com" xmlns:opsapp="http://ops.app.fc.ofss.com"
xmlns:responseservice="http://response.service.fc.ofss.com">
- <ns13:return>

<responseservice:configVersionId>186</responseservice:configVersio
nId>
- <responseservice:status>
<responseservice:errorCode>0</responseservice:errorCode>
<responseservice:extendedReply />

<responseservice:internalReferenceNumber>2016305031622003</respons
eservice:internalReferenceNumber>
<responseservice:isOverriden>false</responseservice:isOverriden>

<responseservice:isServiceChargeApplied>false</responseservice:isS
erviceChargeApplied>
- <responseservice:postingDate>
<datatype:dateString>20130228000000</datatype:dateString>
</responseservice:postingDate>
<responseservice:replyCode>0</responseservice:replyCode>
<responseservice:replyText>Operation completed
successfully.</responseservice:replyText>
<responseservice:spReturnValue>0</responseservice:spReturnValue>
</responseservice:status>
</ns13:return>
</ns13:fetchAllAndExportExportImportResponse>
</S:Body>
</S:Envelope>
```

In case of DB Data Store, exported data is stored in **flx_ops_config_data_item** and for File Data Store, exported files are generated at the path specified for export configuration. For more information, see Chapter 8.7 Data Store Configuration.

Figure 8–2 Exported Data

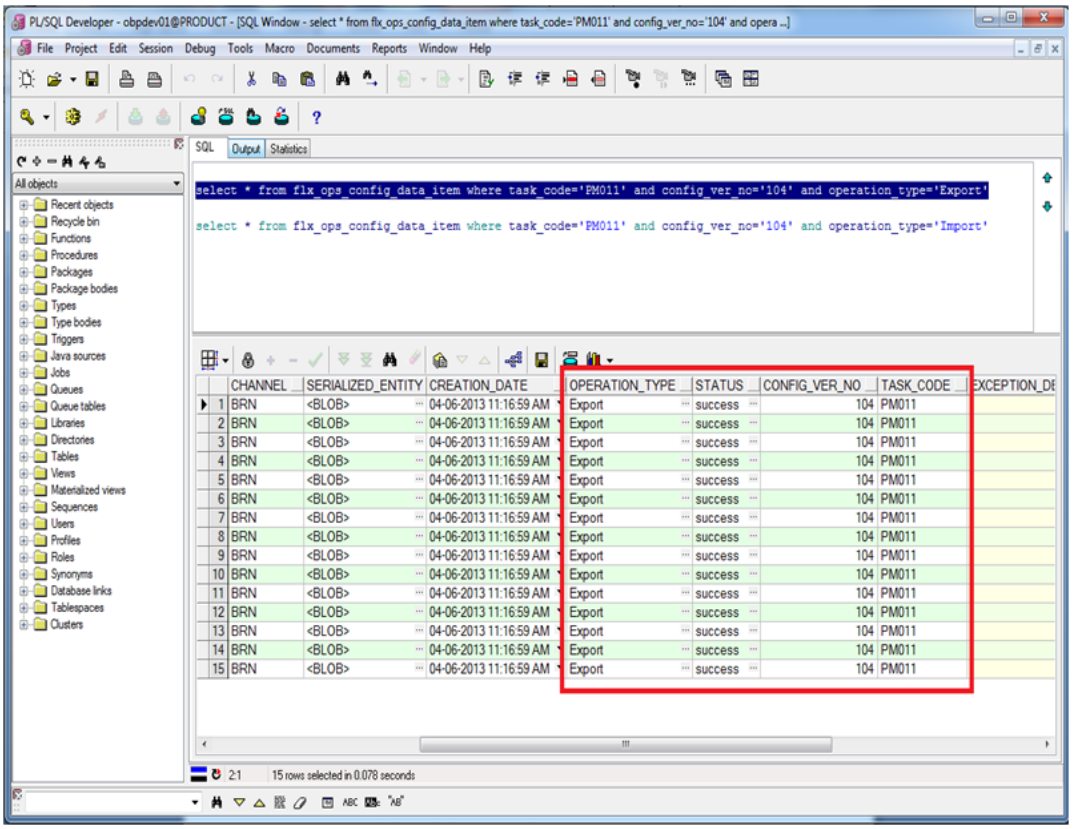
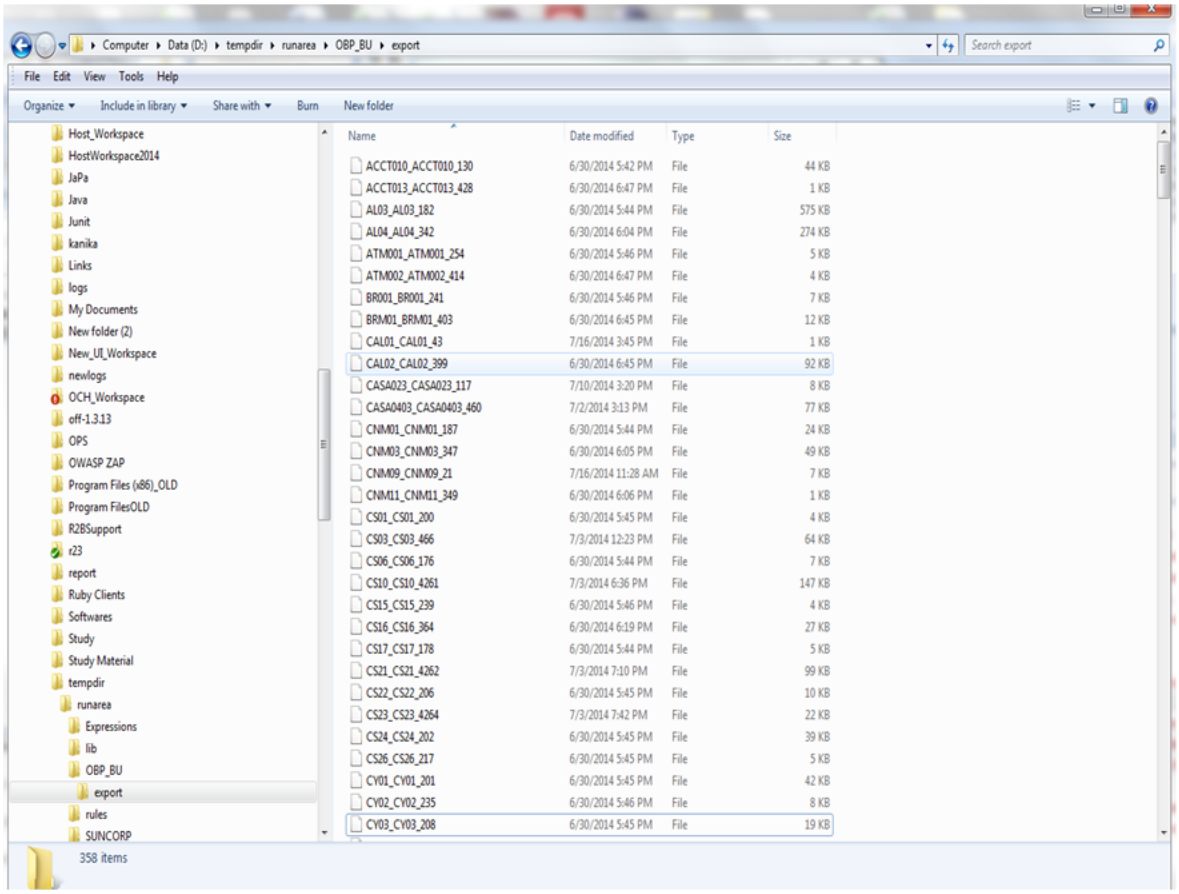


Figure 8–3 Exported Files



8.9 How to Import Records

Import Operations can be performed using webservice clients. This operation can be used to import either a single record or multiple records based on the requirement.

8.9.1 Importing Single Record

This operation is used to import single record of an configuration.

8.9.1.1 Using API Client

A single record of a business configuration entity can be imported using the **ExportImportApplicationService**, which provides an '**importAll**' method.

The request parameters to this service are:

- SessionContext
- TaskCode
- configVersionNo (from Export Single Record response)

The steps to import single record using API client are same as importing all records. These are mentioned in the further section.

Sample request and response are as below:

Import All Request

```
- <soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:exp="http://eximp.service.ops.app.fc.ofss.com/ExportImportAp
plicationService" xmlns:con="http://context.app.fc.ofss.com"
xmlns:exc="http://exception.infra.fc.ofss.com">
<soapenv:Header />
- <soapenv:Body>
- <exp:importAllExportImport>
- <!-- Optional:
-->
- <exp:sessionContext>
<con:bankCode>08</con:bankCode>
<con:businessUnit>OBP_BU</con:businessUnit>
<con:channel>BRN</con:channel>
<con:marketEntity>SUN01</con:marketEntity>
<con:postingDateText>20130228000000</con:postingDateText>
<con:targetUnit>OBP_BU</con:targetUnit>
<con:transactionBranch>089999</con:transactionBranch>
<con:userId>OFSSUser</con:userId>
</exp:sessionContext>
<exp:taskCode>AL04</exp:taskCode>
<exp:versionNo>186</exp:versionNo>
</exp:importAllExportImport>
</soapenv:Body>
</soapenv:Envelope>
```

Import All Response

```
- <S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
- <S:Header>
<work:WorkContext
xmlns:work="http://oracle.com/weblogic/soap/workarea/">rO0ABXdWABx
3ZWJsb2dpYy5hcHAub2JwLXdlYnNlcnZpY2VzAAAA1gAAACN3ZWJsb2dpYy53b3JrY
XJlYS5TdHJpbmdXb3JrQ29udGV4dAAJMi4yLjAuMC4wAAA=</work:WorkContext>
</S:Header>
- <S:Body>
- <ns13:importAllExportImportResponse
xmlns:ns13="http://eximp.service.ops.app.fc.ofss.com/ExportImportA
pplicationService" xmlns:ns12="http://ops.enumeration.fc.ofss.com"
xmlns:ns11="http://fact.enumeration.fc.ofss.com"
xmlns:ns10="http://enumeration.fc.ofss.com"
xmlns:exceptioninfra="http://exception.infra.fc.ofss.com"
xmlns:validationdtoapp="http://validation.dto.app.fc.ofss.com"
xmlns:datatype="http://datatype.fc.ofss.com"
xmlns:contextapp="http://context.app.fc.ofss.com"
xmlns:dtocoreseedopsapp="http://dto.core.seed.ops.app.fc.ofss.com"
xmlns:dtocommondomainframework="http://dto.common.domain.framework
.fc.ofss.com"
```

```
xmlns:errorvalidationinfra="http://error.validation.infra.fc.ofss.
com" xmlns:opsapp="http://ops.app.fc.ofss.com"
xmlns:responseservice="http://response.service.fc.ofss.com">
- <ns13:return>
<responseservice:errorCode>0</responseservice:errorCode>
<responseservice:extendedReply />

<responseservice:internalReferenceNumber>2016305031622004</respons
eservice:internalReferenceNumber>
<responseservice:isOverriden>false</responseservice:isOverriden>

<responseservice:isServiceChargeApplied>false</responseservice:isS
erviceChargeApplied>
<responseservice:replyCode>0</responseservice:replyCode>
<responseservice:spReturnValue>0</responseservice:spReturnValue>
</ns13:return>
</ns13:importAllExportImportResponse>
</S:Body>
</S:Envelope>
```

8.9.2 Importing All Records

This operation is used to import the records belonging to the given '**TaskCode**' and '**configVersionNo**' into the target environment. This method fetches the exported records based on the '**configVersionNo**' and '**TaskCode**', and upserts the same into the target environment. All records of a Business configuration entity can be imported using the **ImportAll** method of **ExportImportApplicationService**.

The request parameters to this service are:

- TaskCode
- ConfigVersionNo

For performing the import operation, the '**Config Data Source**' needs to be configured in the target environment, this datasource points to the database of the reference environments.

Sample request and response are as below:

Import All Request

```
- <soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:exp="http://eximp.service.ops.app.fc.ofss.com/ExportImportAp
plicationService" xmlns:con="http://context.app.fc.ofss.com"
xmlns:exc="http://exception.infra.fc.ofss.com">
<soapenv:Header />
- <soapenv:Body>
- <exp:importAllExportImport>
- <!-- Optional:
-->
- <exp:sessionContext>
<con:bankCode>08</con:bankCode>
<con:businessUnit>OBP_BU</con:businessUnit>
```

```
<con:channel>BRN</con:channel>
<con:marketEntity>SUN01</con:marketEntity>
<con:postingDateText>20130228000000</con:postingDateText>
<con:targetUnit>OBP_BU</con:targetUnit>
<con:transactionBranch>089999</con:transactionBranch>
<con:userId>OFSSUser</con:userId>
</exp:sessionContext>
<exp:taskCode>AL04</exp:taskCode>
<exp:versionNo>186</exp:versionNo>
</exp:importAllExportImport>
</soapenv:Body>
</soapenv:Envelope>
```

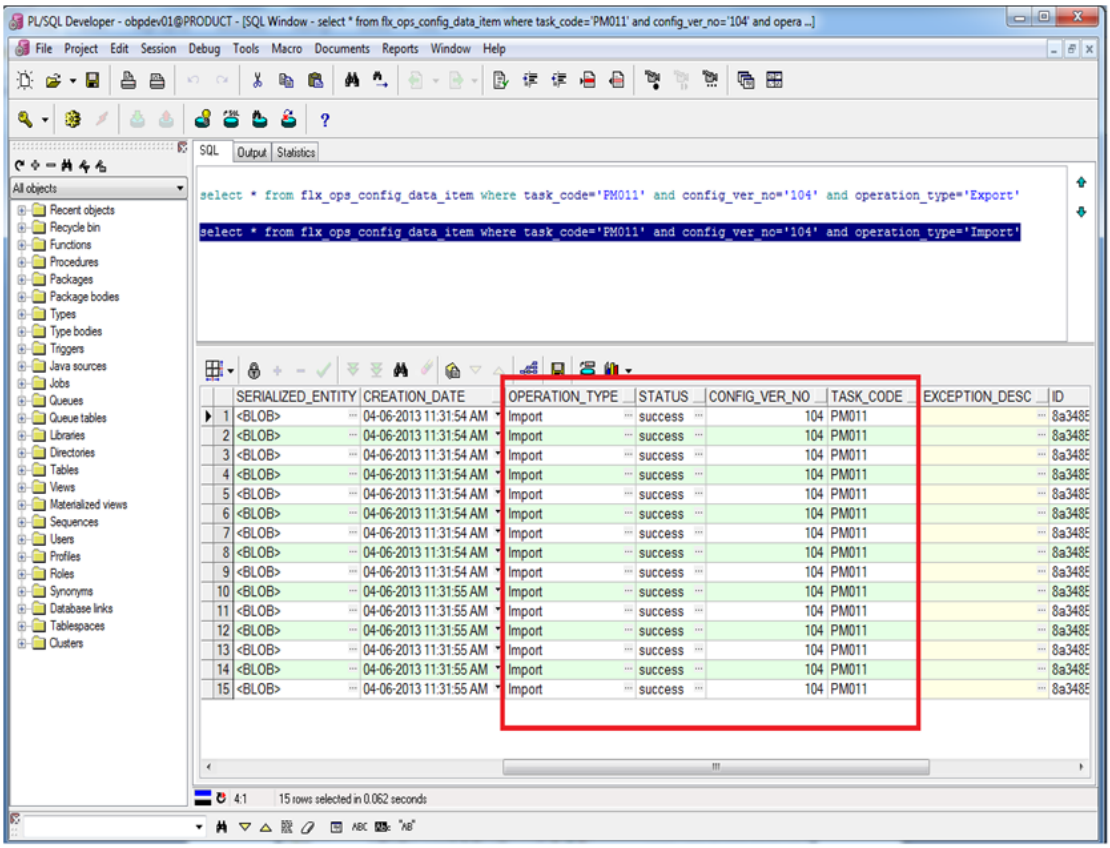
Import All Response

```
- <S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
- <S:Header>
<work:WorkContext
xmlns:work="http://oracle.com/weblogic/soap/workarea/">rO0ABXdWABx
3ZWJsb2dpYy5hcHAub2JwLXdlYnNlcnZpY2VzAAAA1gAAACN3ZWJsb2dpYy53b3JrY
XJlYS5TdHJpbmdXb3JrQ29udGV4dAAJMi4yLjAuMC4wAAA=</work:WorkContext>
</S:Header>
- <S:Body>
- <ns13:importAllExportImportResponse
xmlns:ns13="http://eximp.service.ops.app.fc.ofss.com/ExportImportA
pplicationService" xmlns:ns12="http://ops.enumeration.fc.ofss.com"
xmlns:ns11="http://fact.enumeration.fc.ofss.com"
xmlns:ns10="http://enumeration.fc.ofss.com"
xmlns:exceptioninfra="http://exception.infra.fc.ofss.com"
xmlns:validationdtoapp="http://validation.dto.app.fc.ofss.com"
xmlns:datatype="http://datatype.fc.ofss.com"
xmlns:contextapp="http://context.app.fc.ofss.com"
xmlns:dtocoreseedopsapp="http://dto.core.seed.ops.app.fc.ofss.com"
xmlns:dtocommondomainframework="http://dto.common.domain.framework
.fc.ofss.com"
xmlns:errorvalidationinfra="http://error.validation.infra.fc.ofss.
com" xmlns:opsapp="http://ops.app.fc.ofss.com"
xmlns:responseservice="http://response.service.fc.ofss.com">
- <ns13:return>
<responseservice:errorCode>0</responseservice:errorCode>
<responseservice:extendedReply />

<responseservice:internalReferenceNumber>2016305031622004</respons
eservice:internalReferenceNumber>
<responseservice:isOverriden>false</responseservice:isOverriden>

<responseservice:isServiceChargeApplied>false</responseservice:isS
erviceChargeApplied>
<responseservice:replyCode>0</responseservice:replyCode>
<responseservice:spReturnValue>0</responseservice:spReturnValue>
```

```
</ns13:return>
</ns13:importAllExportImportResponse>
</S:Body>
</S:Envelope>
```

Database entry for imported response will be stored in **flx_ops_config_data_item**.

Figure 8–4 Importing Data Using SOAP UI - Storing Response



8.10 Configuration Comparison

This section describes the details of configuration comparison.

8.10.1 Compare Business Configuration (Fast Path: OPA005)

This page is used to compare two entities on basis of its content.

It provides facility to compare Domain Objects, with same key, for a given task code. It aims at comparing the entities from two database which are termed as **TO** and **FROM** database. The comparison is such as it contains following information:

- Present only in TO database (presently working environment).
- Present only in FROM database (configurable DB environment).

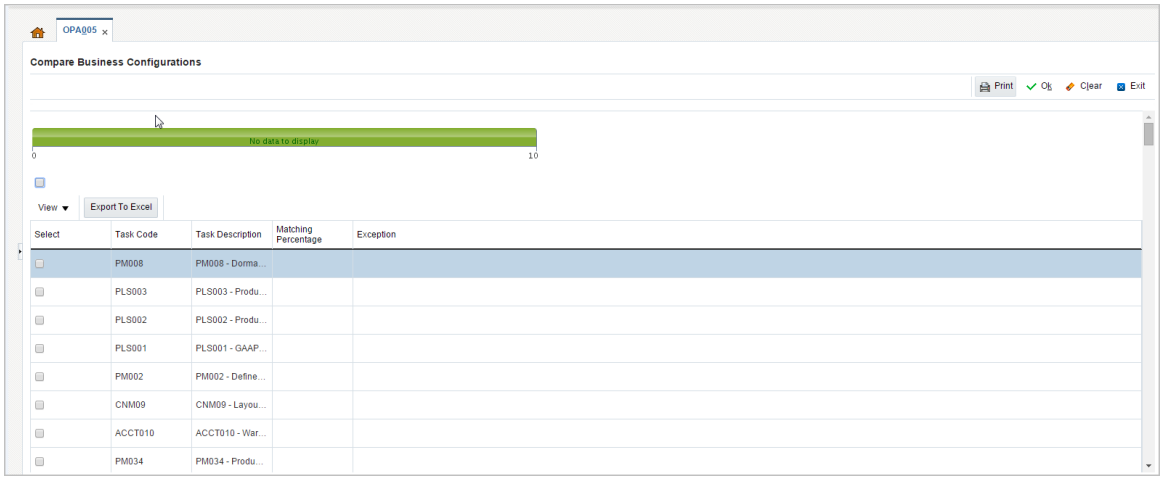
- Present in both, but data is different.

8.10.2 Usage

The comparison results among entities can be generated by performing the following steps:

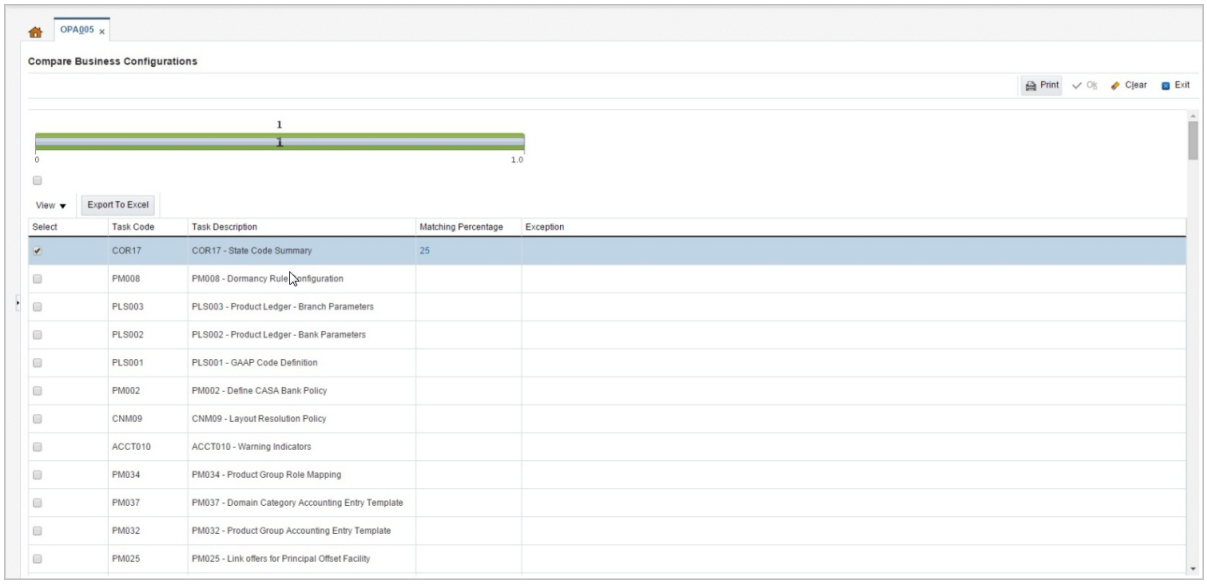
1. Open the OPA005 page, which loads all the entities configured in the table **flx_ops_task_defn**.

Figure 8–5 Entity Comparison



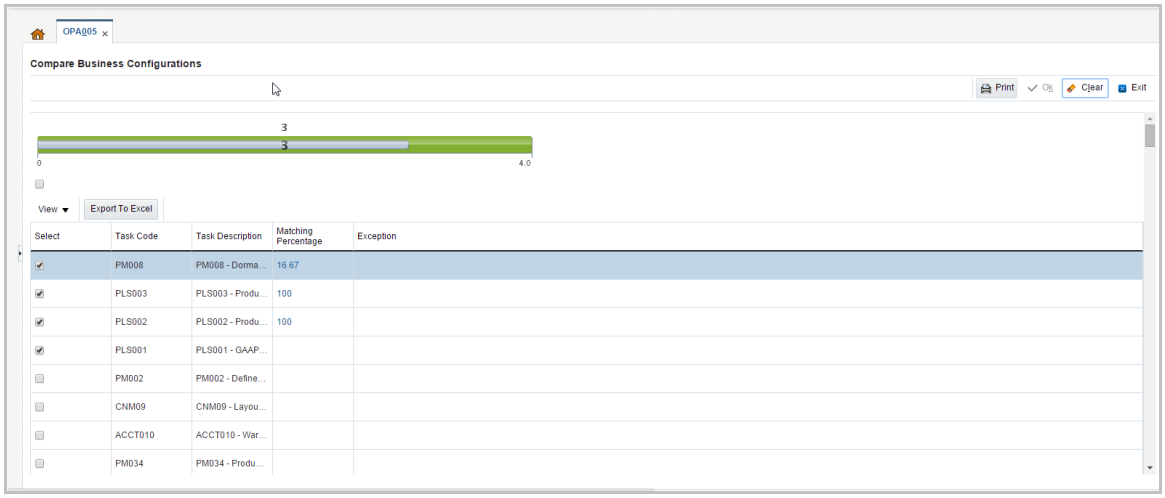
2. Select the option under **Select** column to do the comparison of configuration/configurations which shows the matching % of data in the two environment.

Figure 8–6 Entity Comparison Results



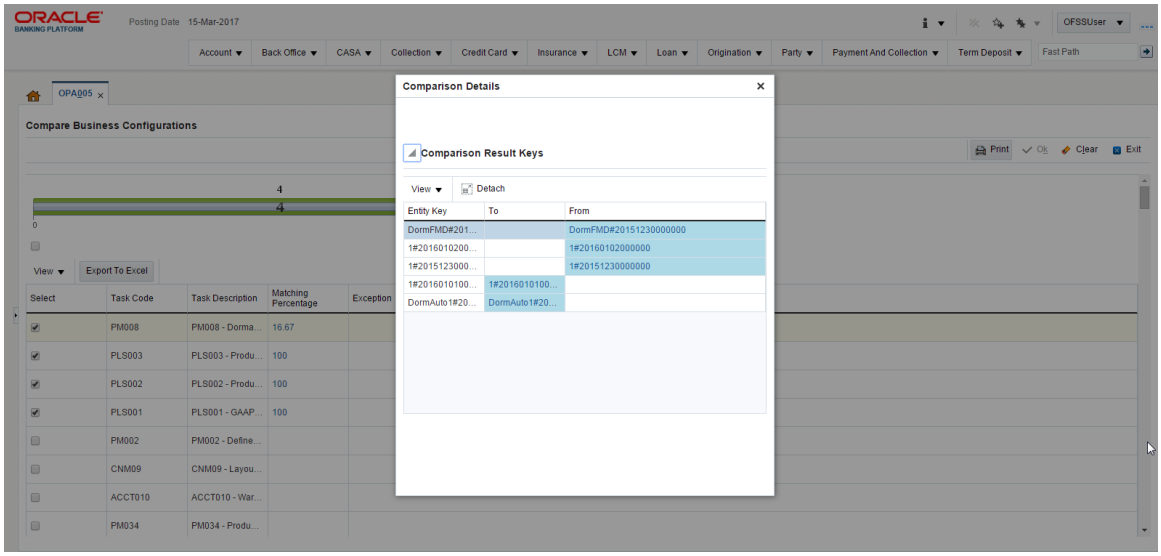
Progress bar denoting that the compare operation has finished.

Figure 8–7 Progress Bar



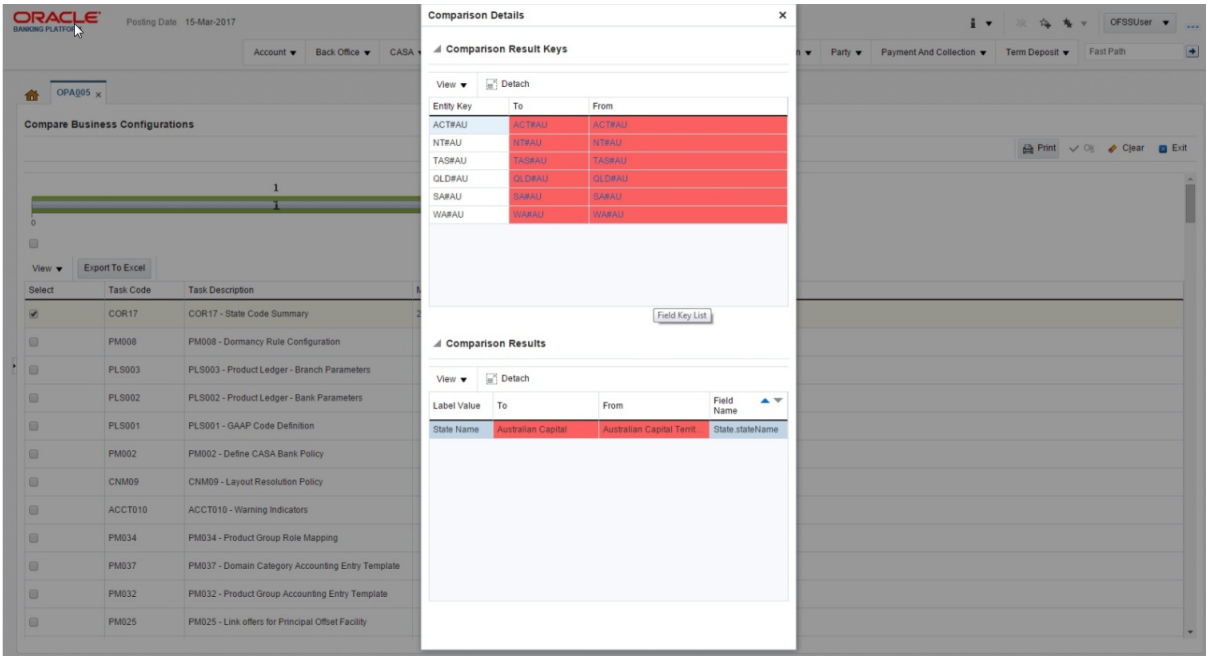
3. Select the % match to launch a pop up which shows the comparison result with different keys associated to it.

Figure 8–8 Comparison Details



4. Select any of the key to see the difference of its associated attributes in the two environment.

Figure 8–9 Attributes Difference



8.11 Application Configuration

This section describes the application configuration details.

8.11.1 Cache Configuration

Configuration cache is where we cache configuration information (stored in the configuration tables in database or some files) for every application on each server in the farm.

The entire application configuration to be cached is pre-defined in **Preferences.xml**.

Sample entries in Preferences.xml are as below:

Preferences.xml

```
<Preferences>
<Nodes>
<Preference name="jdbcpreference"
PreferencesProvider="com.ofss.fc.infra.config.impl.PropertiesFileC
onfigProvider" parent="" propertyFileName="jdbc.properties"
syncTimeInterval="600000" />
<Preference name="ConfigurationVariable"
PreferencesProvider="com.ofss.fc.infra.config.impl.DBBasedProperty
Provider" parent="jdbcpreference" propertyFileName="select prop_id,
prop_value from flx_fw_config_var_b" syncTimeInterval="600000" />
<Preference name="ChannelConstants"
PreferencesProvider="com.ofss.fc.infra.config.impl.JavaConstantsCo
nfigProvider" parent="jdbcpreference"
propertyFileName="com.ofss.fc.common.ChannelConstantsConfiguration"
syncTimeInterval="600000" />
```

```
<Preference name="JSONServiceMap"
overriddenBy="JSONServiceMapOverride"
PreferencesProvider="com.ofss.fc.infra.config.impl.JavaConstantsCo
nfigProvider" parent="jdbcpreference"
propertyFileName="com.ofss.fc.common.JSONServiceConfig"
syncTimeInterval="600000" />

……..
</Nodes>
</Preferences>
```

Important parameters in preferences.xml are as follows:

- **PreferencesProvider**: DB based provider, File base provider or Java constant base provider.
- **propertyFileName**: Describes the configuration source. Either sql query, file name or fully qualified Java constant class name.
- **syncTimeInterval**: Refresh time
- **name**: Acts as configuration key in the cache
- **parent**: Enables building the dependency hierarchy
- **overriddenBy**: This parameter specifies the name of preference which will override the current one.

9 Batch Shells in OBPM

This chapter describes the batch shells used in OBPM and their execution sequence.

9.1 Batch Shells Description

The following table lists the batch shells along with their detailed description.

Table 9–1 Shell Description

Sr. No.	Process Category	Category Description	Shell	Shell Description	Module Code	Detailed Description
1	100	Health Check	health_ chk_shell	Health Checkup Shell	FW	This is dummy shell. It is used to check whether framework is ready to process batch and to check if there is any issue on framework or not.
2	100	Health Check	excep_ pending_ shell	Batch Exceptions Pending Check Shell	FW	This shell will check in exception log for all unprocessed records which marked as deferred. If any record is found, then this shell will be marked as aborted as there are still pending exception record available.
3	0	Reports Health Check	reports_ chk_shell	Reports Health Check	FW	This is dummy shell. It is used to check whether report framework is ready to process reports and to check if there is any issue on framework or not.
4	3	Cut Off	co_cutoff_ prologue	Cutoff Prologue	CO	This shell is used to indicate that the cut off has been started by setting the flg_cutoff_ run_today in flx_cs_branch_dates_b to 'Y'. (This flag is set back to 'N' at the end of the EOD category.)
5	1	End of Day	pi_srv_ord_ sta_eod	Service Order Deactivation EOD Shell	PI	This shell handles de-activation of Service Orders based on Service Order end date.
6	1	End of Day	eod_ report_shell	EOD Reports	AL	This shell generates reports in EOD.
7	1	End of Day	eod_	EOD	EO	This shell waits for all the reports to be generated and changes the Process date.

Sr. No.	Process Category	Category Description	Shell	Shell Description	Module Code	Detailed Description
			epilogue	Epilogue		
8	16	Internal System EOD	pi_eod_ shell	Party EOD Shell	PI	This shell is used to process the Party Due Diligence Expiry if the Expiry date is a holiday.
9	16	Internal System EOD	int_eod_ report_shell	Internal EOD Reports	AL	This shell generates Reports during Internal system EOD.
10	16	Internal System EOD	int_eod_ epilogue	Int EOD Epilogue	EO	This shell waits for all the reports to be generated.
11	2	Beginning of Day	pi_srv_ord_ sta_bod	Service Order Activation BOD Shell	PC	This shell handles activation of Service Orders based on Service Order start date.
12	2	Beginning of Day	wf_task_ resume_ shell	Human Task EOD Resume Shell	WF	
13	2	Beginning of Day	bod_ report_shell	BOD Reports	AL	This shell generates reports.
14	2	Beginning of Day	bod_ epilogue	BOD Epilogue	EO	This shell waits for all the reports to be generated.
15	117	Housekeeping	pi_bod_ shell	Party BOD Shell	PI	This shell is used to process the Party Due Diligence Expiry if it falls on a working day. This shell also updates the future party address getting active on this day. It updates such addresses to be current and marks the previous current address as past.
16	120	Alert Generation	ep_ generation_ shell	Alert generation Shell	DI	All pending alert requests is picked and processed. If the status of the alert is generated state, it means processing is pending for the alerts. This shell picks the alerts which are less than current date.
17	69	mdm_Publish	mdm_ publish_ shell	mdm Publish	IN	This shell publishes to OCH following status updates taken place during EOD: ■ Account closures during batch execution ■ Account opening during batch execution

Sr. No.	Process Category	Category Description	Shell	Shell Description	Module Code	Detailed Description
						■ KYC updates during batch execution
18	12	FSDF master data hand-off	handoff_ shell	Analytics Hand-Off Shell	DI	This shell initiates all ODI scenario execution defined in FLX_DI_ETL_JOB_ DEFINITION table.
19	12	FSDF master data hand-off	handoff_ initialise	Analytics Hand-Off Initialisation Shell	DI	This shell initializes the start time of CSA (Common staging area) data handoff time frame.
20	12	FSDF master data hand-off	epilogue_ shell	Analytics epilogue Shell	DI	This shell checks all running ODI scenario execution status. If any error occurs, the scenario execution is restarted after resolve.
21	13	FSDF EOD data hand-off	handoff_ shell	Analytics Hand-Off Shell	DI	This shell is used to transfer data for Common Services, Party, Loan, and PMU from OBP to CSA (Common staging area).
22	13	FSDF EOD data hand-off	epilogue_ shell	Analytics epilogue Shell	DI	This shell monitors the execution status of all ODI scenarios requested for execution during handoff_shell.
23	14	FSDF Txn data Hand-off	handoff_ shell	Analytics Hand-Off Shell	DI	This shell is used to transfer data for Accounting, DDA, LOAN, TD, and Facility from OBP to CSA (Common staging area).
24	14	FSDF Txn data Hand-off	epilogue_ shell	Analytics epilogue Shell	DI	This shell monitors the execution status of all ODI scenarios requested for execution during handoff_shell.
25	15	Analytics Batch Data Hand-off	handoff_ initialise	Analytics Hand-Off Initialisation Shell	DI	This shell initializes the start time of CSA (Common staging area) data handoff time frame. This shell is used when reporting db is present as source database.
26	15	Analytics Batch Data Hand-off	handoff_ shell	Analytics Hand-Off Shell	DI	This shell is used to transfer data for all the modules from OBP to CSA (Common staging area). This shell is used when reporting db is present as source database.

Sr. No.	Process Category	Category Description	Shell	Shell Description	Module Code	Detailed Description
27	15	Analytics Batch Data Hand-off	epilogue_ shell	Analytics epilogue Shell	DI	This shell monitors the execution status of all ODI scenarios requested for execution during handoff_shell. This shell is used when reporting db is present as source database.

9.2 Batch Shells Execution Sequence

The following table presents the execution sequence of the batch shells.

Table 9–2 Shell Execution Sequence

Sr. No.	Process Category	Category Description	Category Significance	Shell Execution Sequence	Shell	Shell Description	Module Code	Required Shells	Required Shell Description
1	100	Health Check	Optional	1	health_chk_ shell	Health Checkup Shell	FW		
2	100	Health Check	Optional	1	excep_ pending_ shell	Batch Exceptions Pending Check Shell	FW		
3	0	Reports Health Check	Optional	1	reports_chk_ shell	Reports Health Check	FW		
4	3	Cut Off	Mandatory	1	co_cutoff_ prologue	Cutoff Prologue	CO		
5	1	End of Day	Mandatory	1	pi_srv_ord_ sta_eod	Service Order Deactivation EOD Shell	PI		
6	1	End of Day	Mandatory	30	eod_report_ shell	EOD Reports	AL		
7	1	End of Day	Mandatory	32	eod_epilogue	EOD Epilogue	EO		
8	16	Internal System EOD	Mandatory	1	pi_eod_shell	Party EOD Shell	PI		
9	16	Internal System	Mandatory	16	int_eod_	Internal EOD Reports	AL		

Sr. No.	Process Category	Category Description	Category Significance	Shell Execution Sequence	Shell	Shell Description	Module Code	Required Shells	Required Shell Description
		EOD			report_shell				
10	16	Internal System EOD	Mandatory	17	int_eod_ epilogue	Int EOD Epilogue	EO		
11	2	Beginning of Day	Mandatory	1	pi_srv_ord_ sta_bod	Service Order Activation BOD Shell	PI		
12	2	Beginning of Day	Mandatory	1	wf_task_ resume_ shell	Human Task EOD Resume Shell	WF		
13	2	Beginning of Day	Mandatory	16	bod_report_ shell	BOD Reports	AL		
14	2	Beginning of Day	Mandatory	17	bod_epilogue	BOD Epilogue	EO		
15	117	Housekeeping	Mandatory	1	pi_bod_shell	Party BOD Shell	PI		
16	120	Alert Generation	Optional	1	ep_ generation_ shell	Alert generation Shell	DI		
17	69	mdm_Publish	Optional	1	mdm_ publish_shell	mdm Publish	IN		
18	12	FSDF master data hand-off	Optional	1	handoff_ shell	Analytics Hand-Off Shell	DI		
19	12	FSDF master data hand-off	Optional	2	handoff_ initialise	Analytics Hand-Off Initialisation Shell	DI	handoff_ shell	Analytics Hand-Off Shell
20	12	FSDF master data hand-off	Optional	2	epilogue_ shell	Analytics epilogue Shell	DI	handoff_ shell	Analytics Hand-Off Shell
21	13	FSDF EOD data hand-off	Optional	1	handoff_ shell	Analytics Hand-Off Shell	DI		
22	13	FSDF EOD data hand-off	Optional	2	epilogue_ shell	Analytics epilogue Shell	DI	handoff_ shell	Analytics Hand-Off Shell

Sr. No.	Process Category	Category Description	Category Significance	Shell Execution Sequence	Shell	Shell Description	Module Code	Required Shells	Required Shell Description
23	14	FSDF Txn data Hand-off	Optional	1	handoff_ shell	Analytics Hand-Off Shell	DI		
24	14	FSDF Txn data Hand-off	Optional	2	epilogue_ shell	Analytics epilogue Shell	DI	handoff_ shell	Analytics Hand-Off Shell
25	15	Analytics Batch Data Hand-off	Optional	1	handoff_ initialise	Analytics Hand-Off Initialisation Shell	DI		
26	15	Analytics Batch Data Hand-off	Optional	2	handoff_ shell	Analytics Hand-Off Shell	DI	handoff_ initialise	Analytics Hand-Off Initialisation Shell
27	15	Analytics Batch Data Hand-off	Optional	3	epilogue_ shell	Analytics epilogue Shell	DI	handoff_ shell	Analytics Hand-Off Shell

10 Information Lifecycle Management (ILM)

This chapter describes the configuration, installation, and policy setup of Information Lifecycle Management (ILM).

Information Lifecycle Management is a set of techniques and technologies available from Oracle that assist in managing the lifecycle of data to support business needs and minimize storage costs. OBPM drives ILM at the Oracle database level using database options and features to manage and move data as it evolves during its lifetime.

10.1 Configuration

The following values for the duration of data retention need to be determined. These values are used to drive ILM configuration.

Table 10–1 Values for ILM Configuration

Pattern Name	Partition Range Type	Data Retention in Active Tier	Data Retention in Less Active Tier	Data Retention in Historical Tier	Purge After
Lifecycle_ Pattern_1	MONTH	2 month	N/A	2 year	2 year
Lifecycle_ Pattern_2	MONTH	6 month	Will be provided by business	N/A	Will be provided by business
Lifecycle_ Pattern_3	YEAR	N/A	N/A	N/A	Will be provided by business
Lifecycle_ Pattern_4	YEAR	1 year	Will be provided by business		N/A
Lifecycle_ Pattern_5	YEAR	1 year	N/A	Will be provided by business	
Lifecycle_ Pattern_6	MONTH	1 month			1 month

10.2 Installation

This section explains the process of ILM installation.

10.2.1 Prepare Scripts

Operator needs to create partition creation script and ADO policy creation script manually based on data provided in ILM_Config.xlsx for each ILM qualified table and attached lifecycle pattern.

Parameters required for populating partition creation script are as follows:

- Table Name (OBPM Tables Worksheet)
- ILM Column (OBPM Tables Worksheet)
- Partition Range Type (Lifecycle Pattern Worksheet)

Parameters required for populating ADO policy creation script are as follows:

- Table Name (OBPM Tables Worksheet)
- Lifecycle Definition (OBPM Tables Worksheet)
- Data retention in different tier (Lifecycle Pattern Worksheet)
- Purging time (Lifecycle Pattern Worksheet)

The following sections describe the steps to be performed during the OBPM database creation.

10.2.2 Create Tablespace

Separate tablespaces need to be created for the following tiers:

- Active tier
- Less Active tier
- Historical tier

The following command is to be used for creation of the above tiers:

```
CREATE TABLESPACE <tablespace_name> datafile <datafile_name> SIZE <allocated_
size> SEGMENT SPACE management auto extent management local autoallocate;
```

For example:

```
CREATE TABLESPACE less_active_data datafile '/oracleE2POC/data01/s2poc/less_
active_data01.dbf' SIZE 10m SEGMENT SPACE management auto extent management
local autoallocate;
```

10.2.3 Create Partition Script

Partitioning script can be generated through partition script creation utility. For ILM qualified tables, the tables should always be partitioned based on range. Partition script can be generated based on Day, Month and Year. The following parameters need to be provided to the utility:

- Table Name
- ILM Column Name
- Schema Name (decided by DBA)
- Partition Interval (Default 1)
- Partition Type (DAY, MONTH and YEAR)
- Directory where partition script will be created (decided by DBA)

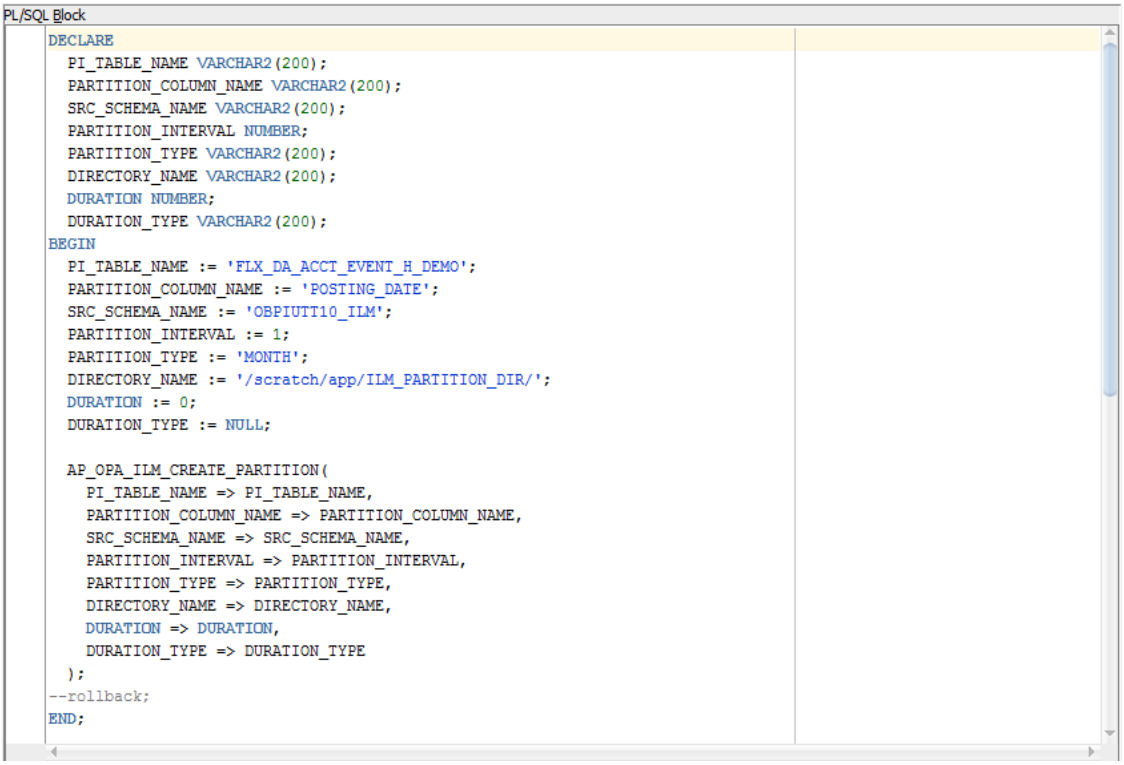
This utility can be run as follows:

1. Connect to OBPM Database.
2. Run the following SQL statement:

```
DECLARE
PI_TABLE_NAME VARCHAR2(200);
```

```
PARTITION_COLUMN_NAME VARCHAR2(200);
SRC_SCHEMA_NAME VARCHAR2(200);
PARTITION_INTERVAL NUMBER;
PARTITION_TYPE VARCHAR2(200);
DIRECTORY_NAME VARCHAR2(200);
DURATION NUMBER;
DURATION_TYPE VARCHAR2(200);
BEGIN
PI_TABLE_NAME := <ILM qualified table name>;
PARTITION_COLUMN_NAME := <ILM column name>;
SRC_SCHEMA_NAME := <Source schema name>;
PARTITION_INTERVAL := <Duration>;
PARTITION_TYPE :=< Partition type as DAY,MONTH or YEAR>;
DIRECTORY_NAME := <Location where partition script will be
created>;
DURATION := 0;
DURATION_TYPE := NULL;

AP_OPA_ILM_CREATE_PARTITION(
PI_TABLE_NAME => PI_TABLE_NAME,
PARTITION_COLUMN_NAME => PARTITION_COLUMN_NAME,
SRC_SCHEMA_NAME => SRC_SCHEMA_NAME,
PARTITION_INTERVAL => PARTITION_INTERVAL,
PARTITION_TYPE => PARTITION_TYPE,
DIRECTORY_NAME => DIRECTORY_NAME,
DURATION => DURATION,
DURATION_TYPE => DURATION_TYPE
);
--rollback;
END
```

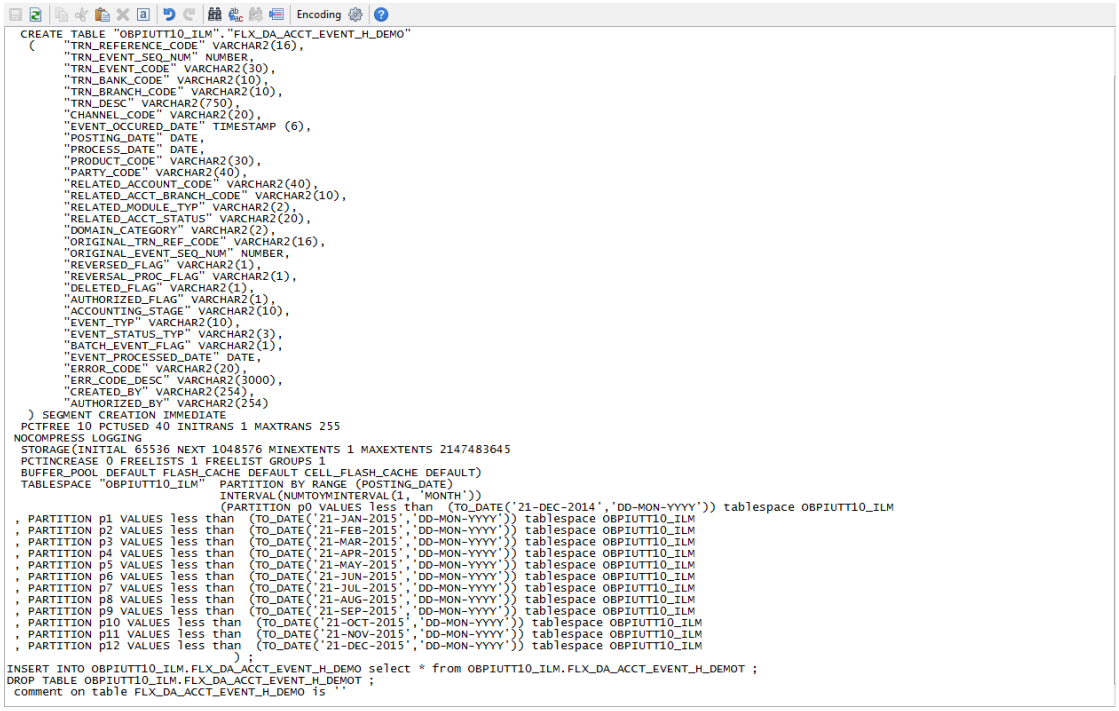
Figure 10–1 Partition Script - SQL Statement



```
PL/SQL Block
DECLARE
  PI_TABLE_NAME VARCHAR2(200);
  PARTITION_COLUMN_NAME VARCHAR2(200);
  SRC_SCHEMA_NAME VARCHAR2(200);
  PARTITION_INTERVAL NUMBER;
  PARTITION_TYPE VARCHAR2(200);
  DIRECTORY_NAME VARCHAR2(200);
  DURATION NUMBER;
  DURATION_TYPE VARCHAR2(200);
BEGIN
  PI_TABLE_NAME := 'FLX_DA_ACCT_EVENT_H_DEMO';
  PARTITION_COLUMN_NAME := 'POSTING_DATE';
  SRC_SCHEMA_NAME := 'OBPIUTT10_ILM';
  PARTITION_INTERVAL := 1;
  PARTITION_TYPE := 'MONTH';
  DIRECTORY_NAME := '/scratch/app/ILM_PARTITION_DIR/';
  DURATION := 0;
  DURATION_TYPE := NULL;

  AP_OPA_ILM_CREATE_PARTITION(
    PI_TABLE_NAME => PI_TABLE_NAME,
    PARTITION_COLUMN_NAME => PARTITION_COLUMN_NAME,
    SRC_SCHEMA_NAME => SRC_SCHEMA_NAME,
    PARTITION_INTERVAL => PARTITION_INTERVAL,
    PARTITION_TYPE => PARTITION_TYPE,
    DIRECTORY_NAME => DIRECTORY_NAME,
    DURATION => DURATION,
    DURATION_TYPE => DURATION_TYPE
  );
--rollback;
END;
```

3. After execution, the utility table creation script appears as shown in the below figure.

Figure 10–2 Utility Table Creation Script



```
CREATE TABLE "OBPIUTT10_ILM"."FLX_DA_ACCT_EVENT_H_DEMO"
  (     "TRN_REFERENCE_CODE" VARCHAR2(16),
        "TRN_EVENT_SEQ_NUM" NUMBER,
        "TRN_EVENT_CODE" VARCHAR2(30),
        "TRN_BANK_CODE" VARCHAR2(10),
        "TRN_BRANCH_CODE" VARCHAR2(10),
        "TRN_DESC" VARCHAR2(750),
        "CHANNEL_CODE" VARCHAR2(20),
        "EVENT_OCCURED_DATE" TIMESTAMP (6),
        "POSTING_DATE" DATE,
        "PROCESS_DATE" DATE,
        "PRODUCT_CODE" VARCHAR2(30),
        "PARTY_CODE" VARCHAR2(40),
        "RELATED_ACCOUNT_CODE" VARCHAR2(40),
        "RELATED_ACCT_BRANCH_CODE" VARCHAR2(10),
        "RELATED_MODULE_TYP" VARCHAR2(2),
        "RELATED_ACCT_STATUS" VARCHAR2(20),
        "DOMAIN_CATEGORY" VARCHAR2(2),
        "ORIGINAL_TRN_REF_CODE" VARCHAR2(16),
        "ORIGINAL_EVENT_SEQ_NUM" NUMBER,
        "REVERSED_FLAG" VARCHAR2(1),
        "REVERSAL_PROC_FLAG" VARCHAR2(1),
        "DELETED_FLAG" VARCHAR2(1),
        "AUTHORIZED_FLAG" VARCHAR2(1),
        "ACCOUNTING_STAGE" VARCHAR2(10),
        "EVENT_TYP" VARCHAR2(10),
        "EVENT_STATUS_TYP" VARCHAR2(3),
        "BATCH_EVENT_FLAG" VARCHAR2(1),
        "EVENT_PROCESSED_DATE" DATE,
        "ERROR_CODE" VARCHAR2(20),
        "ERR_CODE_DESC" VARCHAR2(3000),
        "CREATED_BY" VARCHAR2(254),
        "AUTHORIZED_BY" VARCHAR2(254)
  ) SEGMENT CREATION IMMEDIATE
 PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255
 NOCOMPRESS LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
  PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
  BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
  TABLESPACE "OBPIUTT10_ILM"  PARTITION BY RANGE (POSTING_DATE)
                              INTERVAL(NUMTOYMINTERVAL(1, 'MONTH'))
                              (PARTITION p0 VALUES less than  (TO_DATE('21-DEC-2014','DD-MON-YYYY')) tablespace OBPIUTT10_ILM
 , PARTITION p1 VALUES less than  (TO_DATE('21-JAN-2015','DD-MON-YYYY')) tablespace OBPIUTT10_ILM
 , PARTITION p2 VALUES less than  (TO_DATE('21-FEB-2015','DD-MON-YYYY')) tablespace OBPIUTT10_ILM
 , PARTITION p3 VALUES less than  (TO_DATE('21-MAR-2015','DD-MON-YYYY')) tablespace OBPIUTT10_ILM
 , PARTITION p4 VALUES less than  (TO_DATE('21-APR-2015','DD-MON-YYYY')) tablespace OBPIUTT10_ILM
 , PARTITION p5 VALUES less than  (TO_DATE('21-MAY-2015','DD-MON-YYYY')) tablespace OBPIUTT10_ILM
 , PARTITION p6 VALUES less than  (TO_DATE('21-JUN-2015','DD-MON-YYYY')) tablespace OBPIUTT10_ILM
 , PARTITION p7 VALUES less than  (TO_DATE('21-JUL-2015','DD-MON-YYYY')) tablespace OBPIUTT10_ILM
 , PARTITION p8 VALUES less than  (TO_DATE('21-AUG-2015','DD-MON-YYYY')) tablespace OBPIUTT10_ILM
 , PARTITION p9 VALUES less than  (TO_DATE('21-SEP-2015','DD-MON-YYYY')) tablespace OBPIUTT10_ILM
 , PARTITION p10 VALUES less than  (TO_DATE('21-OCT-2015','DD-MON-YYYY')) tablespace OBPIUTT10_ILM
 , PARTITION p11 VALUES less than  (TO_DATE('21-NOV-2015','DD-MON-YYYY')) tablespace OBPIUTT10_ILM
 , PARTITION p12 VALUES less than  (TO_DATE('21-DEC-2015','DD-MON-YYYY')) tablespace OBPIUTT10_ILM
                              ) ;
INSERT INTO OBPIUTT10_ILM.FLX_DA_ACCT_EVENT_H_DEMO select * from OBPIUTT10_ILM.FLX_DA_ACCT_EVENT_H_DEMOT ;
DROP TABLE OBPIUTT10_ILM.FLX_DA_ACCT_EVENT_H_DEMOT ;
 comment on table FLX_DA_ACCT_EVENT_H_DEMO is ''
```

10.2.4 Run Partition Script

The steps to run the partition script are as follows:

1. Download the newly created partition script from specified directory.
2. Verify created partition script before running.
3. Execute the script on OBPM database as follows:
    a. Connect to OBPM Database.
    b. Run partition creation script:

    @ <Tablename>par.sql

    For example:

    ```
    @ /scratch/app/ILM_PARTITION_DIR/FLX_DA_ACCT_EVENT_H_DEMOpar.sql
    ```

10.2.5 Create and Register ADO Policies based on Lifecycle Pattern

Automatic Data Optimization (ADO) is used to create policies and automate actions based on those policies, for implementing the ILM strategy. The data is moved across storage tiers. The following script needs to be executed to create the ADO policies:

1. Connect to OBPM Database.
2. Run ADO policy creation script:

    ```
    @ <Tablename>ado.sql
    ```

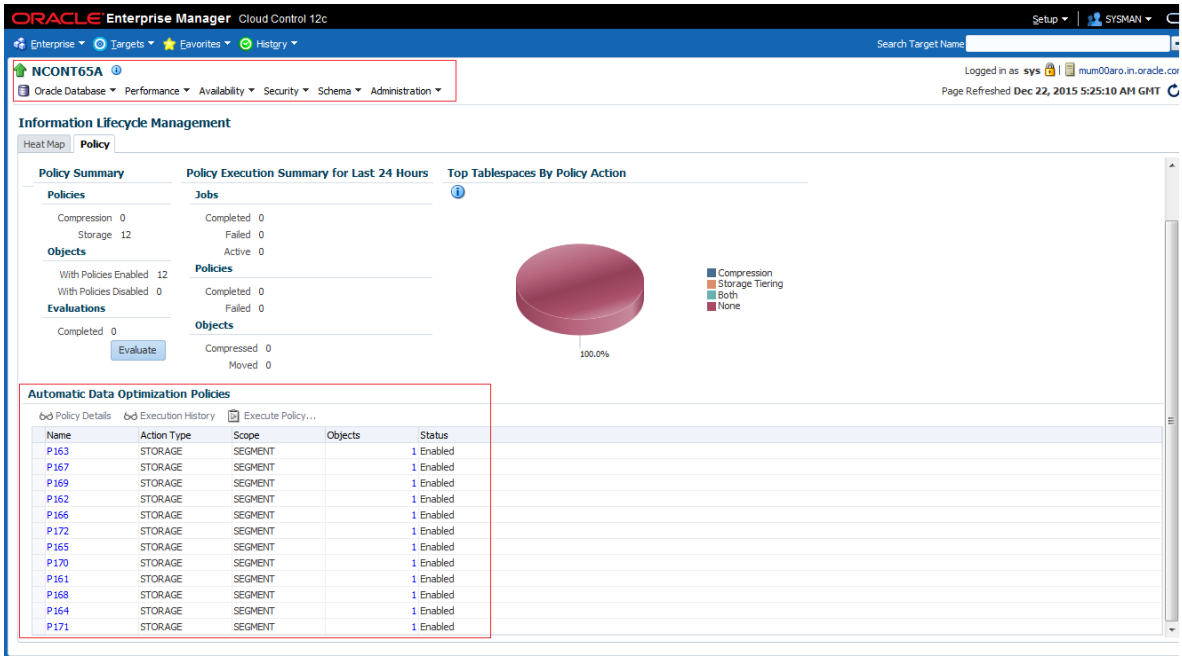
For example:

```
@ /scratch/app/ILM_ADO_DIR/FLX_DA_ACCT_EVENT_H_DEMOado.sql
```

10.2.6 Verify Registered ADO Policies

The created ADO policies can be verified through Oracle Enterprise Manager.

Figure 10–3 Verify ADO Policies



10.3 Policy Execution

ADO policies are required to be scheduled to execute automatically by configuring the database maintenance period. This can be determined during the implementation phase.

These ADO policies can be additionally executed manually with the following command:

```
declare
v_executionid number;
begin
dbms_ilm.execute_ilm (ilm_scope=>dbms_ilm.scope_schema,
execution_mode=>dbms_ilm.ilm_execution_offline,
task_id=>v_executionid);
end;
/
```

11 Transparent Data Encryption (TDE)

This chapter describes the configuration, installation, and policy setup of Transparent Data Encryption (TDE).

Transparent Data Encryption is a technology used to encrypt database files. This feature enables you to protect sensitive data in database columns stored in operating system files by encrypting it. Then, to prevent unauthorized decryption, it stores encryption keys in a security module external to the database.

11.1 Configuration

The following is the classification of information related to OBPM. This information is used to drives TDE configuration.

Table 11–1 TDE Configuration

Classification	Details	Access and Distribution	Action
Public	This information is not sensitive, and there is no value with it remaining confidential to Bank.	No restrictions	No Encryption
Confidential Internal	It is important that this information remains confidential to Bank.	May be accessed by and distributed to all support person. Distribution to third parties must be authorized by the information owner and requires that an appropriate confidential disclosure agreement be in place.	No Encryption
Confidential Restricted	It is very important that this information remains confidential to Bank and that access within bank is restricted on a need-to-know basis.	Internal access/distribution must be on a business need-to-know basis. Not authorized for information unless the information is encrypted using Oracle-approved encryption.	Need to set encryption rule during TDE
Confidential Highly Restricted	It is essential that this information remains confidential to Bank and that access within bank is restricted on a need-to-know basis.	Internal access/distribution must be very limited and is on a stringent business need-to-know basis. Not authorized for information unless the information is encrypted using Oracle-approved encryption.	Need to set encryption rule during TDE

All tables in OBPM are classified based on above classification and columns of those tables are marked based on sensitivity.

11.2 Installation

This section explains the installation process.

11.2.1 Prepare Scripts to Encrypt Sensitive Data

Database administrator needs to create alter script to encrypt sensitive data. The utility tool (obpencryption.sh) is used to create this alter script for TDE. To run the tool, the following prerequisites are required.

Prerequisites

- Create a folder "obpencryption" where user wants to run the tool.
- Upload Sensitive_Data_List.xlsx, obp-encryption-script-gen.jar, obpencryption.sh, DB_RESOURCEBUNDLE.properties. These files are available in maskingencryption.zip. The maskingencryption.zip is part of host.zip available in installer.
- Update database details in DB_RESOURCEBUNDLE.properties file before running the script.
- Update value "encryptLocation" variable with obp encryption path in obpencryption.sh at line 6.

  For example: `encryptLocation="/scratch/app/product/obpencryption"`

Run Encryption Tool

- Create update scripts for all the tables containing sensitive data. Run obpencryption.sh with TDE and ENCRYPT.

  For example: `/obpencryption.sh TDE ENCRYPT`

11.2.2 Create TDE Keystore

Perform these steps to create keystore which is required for encryption and decryption. Perform the following steps.

- Create keystore location with mkdir -p <location>.

  For example: `mkdir -p /scratch/app/admin/TDE/encryption_keystore/`

- Log in to database with sysdba.

  For example: `sqlplus / as sysdba`

- Run the following sql instruction:
  - ADMINISTER KEY MANAGEMENT CREATE KEYSTORE '{Keystore loaction}' IDENTIFIED BY {Password}

    For example: `SQL>ADMINISTER KEY MANAGEMENT CREATE KEYSTORE '/scratch/app/admin/TDE/encryption_keystore/' IDENTIFIED BY myPassword`
  - ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY welcome1 CONTAINER=ALL;

    For example: `SQL>ADMINISTER KEY MANAGEMENT SET KEYSTORE OPEN IDENTIFIED BY welcome1 CONTAINER=ALL;`
  - ADMINISTER KEY MANAGEMENT CREATE KEY using tag 'KEY5' IDENTIFIED BY welcome1 WITH BACKUP CONTAINER =all;

For example: `SQL>ADMINISTER KEY MANAGEMENT CREATE KEY using tag 'KEY5' IDENTIFIED BY welcome1 WITH BACKUP CONTAINER =all;`

- ADMINISTER KEY MANAGEMENT SET KEY using tag 'KEY5' IDENTIFIED BY welcome1 WITH BACKUP CONTAINER=ALL

  For example: `SQL>ADMINISTER KEY MANAGEMENT SET KEY using tag 'KEY5' IDENTIFIED BY welcome1 WITH BACKUP CONTAINER=ALL;`

- Check the encryption keys generated.

  For example: `SQL> SELECT con_id, key_id FROM v$encryption_keys;`

- Check the wallet status.

  For example: `SQL> SELECT * FROM v$encryption_wallet;`

11.2.3 Edit sqlnet.ora file

Perform this step to enter the TDE wallet location.

- Take a backup of sqlnet.ora file before update for TDE.
- Add entries of sqlnet.ora file as follows:

  ENCRYPTION_WALLET_LOCATION =

  (SOURCE =(METHOD = FILE)(METHOD_DATA =

  (DIRECTORY = {Keystore location})

  For example:
  ```
  ENCRYPTION_WALLET_LOCATION =
  (SOURCE =(METHOD = FILE)(METHOD_DATA =
  (DIRECTORY = /scratch/app/admin/TDE/encryption_keystore/)
  ```

11.2.4 Run Created Alter Script

- Get TDE_Encryption.sql script from obpencryption/generatedScript/tde.
- Log in to database.
- Run TDE_Encryption.sql.

12 Masking Customer Private Data

This chapter describes the configuration, installation, and policy setup to mask customer private data categories as sensitive or Personally Identifiable Information (PII).

12.1 Configuration

The following is the classification of information related to OBPM. This information is used to drive TDE configuration.

Table 12–1 TDE Configuration

Classification	Details	Access and Distribution	Action
Public	This information is not sensitive, and there is no value with it remaining confidential to Bank.	No restrictions	No Encryption
Confidential Internal	It is important that this information remains confidential to Bank.	May be accessed by and distributed to all support persons. Distribution to third parties must be authorized by the information owner and requires that an appropriate confidential disclosure agreement is in place.	No Encryption
Confidential Restricted	It is very important that this information remains confidential to Bank and that access within bank is restricted on a need-to-know basis.	Internal access/distribution must be on a business need-to-know basis. Not authorized for information unless the information is encrypted using Oracle-approved encryption.	Need to set encryption rule during masking Tables containing this type of data will be accessed through view for RO user. Synonym needs to be created for the tables and views containing this type of data for RO and ERO user.
Confidential Highly Restricted	It is essential that this information remain confidential to Bank and that access within bank is restricted on a need-to-know basis.	Internal access/distribution must be very limited and is on a stringent business need-to-know basis. Not authorized for information unless the information is encrypted using Oracle-approved encryption.	Need to set encryption rule during masking. Tables containing this type of data will be accessed through view for RO user. Synonym needs to be created for the tables and views containing this type of data for RO and ERO user.

All tables in OBPM are classified based on above classification and columns of these tables are marked based on sensitivity.

12.2 Installation

This section explains the installation process.

12.2.1 Prepare Scripts to Encrypt Sensitive Data

Database administrator needs to create the following script for masking sensitive data.

- View creation script of the tables containing sensitive data and mask them for RO (Read only) user.
- Synonym creation script of created view of the containing sensitive data for RO (Read only) user.
- Synonym creation script of tables containing sensitive data for ERO (E Read only) user.

The utility tool (obpencryption.sh) is used to create above script. To run the tool, the following prerequisites are required.

Prerequisites

- Create a folder "obpencryption" where user wants to run the tool.
- Upload Sensitive_Data_List.xlsx, obp-encryption-script-gen.jar, obpencryption.sh, DB_RESOURCEBUNDLE.properties. These files are available in maskingencryption.zip. The maskingencryption.zip is part of host.zip available in installer.
- Update database details in DB_RESOURCEBUNDLE.properties file before running the script.
- Update value "encryptLocation" variable with obp encryption path in obpencryption.sh at line 6.

  For example: `encryptLocation="/scratch/app/product/obpencryption"`

Run Encryption Tool for View Creation script and mask data

- Create view creation scripts for all the tables containing sensitive data after mask. Run obpencryption.sh with MASK and VIEWCREATE as parameter.

  For example: `/obpencryption.sh MASK VIEWCREATE`

Run Encryption Tool for Synonym Creation script for RO user

- Create synonym creation scripts for all the created containing sensitive data. Run obpencryption.sh with MASK and SYNONYMRO as parameter.

  For example: `/obpencryption.sh MASK SYNONYMRO`

Run Encryption Tool for Synonym Creation script for ERO user

- Create synonym creation scripts for all the tables containing sensitive data. Run obpencryption.sh with MASK and SYNONYMEERO as parameter.

  For example: `/obpencryption.sh MASK SYNONYMERO`

12.2.2 Create Schema for RO and ERO User

To create schema for RO and ERO user, execute the following steps.

- Create Read-Only (RO) and E Read-Only (ERO) user for accessing masked data from view and table.
- Grant for proper access.

12.2.3 Execute Created Scripts through Encryption Tool

Run all created scripts through the encryption tool for the following task.

- Mask sensitive data for RO user.
- Create view for tables contain sensitive data.
- Create synonym to access the view.
- Create synonym to access the table for ERO user.

To do the above tasks, perform the following steps.

- Get all view creation scripts from obpencryption /generatedScript/masking/viewforRO location and run after logging in to database.
- Get synonym creation script (MaskingSynonymForRO.sql) for RO user from obpencryption/generatedScript/masking/synonymForRO and run after logging in to database.
- Get synonym creation script (MaskingSynonymForERO.sql) for ERO user from obpencryption/generatedScript/masking/ synonymForERO and run after logging in to database.

13 Configure ODI for Inbound Document Upload

This chapter provides the steps to configure ODI for Inbound Document Upload.

For document upload ODI execution, complete the following configurations:

1. Configuring the Input directory:
    a. For example, if input directory is /scratch/odi/InboundDocument/Upload/lendingZone/
    b. Update the configuration in the table using the following SQL:

        update FLX_FW_ODI_SUB_INTERFACE_TYPE set IN_FILE_ PATH=/scratch/odi/InboundDocument/Upload/lendingZone/ ' where SUB_INTERFACE_ ID='9551';

    c. Note that the SUB_INTERFACE_ID='9551' should not be changed.
2. Configure the Schema directory:
    a. Provide the directory where all the schemas are present.

        Framework configuration:

        update FLX_FW_ODI_SUB_INTERFACE_TYPE set SCHEMA_FILE_PATH = '/scratch/odi/InboundDocument/Upload/schema/' where SUB_INTERFACE_ID ='7002';

    b. Copy all the schema for ODI mediapack zip from directory schema to the new directory which is configured for framework.
    c. Provide schema file for Document Upload.

        update FLX_FW_ODI_SUB_INTERFACE_TYPE set schema_file_ path='D:\work\odi\inDocUpload\InboundDocument\Upload\schema\scan_images_ request.0.1.XSD' where SUB_INTERFACE_ID='9551';

3. Configure Temporary directory:

    update FLX_FW_ODI_SUB_INTERFACE_TYPE set temp_file_path= '/scratch/odi/InboundDocument/Upload /lendingZonetmp/' where SUB_INTERFACE_ID='9551';

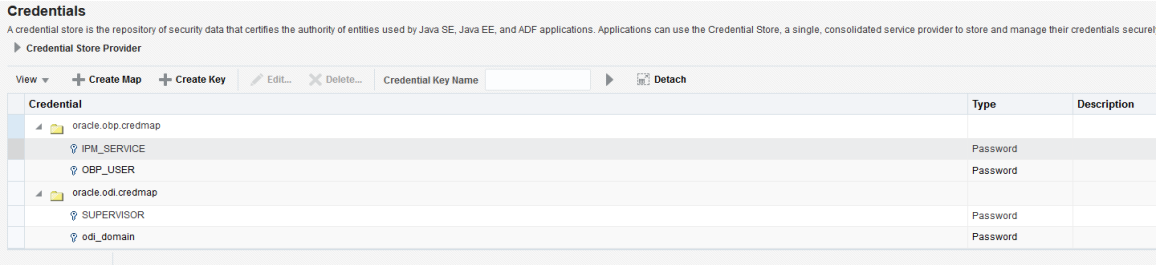
4. Configure Archive directory:

    update FLX_FW_ODI_SUB_INTERFACE_TYPE set archive_file_ path=/scratch/odi/InboundDocument/Upload/archieve/' where SUB_INTERFACE_ID='9551';

5. Create users in connector: Create two credential maps:
    - oracle.obp.credmap: This has two keys.
        - IPM_SERVICE: It has the username and password of IPM. It is used to upload the documents to IPM.
        - OBP_USER: It is required to make web service call to OBP.

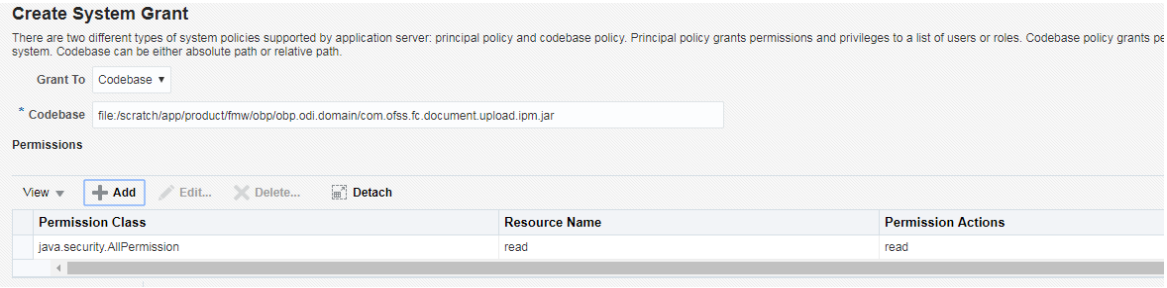
- oracle.odi.credmap
  - SUPERVISOR: It has supervisor username and password.
  - odi_domain: It has domain username and password.

Figure 13–1 Credentials



6. Provide permission to the java project for fetching the user credentials. Provide read permission to Java binary com.ofss.fc.document.upload.ipm.jar from EM.

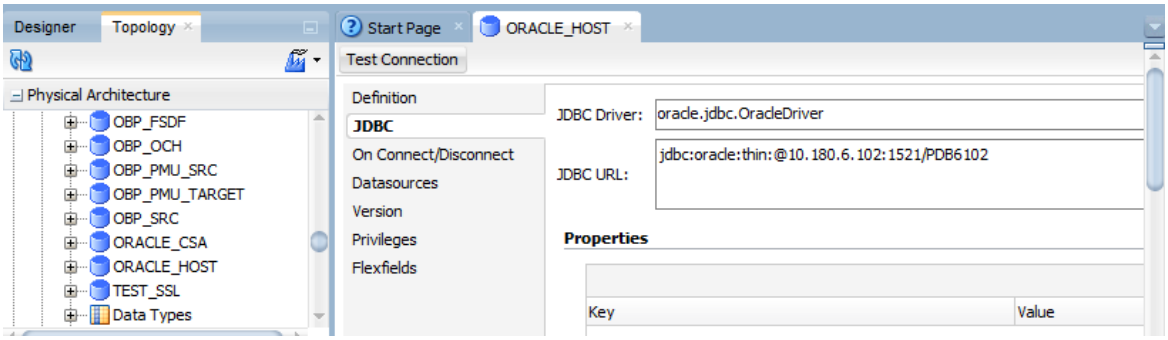
Figure 13–2 Example of premissions



7. Configuring IPM URL:
   a. update FLX_FW_ODI_SUB_INTERFACE_TYPE set GEFU_IN_FILE_PATH='http://${IPM-HOSTNAME}:${IPM-PORT}/imaging/ws' where SUB_INTERFACE_ID='9551';
   b. Replace ${IPM-HOSTNAME} with IPM Hostname or IP address.
   c. Replace ${IPM-PORT} with IPM Server port number.
8. Configure config/properties/OutboundWebserviceConfig.properties to provide OBP Host web service configuration.
   a. Replace ${OBP-HOST-IP} with OBP Host IP address or hostname.
   b. Replace ${OBP-HOST-PORT} with OBP Host managed server port.

9. Provide FJ Connection Details in ODI Topology data server ORACLE_HOST.

Figure 13–3 Connection details



14 Additional Recommendations

This chapter provides specific recommendations to be considered for implementation:

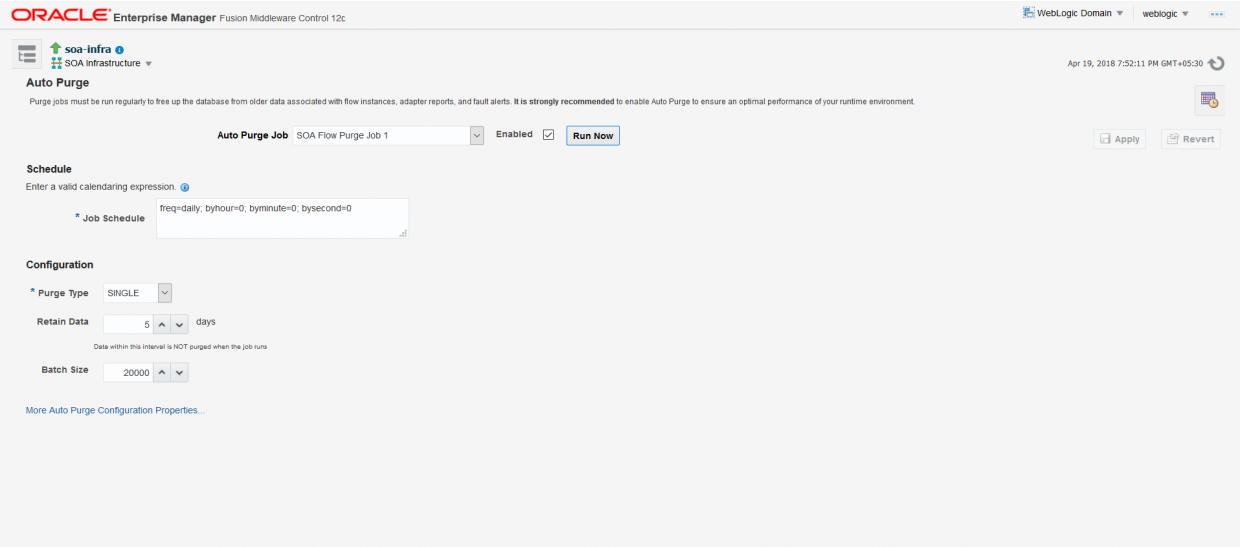
14.1 SOA Related

This section mentions certain recommendations for setting up the properties on SOA server.

14.1.1 Enable Auto Purge Job

- Oracle SOA Suite team strongly recommends periodic purging of composite instances. Purge instances as soon as they are available for purge.
- SOA suite 12c comes up with default purge job enabled with retention period of 7 days.
- It is recommended to keep this default job enabled in the production.

Figure 14–1 Auto Purge

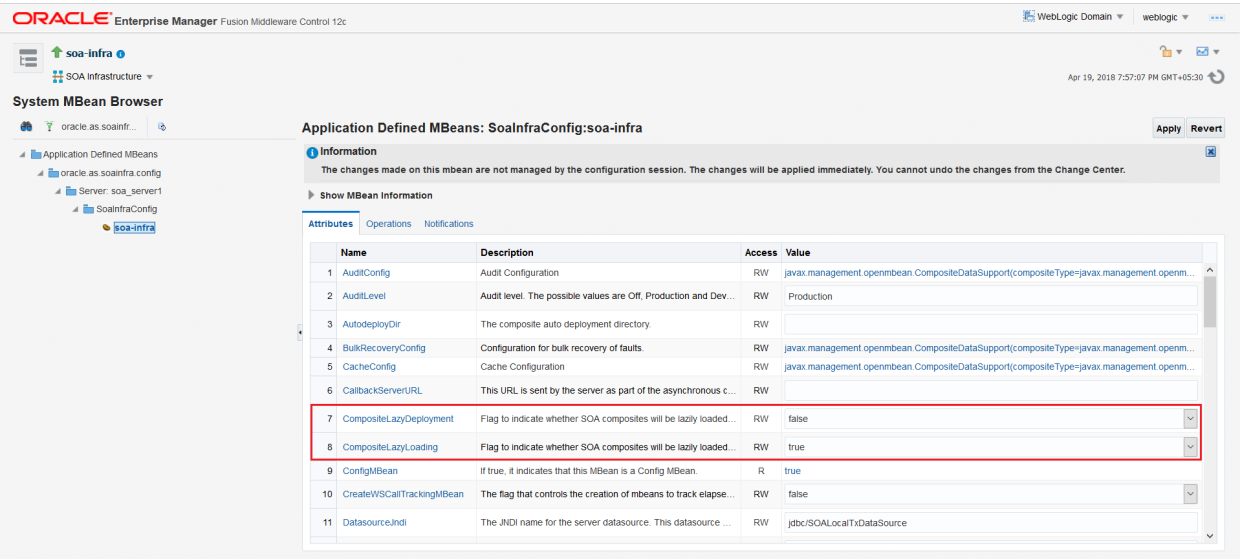


14.1.2 Enable Lazy Loading

- 12c supports lazy loading of composites on deployment as well as server startup.
- **CompositeLazyDeployment**: Loads the composites lazily on deployment.
  - More useful for non-production environments where there is frequent deployment of all the composites.
- **CompositeLazyLoading**: Loads the composites lazily on server start up.
- It is recommended to keep the default settings unchanged, that is CompositeLazyDeployment as false and CompositeLazyLoading as true in production environment.

- This will reduce the SOA server startup time.
- To confirm the settings, in SOA EM console, go to soa-infra > SOA Administration > Common Properties. Click More SOA Infra Advanced Configuration Properties link.

Figure 14–2 Lazy Loading Settings



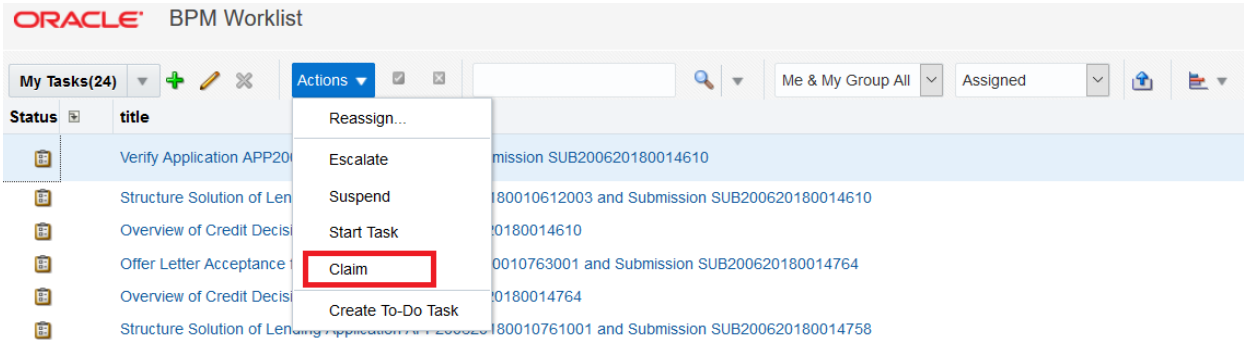
14.2 BPM Worklist Related

This section mentions certain recommendations on usage of BPM worklist.

14.2.1 Disable Claim Action from Task Details Page

- Always claim the task using worklist action menu.
- Claim option is disabled by default inside task details page.
- It can be enabled (configurable), if needed (but not recommended).
- Disabling this option helps to avoid loading of task details page twice, that is before claiming the task and after claiming the task.

Figure 14–3 Claim Action



14.2.2 Always Open Human Task Details in External Window

- BPM worklist supports two options to display the task details.
  - **Same Window**: Human task details are opened in the same browser window just below the worklist grid.
  - **External Window**: Human task details are opened in the new browser window (as a child popup).
- It is strongly recommended to make use of **External Window** option.
  - Human task details can be seen in full-screen mode.
  - Avoids unnecessary loading the task details page if user is browsing through the list of human tasks in worklist grid.
- This option can be enabled from worklist administration page.

Figure 14–4 Enable External Window option