Oracle® Retail Allocation

Operations Guide

Release 15.0

E65743-01

December 2015

Oracle® Retail Allocation Operations Guide, Release 15.0

E65743-01

Primary Author: Kris Lange

Value-Added Reseller (VAR) Language

Oracle Retail VAR Applications

This documentation is in preproduction status and is intended for demonstration and preliminary use only. It may not be specific to the hardware on which you are using the software. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to this documentation and will not be responsible for any loss, costs, or damages incurred due to the use of this documentation.

Contents

6 Oracle Business Intelligence Enterprise Edition Integration in Retail Application

7 Troubleshooting

8 Functional Design

10 Oracle Retail Extract, Transform, and Load (RETL) Batch Processing

11 Java Batch Process

14 Implementing Functional Security

Send Us Your Comments

Oracle® Retail Allocation Operations Guide, Release 15.0

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document.

Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

> **Note:** Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the Online Documentation available on the Oracle Technology Network Web site. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: retail-doc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our Web site at http://www.oracle.com.

Preface

The *Oracle Retail Allocation Operations Guide* provides critical information about the processing and operating details of the application, including the following:

- System configuration settings
- Technical architecture
- Functional integration dataflow across the enterprise
- Batch processing

Audience

This guide is for:

- Systems administration and operations personnel
- Systems analysts
- Integrators and implementation personnel
- Business analysts who need information about product processes and interfaces

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle Retail Allocation documentation set:

- *Oracle Retail Allocation Release Notes*
- *Oracle Retail Allocation Installation Guide*

- *Oracle Retail Allocation User Guide and Online Help*
- *Oracle Retail Allocation Data Model*
- *Oracle Retail Merchandising Implementation Guide*
- *Oracle Retail Merchandising Security Guide*
- *Oracle Retail Merchandising Batch Schedule*
- *Oracle Retail Operational Insights User Guide*

Customer Support

To contact Oracle Customer Support, access My Oracle Support at the following URL:

`https://support.oracle.com`

When contacting Customer Support, please provide the following:

- Product version and program/module name
- Functional and technical description of the problem (include business impact)
- Detailed step-by-step instructions to re-create
- Exact error message received
- Screen shots of each step you take

Review Patch Documentation

When you install the application for the first time, you install either a base release (for example, 15.0) or a later patch release (for example, 15.1). If you are installing the base release, additional patch, and bundled hot fix releases, read the documentation for all releases that have occurred since the base release before you begin installation. Documentation for patch and bundled hot fix releases can contain critical information related to the base release, as well as information about code changes since the base release.

Improved Process for Oracle Retail Documentation Corrections

To more quickly address critical corrections to Oracle Retail documentation content, Oracle Retail documentation may be republished whenever a critical correction is needed. For critical corrections, the republication of an Oracle Retail document may at times not be attached to a numbered software release; instead, the Oracle Retail document will simply be replaced on the Oracle Technology Network Web site, or, in the case of Data Models, to the applicable My Oracle Support Documentation container where they reside.

This process will prevent delays in making critical corrections available to customers. For the customer, it means that before you begin installation, you must verify that you have the most recent version of the Oracle Retail documentation set. Oracle Retail documentation is available on the Oracle Technology Network at the following URL:

`http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html`

An updated version of the applicable Oracle Retail document is indicated by Oracle part number, as well as print date (month and year). An updated version uses the same part number, with a higher-numbered suffix. For example, part number E123456-02 is an updated version of a document with part number E123456-01.

If a more recent version of a document is available, that version supersedes all previous versions.

Oracle Retail Documentation on the Oracle Technology Network

Documentation is packaged with each Oracle Retail product release. Oracle Retail product documentation is also available on the following Web site:

`http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html`

(Data Model documents are not available through Oracle Technology Network. These documents are packaged with released code, or you can obtain them through My Oracle Support.)

Documentation should be available on this Web site within a month after a product release.

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 Introduction

Welcome to the Oracle Retail Allocation Operations Guide. The guide is designed so that you can view and understand key system-administered functions, the flow of data into and out of the application, and the application's behind-the-scenes processing of data.

Allocation Overview

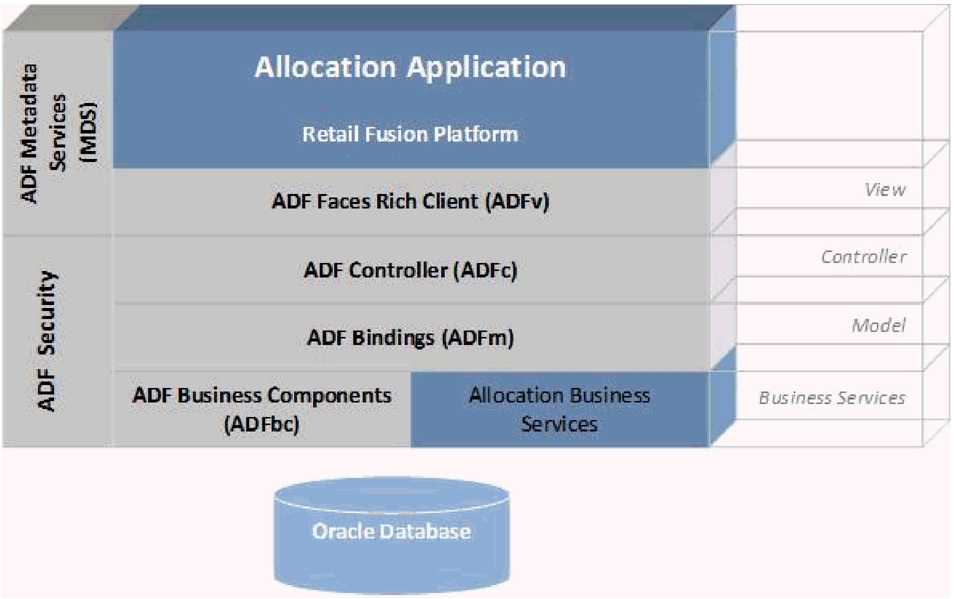
A retailer that acquires Oracle Retail Allocation gains the ability to achieve more accurate allocations on a stable product. Having the right product in the right stores or warehouses allows for service levels to be raised, sales to be increased, and inventory costs to be lowered. By accurately determining which locations should get which product, retailers can meet their turnover goals and increase profitability.

The Oracle Retail Allocation retailer benefits from the following capabilities:

- Built on ADF Technology stack, it allows the ability to quickly add UI based on ready to use design patterns, metadata driven tools and visual tools. Debugging can be performed more rapidly; maintenance and alteration costs are kept low using the metadata driven application development.
- The application's interface takes advantage of the Java Database Connectivity (JDBC), ADF's built-in transaction management, along with connections to data sources handled in WebLogic server which minimizes the interface points that need to be maintained.
- The application's robust algorithm executes rapidly, and the call to the calculation engine has been ported over from C++ library to a Java Library, thus minimizing the overhead/issues related to maintaining codebase consisting of two disparate languages.
- For retailers with other Oracle Retail products, integration with the Oracle Retail product suite means that item, purchase order, supplier, sales, and other data are accessed directly from the RMS tables, with no need for batch modules. Purchase order, item, location, and allocation information are passed from RMS to a warehouse management system, such as the Oracle Retail Warehouse Management System (RWMS).
- Access control to the system is better managed by using Fusion Security Architecture.
- The application allows for retailers to adjust to changing trends in the market by facilitating real time allocations.
- Oracle Retail Allocation accounts for flexible supply chain paths to support importing and domestic inventory supply.

The following diagram illustrates the Allocation n-tier architecture:

Figure 1–1 Oracle Retail Allocation's n-tier Architecture



RMS owns virtually all of the information that Oracle Retail Allocation needs to operate, and the information that Oracle Retail Allocation provides is of primary interest/use for RMS. As a result, Oracle Retail Allocation has limited interaction with other Oracle Retail Merchandising Operations Management applications. For Oracle Retail Merchandising Operations Management applications that Oracle Retail Allocation does interact with, it is managed through direct reads from Oracle Retail Merchandising Operations Management application tables, direct calls to Oracle Retail Merchandising Operations Management packages, and Oracle Retail Allocation packages based on Oracle Retail Merchandising Operations Management application tables.

2

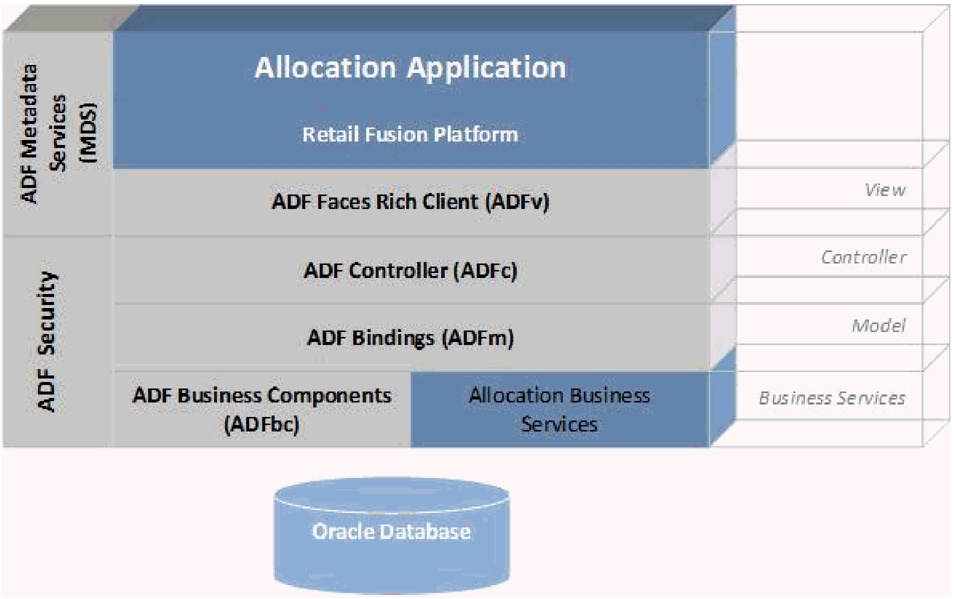
Technical Architecture

This chapter describes the overall software architecture for Oracle Retail Allocation. The chapter provides a high-level discussion of the general structure of the system, including the various layers of Java code.

Overview

Retail Applications are based on the Oracle Application Development Framework (ADF). The following diagram shows the key components that make up the architecture of Retail Applications.

Figure 2–1 Oracle Retail Allocation's n-tier Architecture



Oracle Application Development Framework (ADF)

Oracle Application Development Framework (ADF) supports organizations in building cutting-edge rich enterprise business applications that can be customized and personalized in all dimensions. Customizations are global changes, visible to all users that are performed by an administrator. Personalizations are user-made changes that are only visible to the person making the change.

ADF is based on the Java Enterprise Edition platform.

Model-View-Controller (MVC) Architectural Pattern

Applications built using ADF follow a Model-View-Controller (MVC) architectural pattern. The goal of the MVC pattern is to clearly separate the application's functionality into a set of cooperating components.

ADF provides a set of components that realize the goals of each part of MVC pattern.

- Model is realized by the ADF Bindings Layer.
- Controller is realized by the ADF Controller Layer.
- View is realized by the ADF Faces Layer.
- ADF Business components and other backend components that sit below the Model layer are called Business Services.

Application Development Framework Security

The ADF security layer provides the following:

- Standards based (Oracle Platform Security Services (OPSS)) security framework with default roles and permissions.
- Tools to generate file-based identity store (for both Oracle Internet Directory and AD) based on the framework.
- Tools to migrate file-based security store in to database for QA and production environments.
- Reference implementation for clients to manage the security based on their business needs.
- OPSS-based batch security framework (Retail Fusion Platform).
- Tools/documentation to implement centralized logout in Single Sign-On (SSO) (Oracle Access Management (OAM)) environments.

Application Development Framework View (ADFv)

The View layer provides the user interface to the application. The view layer uses HTML, rich Java components or XML and its variations to render the user interface. JSF based tag libraries are used to display the User Interface (UI).

Application Development Framework Controller (ADFc)

The ADF Controller layer controls the application's flow. Web based applications are composed of multiple web pages with dynamic content. The controller layer manages the flow between these pages. Different models can be used when building this later. The most prominent architecture for Java-based web applications relies on a servlet that acts as the controller. The Apache Jakarta Struts controller, an open source framework controller, is the de facto standard for Java-based web systems. Oracle ADF uses the Struts controller to manage the flow of web applications.

Application Development Framework Business Components (ADFbc)

The business service layer manages the interaction with a data persistence layer. It provides services as data persistence, object/relational mapping, transaction management, and business logic execution.

Business Components easily map the database object and extend it with business logic, validation, and so on.

The idea behind Business Components is to abstract the data layer from the view layer. This is a key concept in the MVC pattern. Business Components expose the interface to the view layer by using an application module that contains View Object. Those view objects contain a specific usage of the data layer.

ADF Business Components implements the business service through the following set of cooperating components:

- Entity object – An entity object represents a row in a database table and simplifies modifying its data by handling all data manipulation language (DML) operations for you. It can encapsulate business logic for the row to ensure that your business rules are consistently enforced. You associate an entity object with others to reflect relationships in the underlying database schema to create a layer of business domain objects to reuse in multiple applications.
- View object – A view object represents a SQL query. You use the full power of the familiar SQL language to join, filter, sort, and aggregate data into exactly the shape required by the end-user task. This includes the ability to link a view object with others to create master-detail hierarchies of any complexity. When end users modify data in the user interface, view objects collaborate with entity objects to consistently validate and save the changes.
- Application module – An application module is the transactional component that UI clients use to work with application data. It defines an updatable data model and top-level procedures and functions (called service methods) related to a logical unit of work related to an end-user task.

Application Development Framework Model (ADFm)

This is the component that acts as the connector between the view and business logic layers.

The Model layer connects the Business Services to the objects that use them in the other layers. Oracle ADF provides a Model layer implementation that sits on top of Business Services, providing a single interface that can be used to access any type of Business Services.

Developers get the same development experience when binding any type of Business Service layer implementation to the view and Controller layers. The Model layer in Oracle ADF served as the basis for JSR 227, A Standard Data binding & Data Access Facility for J2EE.

Oracle Metadata Services (MDS)

The ability of an application to adapt to changes is a necessity that needs to be considered in the application design and that should drive the selection of the development platform and architecture. Flexible business applications must be able to adapt to organizational changes, different end user preferences and changes in the supported business are required.

MDS is the customization and personalization framework integral to Oracle Fusion Middleware and a key differentiator of the Oracle development platform. MDS provides a repository for storing metadata for applications, such as customizations and persisted personalization files and configurations.

Retail applications allow the following through MDS:

- Personalization of saved searches through MDS.
- Implicit personalization of few ADF UI attributes.

Retail Fusion Platform

The Retail Fusion Platform (commonly referred to as Platform) is a collection of common, reusable software components that serve as foundation for building Oracle Retail's next generation ADF-based applications. The Platform imposes standards and patterns along with a consistent look and feel for Oracle Retail's ADF applications.

Java Back-end Components

Allocation services in legacy code are maintained to reduce the impact of the Fusion re-architectural changes. These Service classes are later called through ADF Business Component's Application module.

Data Access Patterns

Database interaction between the middle tier and the database is done using the industry standard Java Database Connectivity Protocol (JDBC). JDBC facilitates the communication between a Java application and a relational database.

Database Access Using ADFbc

JDBC is ingrained within Oracle ADF Business Components as the primary mechanism for its interaction between the middle tier and the database. SQL is realized within ADF business components to facilitate create, read, update and delete (CRUD) actions.

Connection Pooling

When the application 'disconnects' a connection, the connection is saved into a pool instead of being actually disconnected. A standard connection pooling technique, this saved connection, enables Retail applications to reuse the existing connection from a pool. In other words, the application does not have to complete the connection process for each subsequent connection.

Data Storage

The Oracle database realizes the database tier in a Retail application’s architecture. It is the application's storage platform, containing the physical data (user and system) used throughout the application. The database tier is only intended to handle the storage and retrieval of information and is not involved in the manipulation or in the delivery of the data. This tier responds to queries; it does not initiate them.

Accessing Merchandising System Data in Real Time

The data that the Retail application utilizes is located in both the application-specific and the merchandising system (RMS, for example) tables. Because Retail applications share the same schema as the merchandising system (RMS, for example), the application is able to interact with the merchandising system's data directly, in real time.

3

Back-end System Administration and Configuration

This chapter is intended for administrators who support and monitor the running system.

The content in this chapter is not procedural, but is meant to provide descriptive overviews of the key system parameters.

Managing Asynchronous Processes

This section covers the following topics:

- Overview of Asynchronous Processes in Retail Applications
- Configuring JMS Resources for Asynchronous Processing
- Monitoring Asynchronous Tasks Via the RAF_ASYNC_TASK Table
- Purging the Asynchronous Tasks Table

Overview of Asynchronous Processes in Retail Applications

Applications may need to execute operations that can take significant time to complete.

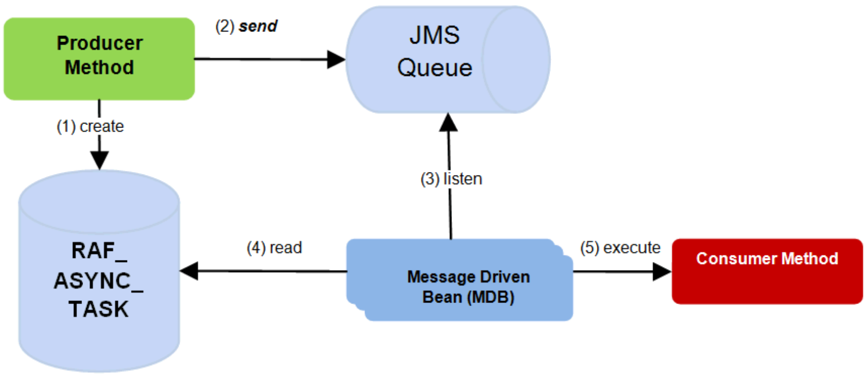
Executing these operations in the application's own thread will cause the application to block other operations until those long-running operations finish.

In contrast, being able to assign the long-running operations to background threads allows the foreground thread to remain active and service further user requests. This is also described as launching those operations asynchronously.

For example, in the Allocation application, when an allocation request is submitted for approval, the system needs to perform a comprehensive validation to ensure that the allocation is valid. The validation should be performed asynchronously. You can proceed with performing other operations within the system. Once the background validation is complete, you are notified.

Retail Applications achieve asynchronous processing using Java Messaging Service (JMS). When a user, for instance, through the application UI screen, submits a business entity for processing asynchronously, the application goes through a process depicted in the diagram below:

Figure 3–1 Asynchronous Process



1. The Retail application's producer process executes to collect the user request for asynchronous processing and **stores that information into a database table** called RAF_ASYNC_TASK. This table serves as a log of asynchronous processing requests as well as storage for any kind of context information in order to complete the processing. For example, the unique identifier of the business object such as an allocation being submitted for asynchronous calculation.
2. The producer method **creates a queue message and sends it to a JMS Queue**.
3. One or more instances of **Message Driven Beans (MDB) are configured to listen to the JMS Queue**.
4. When messages are detected on the queue, the **MDBs are dispatched** to execute a task asynchronously. The **MDB reads the context information** about the task from the TASK table.
5. The **MDB executes the required processing** for the task.

Configuring JMS Resources for Asynchronous Processing

The following resources in the WebLogic Administrator Console application should be configured in order to allow asynchronous processing to function in a Retail application:

Table 3–1 WebLogic Administrator Console Resources

WebLogic Server Resource	Required Names/References	Configuration Notes
JMS Server	Any name can be used.	None. Defaults provided by WebLogic acceptable.
JMS Module	Any name can be used.	None. Defaults provided by WebLogic acceptable.
JMS Queue	Name: calcQueue JNDI: oracle.retail.apps.alc.async.AllocCalcQueue	▪ Create the queue under the JMS Module. ▪ Associate a sub-deployment for the queue. If none exists, create it. The sub-deployment must be the same as the JMS Queue Connection Factory's. ▪ Redelivery Limit must be set to zero (0). ▪ Error Destination must be set to NONE.

Table 3–1 (Cont.) WebLogic Administrator Console Resources

WebLogic Server Resource	Required Names/References	Configuration Notes
JMS Queue Connection Factory	Name: allocCalcQueueCF JNDI: oracle.retail.apps.alc.async.AllocCalcQueueCF	■ Create under the same JMS Module as the queue. ■ Associate a sub-deployment for the queue connection factory. If none exists, create it. The sub-deployment must be the same as the JMS Queue's. ■ Maximum Messages per Session must be set to one (1). ■ XA Connection Factory Enabled must be true.
Work Manager	Name: AllocWorkManager	■ Create Minimum and Maximum Thread Constraints and assign to the work manager. - The recommended Minimum Thread Constraint is 8. - The Maximum Thread Constraint must be set. Typically, it should be set to the maximum number of anticipated application users on the system at any given time. ■ Associate the work manager to the Retail Application asynchronous consumer MDB,AllocConsumerMDBBean, by specifying the work manager name in the MDB's Dispatch Policy field.

Refer to WebLogic documentation found in `http://www.oracle.com/technetwork/middleware/weblogic/documentation/index.html`.

for detailed information about how to configure the above resources.

Monitoring Asynchronous Tasks Via the RAF_ASYNC_TASK Table

The RAF_ASYNC_TASK table can be used to view error codes for failures, and to monitor for any failures requiring triage from production support team. As described in the section, Overview of Asynchronous Processes in Retail Applications, each asynchronous processing request is logged as a row in the RAF_ASYNC_TASKS table. As the task is sent to the queue, picked up and completed by the consuming message driven bean, the rows in this table are updated with status information.

Table 3–2 RAF_ASYNC_TASK Table Details

RAF_ASYNC_TASK Column	Description
ASYNC_TASK_ID	A unique identifier for each row in this table. Each asynchronous task is given a unique ASYNC_TASK_ID.
APPLICATION_CODE	A code representing the Retail Application that generated the asynchronous task.
TASK_DESC	A short description of the asynchronous task.

Table 3–2 (Cont.) RAF_ASYNC_TASK Table Details

RAF_ASYNC_ TASK Column	Description
TASK_CONTEXT	A string containing information needed to process the asynchronous task. Usually indicates the ID of the business entity to be processed (example: the ID of the allocation request to be approved)
STATUS	The status of the asynchronous task. ■ NEW - The task is ready to be asynchronously processed. ■ IN-PROGRESS - The task is currently being processed by the application. ■ SUCCESS - Processing for the task completed successful. ■ FAILED - Processing for the task completed with failures. Additional information about the failure can be found in the PROCESS_ERROR_TXT column and/or the server logs.
TASK_ COMMAND_ CLASS_NAME	A fully-qualified Java class name that contains the business logic for completing the task.
PUBLISH_ TIMESTAMP	The time and date when the asynchronous task was published. That is, when the status was changed to NEW.
PROCESS_START_ TIMESTAMP	The time and date when processing of the asynchronous task started. That is, when the status was changed to IN-PROGRESS.
PROCESS_END_ TIMESTAMP	The time and date when the processing of the asynchronous task ended with either a SUCCESS or FAILED status.
PROCESS_ ERROR_TXT	When the status is FAILED, this column would contain information about the failure. The server logs may need to be inspected for more detailed information about the cause of the processing failure. ■ Invalid size profile. <Item> -<Loc>: This error can arise if the typical size profile message <item> is your itemId with missing size profile and <loc> is the store number. ■ CALCULATE_WITH_NO_ITEM_SOURCE_AVAIL: This error can arise if there is no available quantity at the source. ■ RELEASE_DATE_BEFORE_CURRENT_DATE: This error can arise if the release date is in the past. ■ RELEASE_DATE_BEFORE_PO_NOT_BEFORE_DATE: This error can arise if the release date selected is before the PO_NOT_ BEFORE_DATE ■ ERRMSG_ITEMSOURCESERVICE_LOAD_ERROR: This error can arise when our itemsource list is 0. This is typically caused by ALC_ITEM_SOURCE being empty for that allocation. ■ ERRMSG_ALLOC_LOAD_NONEXISTANT_PO: This error can arise if you tried to load a PO that doesn't exist. Unlikely, but handled exception. ■ ERRMSG_ALLOC_LOAD_INVALID_PO: This error can arise if you tried to load a PO that is no longer valid. Unlikely, but handled exception. ■ LIST_LOCATION_NO_VALID_STORES: This occurs when ALC_ CALC_DESTINATION_TEMP is empty. Usually, this table is empty when you miss an exclusion.
CREATED_BY	Audit field pertaining to the logged in application user that requested the task.
CREATE_DATE	Audit field pertaining to when the row was created.

Table 3–2 (Cont.) RAF_ASYNC_TASK Table Details

RAF_ASYNC_ TASK Column	Description
LAST_UPDATED_ BY	Audit field pertaining to the user that last updated the row.
LAST_UPDATE_ DATE	Audit field pertaining to the last date/time when the row was updated.

Purging the Asynchronous Tasks Table

As previously mentioned, each asynchronous task is logged as a row in the RAF_ ASYNC_TASK table. Over time, this table can grow significantly which can degrade performance of the asynchronous task mechanism.

It is recommended that retailers periodically purge this table.

Retail applications provide a simple PL/SQL function to purge contents of the RAF_ ASYNC_TASK table based on retention time period (in days).

```
declare
 retval number(10);
begin
 -- remove rows older than 5 days
 retval := raf_async_task_pkg.delete_async_task(5);
end;
/
```

Managing the Notifications Feature

This section covers the following topics:

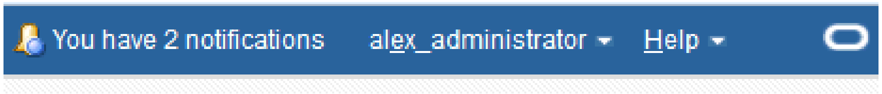
- Overview of Notifications in Retail Applications
- Purging the Notifications Table

Overview of Notifications in Retail Applications

Retail applications provide the ability to notify authenticated users in the application when business events occur. An example of a business event is when a background asynchronous task such as an approval of an allocation or a validation of a price change has completed. Typically, you are expected to take action on these notifications.

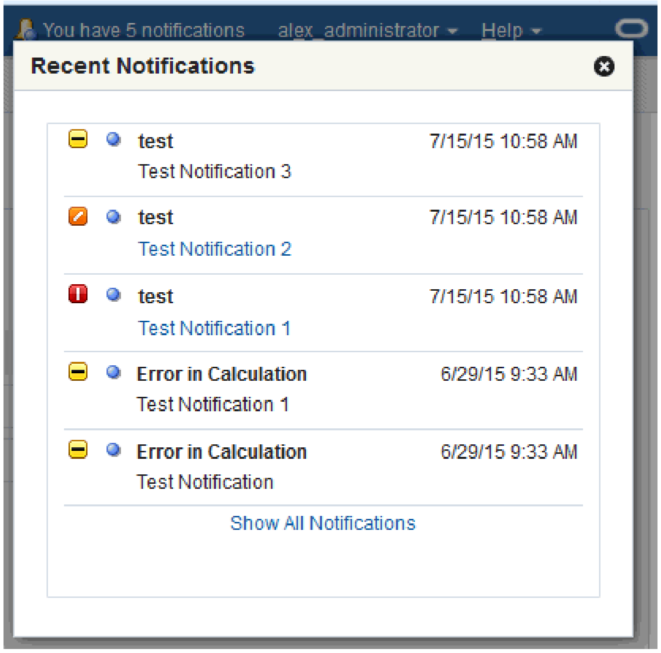
When a notification is generated for the user, the notification icon on the Retail Application's global area appears along with a count of the number of pending notifications for the user.

Figure 3–2 Pending Notifications



A popup is presented to the user when he/she clicks on the notification icon. The popup will show five latest notifications. Clicking on "Show All Notifications" will show all notifications in a new tab.

Figure 3–3   Pending Action for Notifications



Recent Notifications

test 7/15/15 10:58 AM
Test Notification 3

test 7/15/15 10:58 AM
Test Notification 2

test 7/15/15 10:58 AM
Test Notification 1

Error in Calculation 6/29/15 9:33 AM
Test Notification 1

Error in Calculation 6/29/15 9:33 AM
Test Notification

Show All Notifications

Clicking the link for each notification opens the specific UI flow that will allow the user to address any pending action for the notification.

Purging the Notifications Table

Notifications are represented as rows in the RAF_NOTIFICATION table. Over a period of time, depending on how notifications are being generated by the application, the size of this table can grow continually, potentially degrading performance of the notifications feature.

It is recommended that retailers purge this table periodically. You can do this via the Purge feature available in the Retail Application Administration Console (RAAC). For more details on the Manage Notifications feature in RAAC, refer to the Oracle Retail Application Administration Console chapter of the *Oracle Retail Merchandising Implementation Guide*.

You could also setup a batch job that runs periodically, that invokes the pl-sql as shown below:

```
declare
 retval number(10);
begin
 -- remove Notifications from the Allocation application that have exceeded   . --
their retention duration.
 retval := raf_notification_task_pkg. DEL_NOTIF_PAST_RETENTION('ALC');
end;
```

Notification ReSTful Web Services

ReST Endpoints have been exposed by the Notifications framework to ease the integration concerns with disparate applications. Refer to section "Notification ReSTful Web Services" in the Allocation ReSTful Web Services Implementation chapter for details on service endpoints.

Managing Application Navigator

Retail Applications provide an ability to switch between applications using the Application Navigator facility. These applications are configured using the Manage Application Navigator screens on Retail Application Administration Console (RAAC). For more details on Application Navigator in RAAC, refer to the Oracle Retail Application Administration Console chapter of the *Oracle Retail Merchandising Implementation Guide*.

Managing Functional Security

This section discusses the functional security for Retail applications and the components used to implement it. Functional security is based on OPSS. For more information on OPSS, refer to the *Oracle Fusion Middleware Application Security Guide*.

This section covers the following topics:

- Introduction to Retail Roles
- Retail Role Hierarchy
- Default Security Reference Implementation
- Extending the Default Security Reference Implementation

Introduction to Retail Roles

Users are not assigned to permissions directly; rather access is assigned to roles. Roles group particular permissions required to accomplish a task; instead of assigning individual permissions, roles match users with the permissions required to complete their particular task.

There are two main types of roles, enterprise and application.

The Identity Store contains enterprise roles that are available across applications. These are created as groups in LDAP, making them available across applications.

Applicable Retail Applications security provides four types of roles: abstract, job, duty, and privilege.

Applicable Retail Applications record job, abstract roles as enterprise roles and duty, privilege roles as application roles.

Security Policy Stripe

Application roles are stored in the application-specific policy store. These roles are normally assigned by the system (based on user attributes), but can be provisioned to a user on request.

Abstract Roles

Abstract roles are associated with a user, irrespective of their job or job function. These are also roles that are not associated with a job or duty. These roles are normally assigned by the system (based on user attributes), but can be provisioned to a user on request.

Naming Convention: All the Retail Abstract role names end with' _ABSTRACT'

Example: APPLICATION_ADMIN_ABSTRACT

Job Roles

JJob roles are associated with the job of a user. A user with this job can have many job functions or job duties.

> **Note:** These roles are called Job roles as the role names closely map to the jobs commonly found in most organizations.

Naming Convention: All the Retail Job role names end with' _JOB'

Example: ALLOCATOR_JOB.

Duty Roles

Job duties are tasks one must do on a job. A person is hired into a job role. These are the responsibilities one has for a job.

Duty roles are roles that are associated with a specific duty or a logical grouping of tasks. Generally, the list of duties for a job is a good indicator of what duty roles should be defined.

Duty roles should:

- Read as a job description at a job posting site.
- Duties that we create should be self-contained and pluggable into any existing or new job or abstract role.

Naming Convention: All the Retail duty role names end with' _DUTY'

Example: ALC_ALLOC_POLICY_MAINTENANCE_MANAGEMENT_DUTY

Privilege Roles

Privilege is the logical collection of permissions. A privilege can be associated with any number of User Interface components. Privileges are expressed as application roles.

Naming Convention: All the Retail Privilege role names end with' _PRIV'

Example: ALC_ALLOC_SEARCH_PRIV

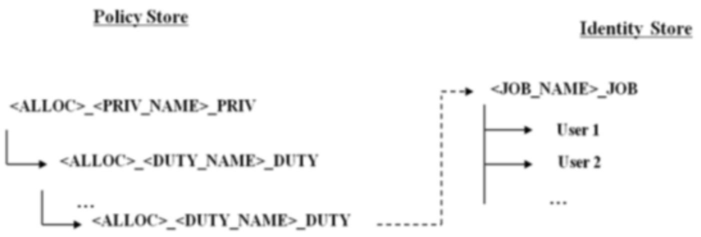
```
Privilege roles carry security grants.
Example:
<grant>
   <grantee>
      <principals>
            <principal>
<class>oracle.security.jps.service.policystore.
ApplicationRole</class>
               <name>ALC_ALLOC_SEARCH_PRIV</name>
            </principal>
       </principals>
     </grantee>
  <permissions>
   <permission>
 <class>oracle.adf.controller.security.TaskFlowPermission</class>
 <name>/oracle/retail/apps/alc/allocsummary/publicUi
/flow/AllocationSummaryFlow.xml#AllocationSummaryFlow</name>
    <actions>view</actions>
  </permission>
 </permissions>
```

```
</grant>
```

Retail Role Hierarchy

Retail role hierarchies are structured to reflect the Retail business process model.

Figure 3–4 Retail Role Hierarchy



Job roles inherit duty roles. For example, the Allocator Job role inherits the ALC_ALLOC_SYSTEM_OPTIONS_INQUIRY_DUTY roles.

```
<app-role>
  <name>ALC_ALLOC_SYSTEM_OPTIONS_INQUIRY_DUTY</name>
  <class>oracle.security.jps.service.policystore.ApplicationRole</class>
  <members>
   <member>
     <class>oracle.security.jps.internal.core.principals.
                JpsXmlEnterpriseRoleImpl</class>
     <name>ALLOCATOR_JOB</name>
  </member>
 </members>
</app-role>
```

Duty roles inherit Privilege roles. Duty roles can inherit one or more other Duty roles.

Example: ALC_ALLOC_SIZE_PROFILE_MANAGEMENT_DUTY inherits ALC_ALLOC_SIZE_PROFILE_INQUIRY_DUTY role.

```
<app-role>
  <name>ALC_ALLOC_SIZE_PROFILE_INQUIRY_DUTY</name>
  <class>oracle.security.jps.service.policystore.ApplicationRole</class>
  <members>
    <member>
     <class>oracle.security.jps.internal.core.principals.
                            JpsXmlEnterpriseRoleImpl</class>
     <name>BUYER_JOB</name>
    </member>
    <member>
     <class>oracle.security.jps.service.policystore.ApplicationRole</class>
     <name>ALC_ALLOC_SIZE_PROFILE_MANAGEMENT_DUTY</name>
    </member>
  </members>
</app-role>
```

Example: ALC_ALLOC_SIZE_PROFILE_INQUIRY_DUTY role inherits the ALC_ALLOC_SIZE_PROFILE_VIEW_PRIV role

```
<app-role>
```

```
 <name>ALC_ALLOC_SIZE_PROFILE_VIEW_PRIV</name>
 <class>oracle.security.jps.service.policystore.ApplicationRole</class>
 <members>
  <member>
   <class>oracle.security.jps.service.policystore.ApplicationRole</class>
   <name>ALC_ALLOC_SIZE_PROFILE_INQUIRY_DUTY</name>
  </member>
 </members>
</app-role>
```

Default Security Reference Implementation

Retail applications ship with default security reference implementations. The source of truth for default reference implementation is the jazn-data.xml file.

Privileges

For more information on privileges, see Default Security Reference Implementation in the Implementing Functional Security chapter.

Table 3–3 Privileges

Name	Description
Search Allocation Priv	A privilege for searching for allocations.

Duties

For more information on duties, see Default Security Reference Implementation in the Implementing Functional Security chapter.

Table 3–4 Duties

Name	Description	List of Privileges
Allocation Management Duty	A duty for managing allocations. This duty is an extension of the Allocation Inquiry Duty.	▪ All privileges found in the Allocation Inquiry Duty ▪ Maintain Allocation Privilege ▪ Delete Allocation Privilege

Role Mapping

For more information on role mapping, see Default Security Reference Implementation in the Implementing Functional Security chapter.

Table 3–5 Role Mapping

Role	Duty	Privileges
Allocation Management Duty	A duty for managing allocations. This duty is an extension of the Allocation Inquiry Duty.	▪ All privileges found in the Allocation Inquiry Duty ▪ Maintain Allocation Privilege ▪ Delete Allocation Privilege

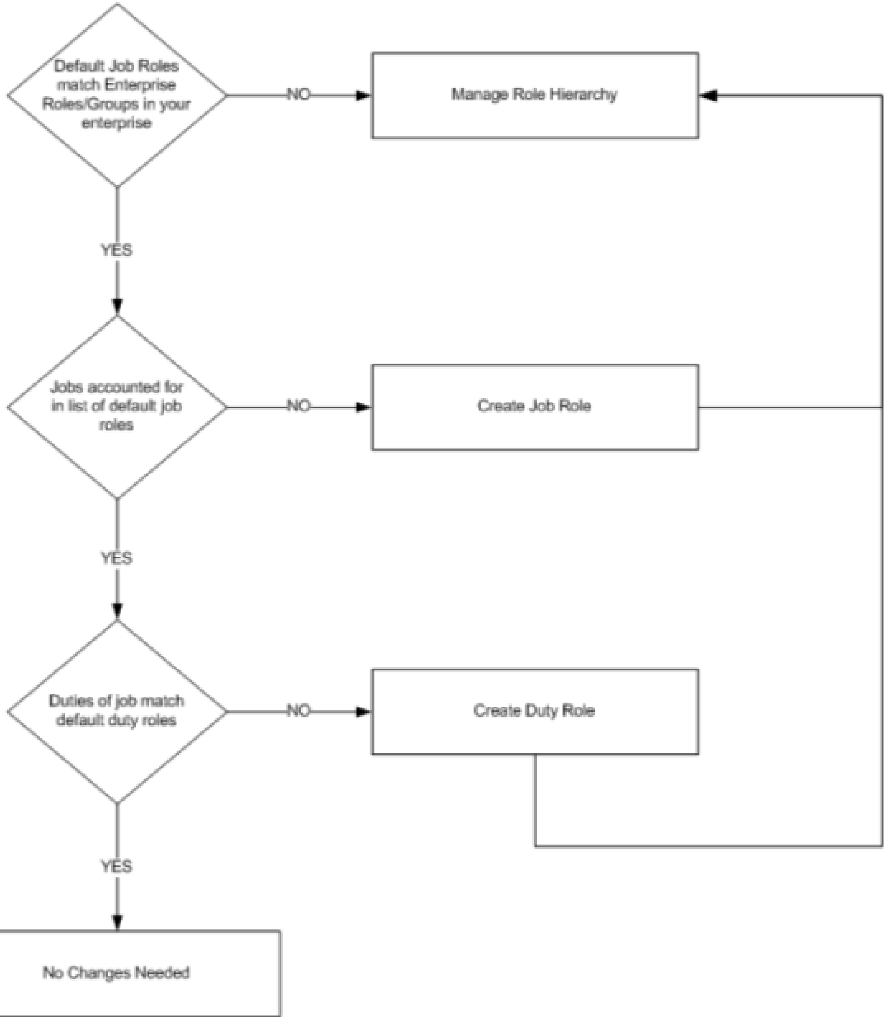
Extending the Default Security Reference Implementation

Note: Make sure that the policy store is loaded with the default security configuration. For more information, see the Post Installation steps in the *Oracle Retail Allocation Installation Guide*.

The common decisions made to match your enterprise to the default security reference implementation include the following:

- Do the default job roles match the equivalent job roles in your enterprise?
- Do the jobs in your enterprise exist in the security reference implementation?
- Do the duties performed by the jobs in your enterprise match the duties in the security reference implementation?

Figure 3–5 Decisions for Default Security Reference Implementation


Default Job Roles match Enterprise Roles/Groups in your enterprise
NO
Manage Role Hierarchy
YES
Jobs accounted for in list of default job roles
NO
Create Job Role
YES
Duties of job match default duty roles
NO
Create Duty Role
YES
No Changes Needed

Important: It is important when constructing a role hierarchy that circular dependencies are not introduced. The best practice is to leave the default security configuration in place and first incorporate your customized application roles in a test environment.

Managing Roles in Retail Application Administration Console

Retail applications provide a way in which retailers can modify the default roles to map to their security groups through the Retail Application Administration Console (RAAC).

RAAC is installed along with the Retail Application. Users with proper security privileges to access RAAC can launch RAAC by clicking on a link from the Retail Application's global menu.

For more information about using RAAC, refer to the Oracle Retail Application Administration Console chapter of the *Oracle Retail Merchandising Implementation Guide*.

Disabling Content

There are situations where administrators need to disable certain links or the default content such as Dashboards due to unavailability or other reasons. Retail Applications provide the flexibility to disable such content so that the application remains largely unaffected.

Safe Mode

Applications can choose to serve certain content such as dashboards to users upon launching the application. This is referred to as "Default Content". However sometimes this default content may cause delays in application launch after logging-in or worse it may render the application unusable.

To handle such scenarios Retail Applications provide a feature for Administrators called "Safe Mode" which. allows the user to log in without serving up any default content. Once this mode is turned on, no default content is shown to any user when the application is launched.

To turn on this mode the property "uishell.load.safe.mode" must be set to true in the RetailAppsViewController.properties file.

Disabling Links in the Sidebar

Administrators may occasionally need to disable content launchable from links in the sidebar navigation tree. Retail applications provide the ability to disable such links.

To disable a link the Administrator must first find the "id" of that link as specified in the SidebarNavigationModel.xml file. This value must then be provided to the property "uishell.sidebar.invalid.item.ids" within the RetailAppsViewController.properties file. To disable more than one link, pass in multiple ids separated by a comma.

Managing Oracle Metadata Services (MDS)

Retail Applications are built using ADF and one of the features within ADF is the Oracle Metadata Services (MDS) framework which provides a facility for retailers to customize the applications.

Refer to the document, *Oracle Fusion Middleware Fusion Developer's Guide* for Oracle Application Development Framework (https://docs.oracle.com/middleware/1213/adf/develop/adf-web-customizing-apps.htm#ADFFD2077), for more information about MDS.

Overview of Oracle Metadata Services

Oracle Metadata Services (MDS) is a key infrastructure component in Oracle Fusion Middleware. It is the layer through which metadata is loaded, saved, cached, stored, managed, and customized both by various middleware components and by the applications built on Fusion Middleware.

The use of MDS in ADF applications, for example, can allow applications to remember how users like to work, and therefore not require them to set up the application for every session. This may include, for example, saving of common searches and screen layouts for every user. This allows making use of the application easier and more intuitive for the users. MDS provides a foundation that can be leveraged by Oracle Application Development Framework (ADF) applications to provide such persistent personalization.

Oracle Metadata Services (MDS) makes use of metadata repositories or partitions. A Metadata repository or partition contains metadata for Oracle Fusion Middleware components. It can also contain metadata about the configuration of Oracle Fusion Middleware and metadata for applications. Oracle Metadata Services (MDS) stores the customizations in a metadata repository and retrieves them at runtime to merge the customizations with the base metadata to reveal the customized application.

A common problem when a patch is installed for a Retail Application is that certain screens would fail to load or UI elements fail to display data properly.

The cause of this issue is commonly attributed to user personalization on screen elements that are now removed in the patch.

For example, prior to patching the application, users may have saved search criterias on certain screens as a way to conveniently recall their desired search results whenever they use the application. Those saved search criterias are persisted by ADF in the MDS repository. If the patch involves the removal of one of the attributes used in the search criterias, applying the patch will cause the screens that have those search criterias fail to load.

The MDS repository is configured in the WebLogic server where the Retail Application is deployed. The repository is database-based and it is organized or subdivided into partitions. Retail Applications are deployed with their own partition within the server's MDS repository.

It is recommended to not delete the MDS partition during the upgrade of the Retail application, instead use the functions described in this document to resolve any issues related to MDS.

Using the System MBean Browser and the MDSAppRuntime MBean

For managing MDS Customizations in Retail Fusion Applications, use the Oracle Enterprise Manager to perform common metadata service tasks such as exporting, deleting, and importing of MDS Customizations. This can be done in the Oracle Enterprise Manager using the System MBean Browser and the MDSAppRuntime MBean.

Perform the following steps to access the MDSAppRuntime MBean.

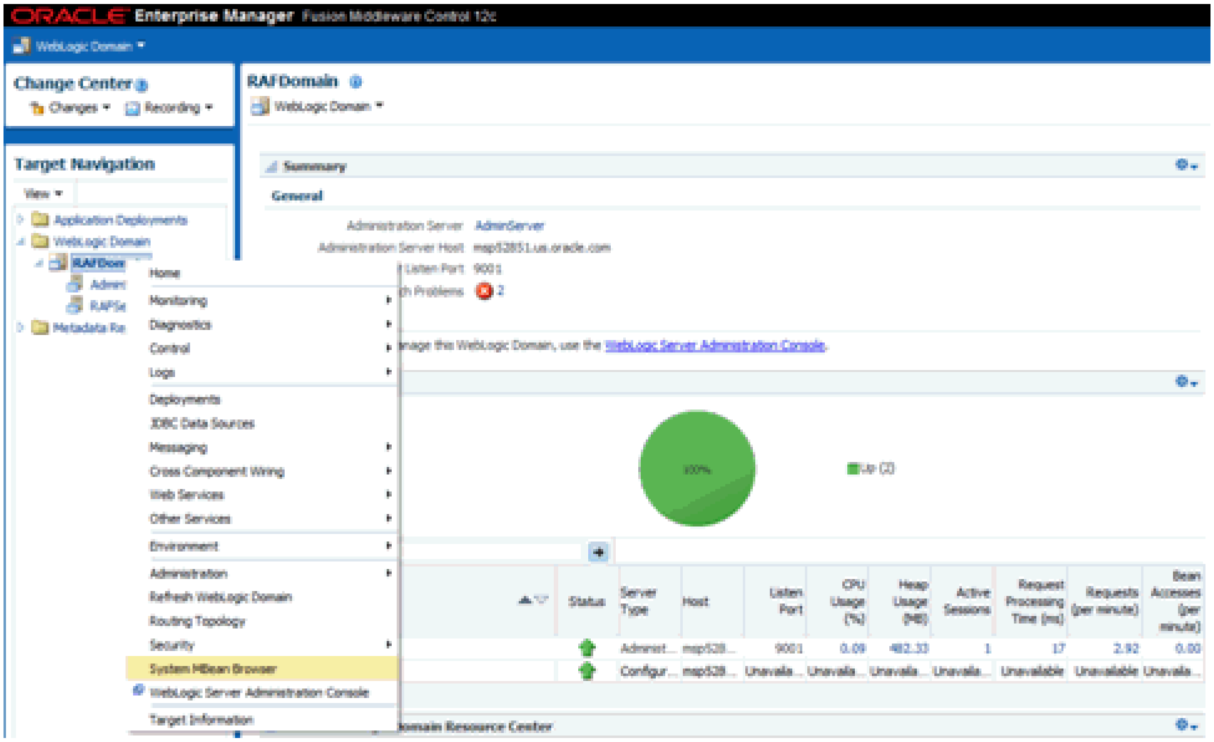
1. Login to the Oracle Enterprise Manager by navigating to the URL in the following format:

http://<host>:<port>/em/

2. From the Navigation menu, under WebLogic Domain, right-click on the application domain and navigate to System MBean Browser from the context menu.

   Choose the correct domain based on your installation. The figure below displays RAFDomain.

Figure 3–6 RAFDomain window



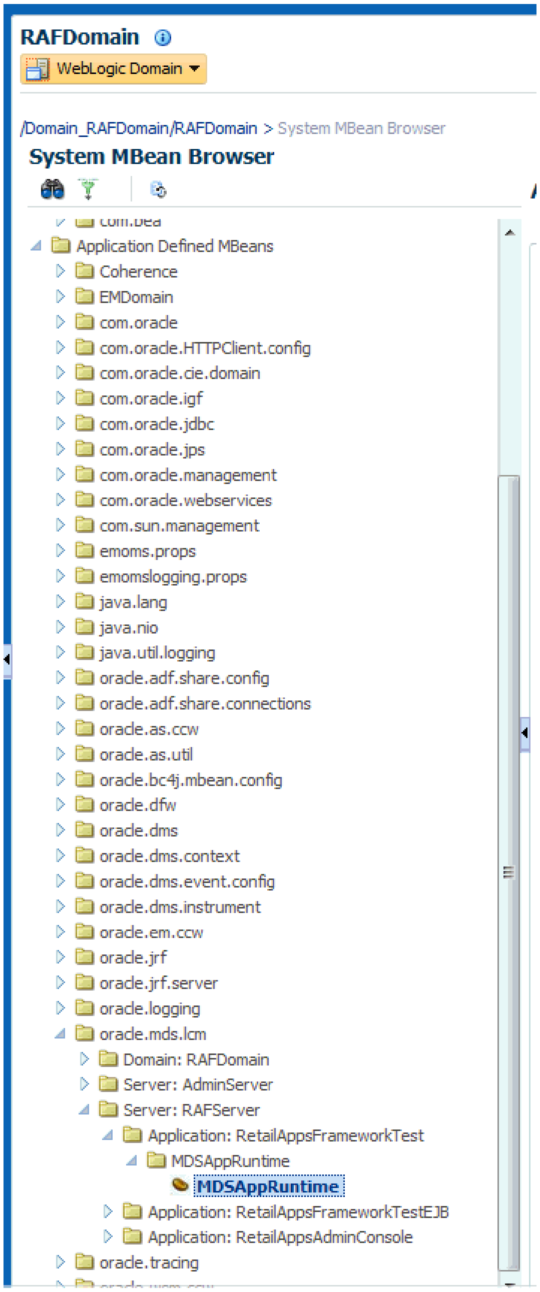
Status	Server Type	Host	Listen Port	CPU Usage (%)	Heap Usage (MB)	Active Sessions	Request Processing Time (ms)	Requests (per minute)	Bean Accesses (per minute)
⬆	Administ...	msp528...	9001	0.09	482.33	1	17	2.92	0.00
⬆	Configur...	msp528...	Unavaila...	Unavaila...	Unavaila...	Unavaila...	Unavailable	Unavailable	Unavaila...

3. Under Application Defined MBeans, locate MDSAppRuntime which can be found under the following folder structure:

```
> oracle.mds.lcm
  > Server: AppServer
    > Application: Application Name
      > MDSAppRuntime
```

   Choose the correct server and the Retail Application name based on your installation. The figure below displays RAFServer and RetailAppsFrameworkTest application.

Figure 3–7 RAFDomain System MBean Browser window



RAFDomain

WebLogic Domain

/Domain_RAFDomain/RAFDomain > System MBean Browser

System MBean Browser

- com.bea
- Application Defined MBeans
  - Coherence
  - EMDomain
  - com.oracle
  - com.oracle.HTTPClient.config
  - com.oracle.cie.domain
  - com.oracle.igf
  - com.oracle.jdbc
  - com.oracle.jps
  - com.oracle.management
  - com.oracle.webservices
  - com.sun.management
  - emoms.props
  - emomslogging.props
  - java.lang
  - java.nio
  - java.util.logging
  - oracle.adf.share.config
  - oracle.adf.share.connections
  - oracle.as.ccw
  - oracle.as.util
  - oracle.bc4j.mbean.config
  - oracle.dfw
  - oracle.dms
  - oracle.dms.context
  - oracle.dms.event.config
  - oracle.dms.instrument
  - oracle.em.ccw
  - oracle.jrf
  - oracle.jrf.server
  - oracle.logging
  - oracle.mds.lcm
    - Domain: RAFDomain
    - Server: AdminServer
    - Server: RAFServer
      - Application: RetailAppsFrameworkTest
        - MDSAppRuntime
          - MDSAppRuntime
      - Application: RetailAppsFrameworkTestEJB
      - Application: RetailAppsAdminConsole
  - oracle.tracing

4. Click MDSAppRuntime management bean (MBean).

5. Select the Operations tab to view the operations available for the MDSAppRuntime MBean.

   These are the operations which can be used in managing MDS Customizations. The exportMetadata, deleteMetadata, importMetadata, createMetadataLabel, listMetadataLabels, deleteMetadataLabel, and promoteMetadataLabel; will be briefly discussed in the following sections.

Exporting All Metadata Services Customizations

1. Navigate to the MDSAppRuntime management bean (MBean), as described in the section "Using the System MBean Browser and the MDSAppRuntime MBean".

2. Export metadata by selecting the exportMetadata operation available from the Operations tab.

3. For toLocation, provide a valid absolute path to a directory or archive in the file system to which the selected documents will be exported. This location must be accessible from the machine where the application is running. If it does not exist, a directory will be created except that when the name ends with ".jar", ".JAR", ".zip", or ".ZIP", an archive will be created. Exporting metadata to an existing archive will overwrite the existing file.

   Example: /tempDir/downloads/mdsExport/RetailAppsFrameworkTestMDS.zip

4. Click **Invoke** (located at the upper-right corner of the page) to proceed with the export operation.

5. Click **Return** to return to the list of operations.

Exporting Metadata Services Customization for a Specific User

1. Navigate to the MDSAppRuntime management bean (MBean), as described in the section "Using the System MBean Browser and the MDSAppRuntime MBean".

2. Export metadata by selecting the exportMetadata operation available from the Operations tab.

3. For toLocation, provide a valid absolute path to a directory or archive in the file system to which the selected documents will be exported. This location must be accessible from the machine where the application is running. If it does not exist, a directory will be created except that when the name ends with ".jar", ".JAR", ".zip", or ".ZIP", an archive will be created. Exporting metadata to an existing archive will overwrite the existing file.

   Example: /tempDir/downloads/mdsExport/RetailAppsFrameworkTestMDS.zip

4. For docs, click the pencil icon. On the Edit Parameter popup, click **Add**, and in the Element box, enter a list of comma-separated, fully qualified document names or document name patterns to export.

   To export customizations for a specific page, simply enter the fully qualified base document name in the Element box.

   Example:

   /oracle/retail/apps/framework/uishell/skin/page/TestTablesAndTrees.jsff

   You can provide the path to multiple documents to export, by clicking the Add button. Do not provide any docs in case you want to export all customizations for the user.

5. For restrictCustTo, click the pencil icon. On the Edit Parameter popup, click **Add**, and in the Element box, enter the list of customization layer names. This can be a list of comma-separated customization layer names used to restrict the operation to only customization documents that match the specified customization layers.

   Each customization layer name can contain, within a pair of brackets, optional customization layer values and value patterns separated by commas. Wildcards (%) may also be used for restricting the operation.

   For example:

Table 3–6 Customization Values

Value	Description
user[buyer]	Restricts operation to the user ’buyer’
user[buyer, better_buyer]	Restricts operation to the users ’buyer’ and ’better_buyer’
user[bu%]	Restricts operation to user names that start with ’bu’ (for example, buyer)
user[be%]	Restricts operation to user names that start with ’be’ (for example, betty_buyer)
user[%bu%]	Restricts operation to users with ’bu’ in the name (for example, buyer and betty_buyer)

6. Click **Invoke**, located at the upper-right corner of the page, to proceed with the export operation.
7. A confirmation message will display in the page. Click **Return** to return to the list of operations.

Deleting All Metadata Services Customizations for a User

1. Navigate to the MDSAppRuntime management bean (MBean), as described in the section "Using the System MBean Browser and the MDSAppRuntime MBean".
2. Delete metadata by selecting the deleteMetadata operation available from the Operations tab.
3. For docs, click the pencil icon. On the Edit Parameter popup, click **Add**, and in the Element box, enter a list of comma-separated, fully qualified document names or document name patterns.

   To delete all customizations for a user, enter "/**" (without the quotes).

   This will recursively delete all customizations under "/" including any other customizations located in the folder(s) under it. Click **OK**.
4. For restrictCustTo, click the pencil icon. On the Edit Parameter popup, click **Add**, and in the Element box, enter the list of customization layer names. This can be used to restrict the operation to only customization documents that match the specified customization layers.

   Each customization layer name can contain, within a pair of brackets, optional customization layer values and value patterns separated by commas. Wildcards (%) may also be used for restricting the operation.

   For example:

Table 3–7 Customization Values

Value	Description
user[buyer]	Restricts operation to the user ’buyer’
user[buyer, better_buyer]	Restricts operation to the users ’buyer’ and ’better_buyer’
user[bu%]	Restricts operation to user names that start with ’bu’ (for example, buyer)
user[be%]	Restricts operation to user names that start with ’be’ (for example, betty_buyer)

Table 3–7 (Cont.) Customization Values

Value	Description
user[%bu%]	Restricts operation to users with 'bu' in the name (for example, buyer and betty_buyer)

5. Click **Invoke**, located at the upper-right corner of the page, to proceed with the delete operation.

6. A confirmation message will display in the page. Click **Return** to return to the list of operations.

Deleting a Customization for a Specific Page for All the Users

1. Navigate to the MDSAppRuntime management bean (MBean), as described in the section "Using the System MBean Browser and the MDSAppRuntime MBean".

2. Delete metadata by selecting the deleteMetadata operation available from the Operations tab.

3. For excludeBaseDocs, select true. This is a Boolean value indicating whether to exclude base metadata documents from being deleted.

4. For docs, click the pencil icon. On the Edit Parameter popup, click **Add**, and in the Element box, enter a list of comma-separated, fully qualified document names or document name patterns.

   To delete all customizations for a user, enter "/**" (without the quotes).

   This will recursively delete all customizations under "/" including any other customizations located in the folder(s) under it. Click **OK**.

5. For restrictCustTo, click the pencil icon. On the Edit Parameter popup, click **Add**, and in the Element box, enter the list of customization layer names. This can be used to restrict the operation to only customization documents that match the specified customization layers.

   Each customization layer name can contain, within a pair of brackets, optional customization layer values and value patterns separated by commas. Wildcards (%) may also be used for restricting the operation.

   For example:

Table 3–8 Customization Values

Value	Description
user[buyer]	Restricts operation to the user 'buyer'
user[buyer, better_buyer]	Restricts operation to the users 'buyer' and 'better_buyer'
user[bu%]	Restricts operation to user names that start with 'bu' (for example, buyer)
user[be%]	Restricts operation to user names that start with 'be' (for example, betty_buyer)
user[%bu%]	Restricts operation to users with 'bu' in the name (for example, buyer and betty_buyer)

6. Click **Invoke**, located at the upper-right corner of the page, to proceed with the delete operation.

7. A confirmation message will display in the page. Click **Return** to return to the list of operations.

Deleting a Customization for a Specific Page for a Particular User

1. Navigate to the MDSAppRuntime management bean (MBean), as described in the section "Using the System MBean Browser and the MDSAppRuntime MBean".
2. Delete metadata by selecting the deleteMetadata operation available from the Operations tab.
3. For excludeBaseDocs, select true. This is a Boolean value indicating whether to exclude base metadata documents from being deleted.
4. For docs, click the pencil icon. On the Edit Parameter popup, click **Add**, and in the Element box, enter a list of comma-separated, fully qualified document names or document name patterns.

   To delete all customizations for a user, enter "/**" (without the quotes).

   This will recursively delete all customizations under "/" including any other customizations located in the folder(s) under it. Click **OK**.
5. For restrictCustTo, click the pencil icon. On the Edit Parameter popup, click **Add**, and in the Element box, enter the list of customization layer names. This can be used to restrict the operation to only customization documents that match the specified customization layers.

   Each customization layer name can contain, within a pair of brackets, optional customization layer values and value patterns separated by commas. Wildcards (%) may also be used for restricting the operation.

   For example:

Table 3–9 Customization Values

Value	Description
user[buyer]	Restricts operation to the user 'buyer'
user[buyer, better_buyer]	Restricts operation to the users 'buyer' and 'better_buyer'
user[bu%]	Restricts operation to user names that start with 'bu' (for example, buyer)
user[be%]	Restricts operation to user names that start with 'be' (for example, betty_buyer)
user[%bu%]	Restricts operation to users with 'bu' in the name (for example, buyer and betty_buyer)

6. Click **Invoke**, located at the upper-right corner of the page, to proceed with the delete operation.
7. A confirmation message will display in the page. Click **Return** to return to the list of operations.

Importing All Metadata Services Customizations

1. Navigate to the MDSAppRuntime management bean (MBean), as described in the section "Using the System MBean Browser and the MDSAppRuntime MBean".
2. Import metadata by selecting the importMetadata operation available from the Operations tab.

3. For fromLocation, enter the path of the directory or archive from which the documents will be imported.

   Example: /tempDir/downloads/mdsExport/RetailAppsFrameworkTestMDS.zip

4. Click **Invoke**, located at the upper-right corner of the page, to proceed with the import operation.

5. A confirmation message will display in the page. Click **Return** to return to the list of operations.

Importing a Specific Page Customization for a User

1. Navigate to the MDSAppRuntime management bean (MBean), as described in the section "Using the System MBean Browser and the MDSAppRuntime MBean".

2. Import metadata by selecting the importMetadata operation available from the Operations tab.

3. For fromLocation, enter the path of the directory or archive from which the documents will be imported.

   Example: /tempDir/downloads/mdsExport/RetailAppsFrameworkTestMDS.zip

4. For excludeBaseDocs, select true. A Boolean value indicating whether to exclude base metadata documents from being imported.

5. For docs, click the pencil icon. On the Edit Parameter popup, click **Add**, and in the Element box, enter a list of comma-separated, fully qualified document names or document name patterns.

   To import customizations for a specific page, simply enter the fully qualified base document name in the Element box.

   For example:
   /oracle/retail/apps/framework/uishell/skin/page/TestTablesAndTrees.jsff

   You can provide the path to multiple documents to be imported, by clicking the **Add**. When done, click **OK**

6. For restrictCustTo, click the pencil icon. On the Edit Parameter popup, click **Add**, and in the Element box, enter the list of customization layer names. This can be used to restrict the operation to only customization documents that match the specified customization layers.

   Each customization layer name can contain, within a pair of brackets, optional customization layer values and value patterns separated by commas. Wildcards (%) may also be used for restricting the operation.

   For example:

Table 3–10 Customization Values

Value	Description
user[buyer]	Restricts operation to the user 'buyer'
user[buyer, better_buyer]	Restricts operation to the users 'buyer' and 'better_buyer'
user[bu%]	Restricts operation to user names that start with 'bu' (for example, buyer)
user[be%]	Restricts operation to user names that start with 'be' (for example, betty_buyer)

Table 3–10 (Cont.) Customization Values

Value	Description
user[%bu%]	Restricts operation to users with 'bu' in the name (for example, buyer and betty_buyer)

7. Click **Invoke**, located at the upper-right corner of the page, to proceed with the import operation.
8. A confirmation message will display in the page. Click **Return** to return to the list of operations.

Creating Metadata Labels

1. Navigate to the MDSAppRuntime management bean (MBean), as described in the section "Using the System MBean Browser and the MDSAppRuntime MBean".
2. Create metadata label by selecting the createMetadataLabel operation available from the Operations tab.
3. For label, enter a valid name of the new metadata label to be created.
4. Click **Invoke**, located at the upper-right corner of the page, to proceed with the operation.
5. A confirmation message will display in the page. Click **Return** to return to the list of operations.

Promoting Metadata Labels

1. Navigate to the MDSAppRuntime management bean (MBean), as described in the section "Using the System MBean Browser and the MDSAppRuntime MBean".
2. Create metadata label by selecting the promoteMetadataLabel operation available from the Operations tab.
3. For label, enter a valid name of the new metadata label to be promoted. Promoting metadata labels can be used to roll back to an earlier version of the document, as captured by the label.
4. Click **Invoke**, located at the upper-right corner of the page, to proceed with the operation.
5. A confirmation message will display in the page. Click **Return** to return to the list of operations.

Listing Metadata Labels

1. Navigate to the MDSAppRuntime management bean (MBean), as described in the section "Using the System MBean Browser and the MDSAppRuntime MBean".
2. 2.List all metadata labels by selecting the listMetadataLabels operation available from the Operations tab.
3. Click **Invoke**, located at the upper-right corner of the page, to proceed with the operation.
4. This will list all the available metadata labels previously created. You can use this, for example, to get the latest metadata label that you want to promote.
5. Click **Return** to return to the list of operations.

Deleting Metadata Labels

1. Navigate to the MDSAppRuntime management bean (MBean), as described in the section "Using the System MBean Browser and the MDSAppRuntime MBean".

2. Delete metadata label by selecting the deleteMetadataLabel operation available from the Operations tab.

3. For label, enter a valid name of the metadata label to be deleted.

4. Click **Invoke**, located at the upper-right corner of the page, to proceed with the operation.

5. A confirmation message will display in the page. Click **Return** to return to the list of operations.

4 In-Context Launching Task Flows in Retail Applications

Retail Applications can expose select task flows retailers can directly launch into. This feature is referred to as in-context launching in a Retail Application.

Retailers can launch these task flows directly through specific URLs.

Retail Applications will provide information about the various task flows retailers can in-context launch into including the URLs and the required parameters.

Retailers can use these URLs in other web pages as links. For example: the URL to a task flow that invokes the Create Allocation Flow in a Retail Application can be added as a link to dashboard report on a BI server.

When the user clicks on a URL to a task flow, the Retail Application will open in a new browser window or tab depending on the specified target of the URL. The requested task flow will be shown as a UI tab within the Retail Application.

Limitations of In-Context Launch via URLs

- The In-Context launch feature will not detect if there is an already opened window or tab for the Retail Application. So when a user clicks on a link to a Retail Application's task flow, say, a report on a BI server, a new browser window or tab will be opened even though the user already has an existing browser window or tab opened for that same Retail Application.
- If a BI dashboard is added on the Retail Application, and if that dashboard has a report that contains links to in-context launchable task flows in the same Retail Application, a new browser window or tab will still be opened.

List of In-Context Launch Task Flows

The Quick Create Allocation container is a quick way to create an allocation. Select a Source and optionally a Source ID along with an Item ID. You can specify that the items should be sourced from a particular warehouse. You can decide to go to the Worksheet for additional processing or to directly skip to the Allocation Maintenance window.

HTTP URL

```
http://<host>:<port>/alloc14/faces/Home?navModelItemId=quickCreateTF
```

Table 4–1 Quick Create

Source	Mandatory Parameter	Optional	Sample URL
PO	source ,sourceIdList and any optional parameter	warehouseIdList itemIdList	http://<host>:<port>/alloc14/faces/Home?navModelItemId=quickCreateTF?source=PO&sourceIdList=<sourceidList> &itemIdList= <itemIdList>
WAREHOUSE	source, warehouseIdList and itemIdList		http://<host>:<port>/alloc14/faces/Home?navModelItemId=quickCreateTF?source=WAREHOUSE&warehouseIdList=<warehouseIdList> &itemIdList=<itemIdList>
WHATIF	source and itemIdList		http://<host>:<port>/alloc14/faces/Home?navModelItemId=quickCreateTF?source=WHATIF&itemIdList=<itemIdList>
Scheduled	source is always warehouse . warehouseIdList,itemIdList are mandatory along with source		http://<host>:<port>/alloc14/faces/Home?navModelItemId=quickCreateTF?source=WAREHOUSE&warehouseIdList=<warehouseIdList> &itemIdList=<itemIdList> &scheduled=true
ASN	source ,sourceIdList and any optional parameter	warehouseIdList itemIdList	http://<host>:<port>/alloc14/faces/Home?navModelItemId=quickCreateTF?source=ASN&sourceIdList=<sourceidList> &itemIdList= <itemIdList>
BOL	source ,sourceIdList and any optional parameter	warehouseIdList itemIdList	http://<host>:<port>/alloc14/faces/Home?navModelItemId=quickCreateTF?source=BOL&sourceIdList=<sourceidList> &itemIdList= <itemIdList>
TRANSFER	source ,sourceIdList and any optional parameter	warehouseIdList itemIdList	http://<host>:<port>/alloc14/faces/Home?navModelItemId=quickCreateTF?source=TRANSFER&sourceIdList=<sourceidList> &itemIdList= <itemIdList>
ALLOCATION	source ,sourceIdList		http://<host>:<port>/alloc14/faces/Home?navModelItemId=quickCreateTF?source=ALLOCATION&sourceIdList=<sourceidList>

Load Allocation

Table 4–2 Load Allocation

Mandatory Parameter	Sample URL
allocationId	http://<host>:<port>/alloc14/faces/Home?navModelItemId=loadAllocationTF?allocationId=<allocationId>

5

Customizing Retail Applications

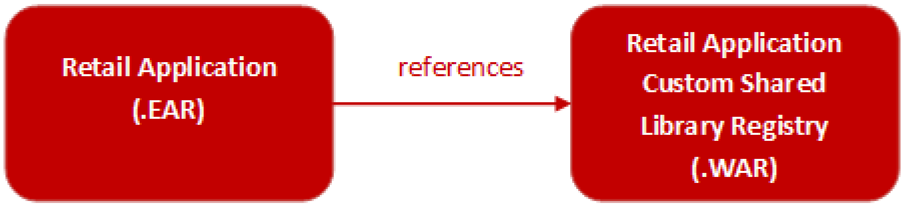
This chapter discusses supported steps for customizing Retail applications.

Using Custom Shared Library for Adding Custom Content

When retailers want to add new content such as new pages, business components, or even Java code into a Retail Application, retailers will need to create a custom shared library, deploy that into the same managed server as the Retail Application, and register that library into the Retail Application.

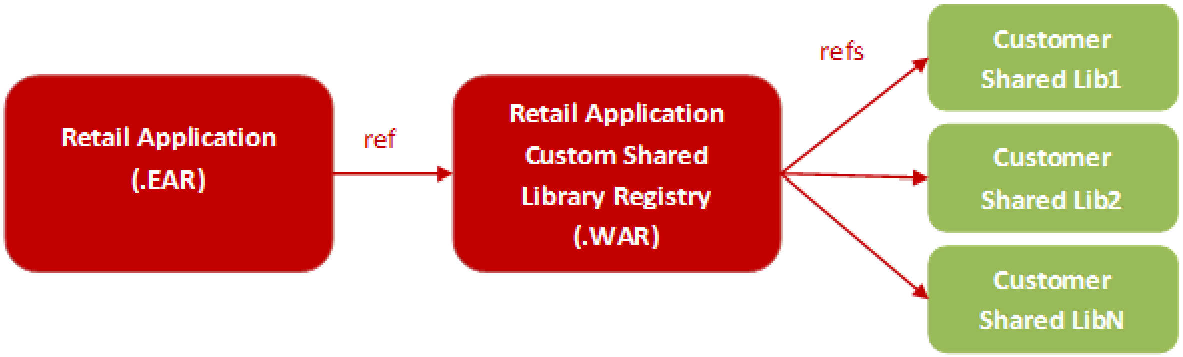
Refer to the WebLogic documentation, *Developing Applications with WebLogic Server*, to learn more about developing and deploying shared libraries in WebLogic.

A Retail application that allows for customization will have as part of its installation an intermediary shared library that will serve as a registry to reference the actual customer-built shared libraries. The diagram below shows the deployment of a Retail Application with the Custom Shared Library Registry.


Retail Application (.EAR)
references
Retail Application Custom Shared Library Registry (.WAR)

When retailers need to add their own content into the application, metadata to register their content into the application's UI including the binaries for the content itself (for example, task flows and pages) are expected to be packaged into a Web Archive (WAR) file and deployed as a shared library in the same managed server as the application itself.

Then, the names of these shared libraries have to be referenced in the Custom Shared Library Registry.


Retail Application (.EAR)
ref
Retail Application Custom Shared Library Registry (.WAR)
refs
Customer Shared Lib1
Customer Shared Lib2
Customer Shared LibN

Creating and Deploying a Custom Shared Library

This section contains instructions on how retailers can create their own shared library that will contain custom content they want to include in the Retail Application UI.

The steps in this section are required before a retailer can customize a Retail Application.

Downloading JDeveloper

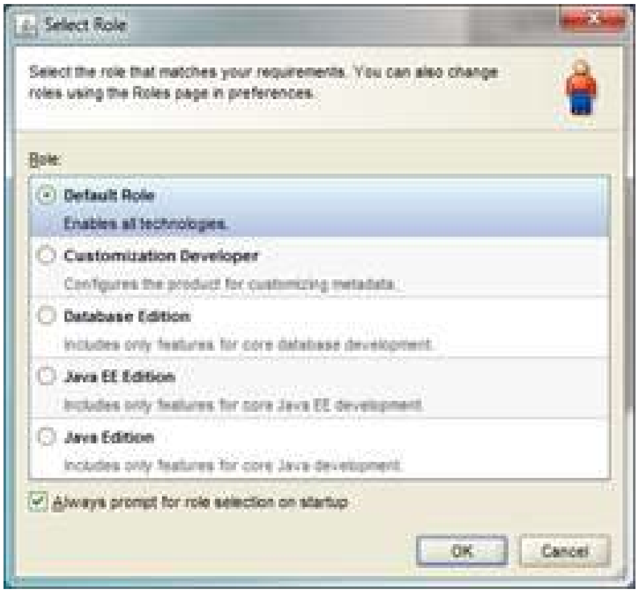
To create the custom shared library, it is recommended that you download and install JDeveloper version 12.1.3 by using the following link:

http://www.oracle.com/technetwork/developer-tools/jdev/downloads/index.html

Creating the Custom Shared Library Workspace through JDeveloper

This section describes the steps to create and configure the Custom Shared Library Workspace in JDeveloper, which will house the code for any custom content the retailer wants to add into the Retail Application. Through JDeveloper, a shared library web application archive file (.WAR) can be generated which can be deployed in the same managed server as the Retail Application.

1. Open JDeveloper and choose **Developer Role** when prompted.



Select Role

Select the role that matches your requirements. You can also change roles using the Roles page in preferences.

Role:

● Default Role
Enables all technologies.

○ Customization Developer
Configures the product for customizing metadata.

○ Database Edition
Includes only features for core database development.

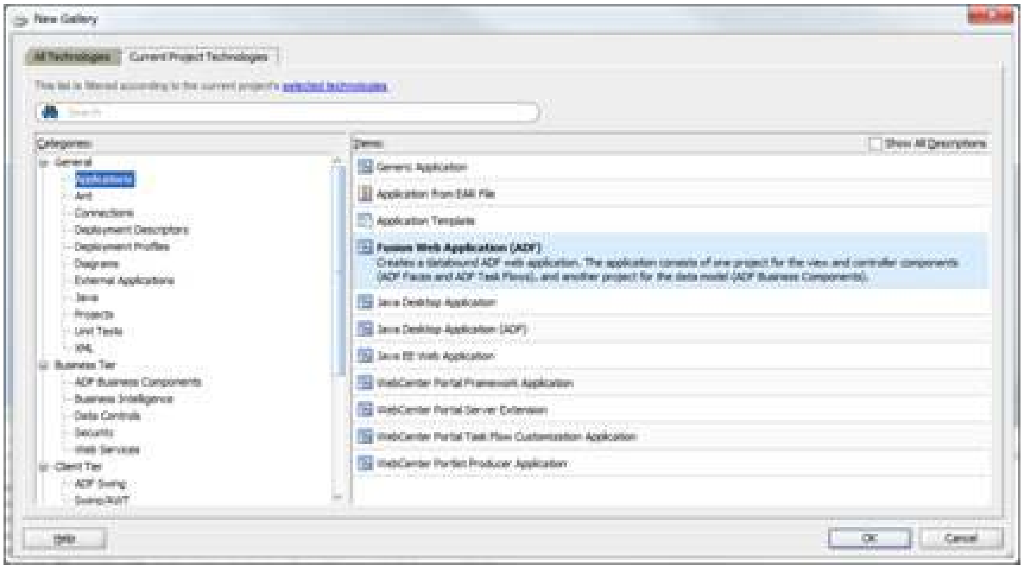
○ Java EE Edition
Includes only features for core Java EE development.

○ Java Edition
Includes only features for core Java development.

☑ Always prompt for role selection on startup

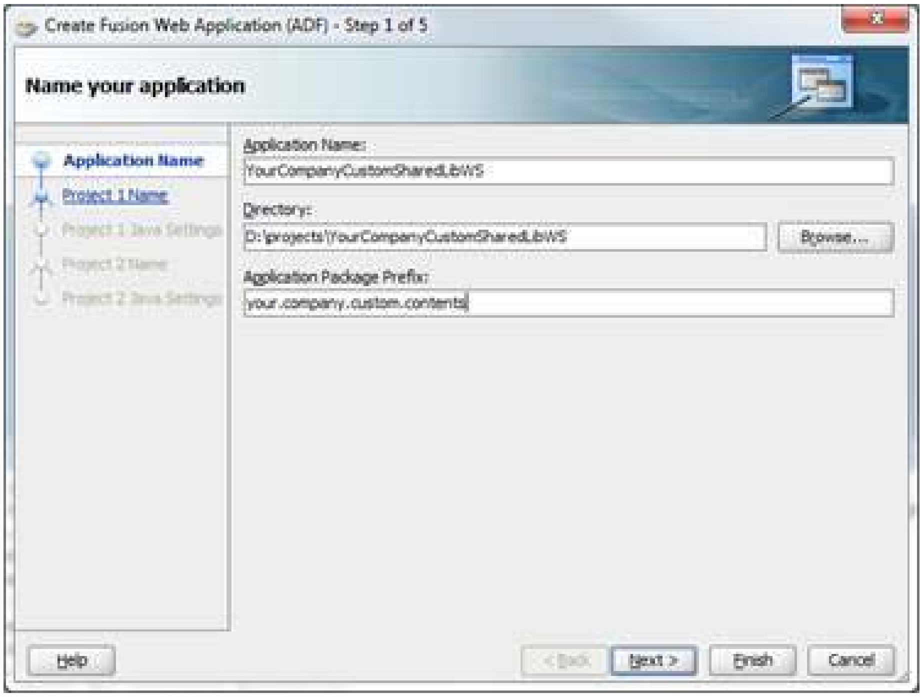
2. Create a new Fusion Web Application JDeveloper workspace.

   a. Go to **File > New** to invoke the New Gallery dialog. Choose **Fusion Web Application (ADF)** as the type of application to create and click **OK**.



Fusion Web Application (ADF)
Creates a databound ADF web application. The application consists of one project for the view and controller components (ADF Faces and ADF Task Flows), and another project for the data model (ADF Business Components).

   b. Provide a meaningful application name, a directory path to the workspace, and an application (Java) package prefix. Click **Finish**.



Application Name: YourCompanyCustomSharedLibWS

Directory: D:\projects\YourCompanyCustomSharedLibWS

Application Package Prefix: your.company.custom.contents

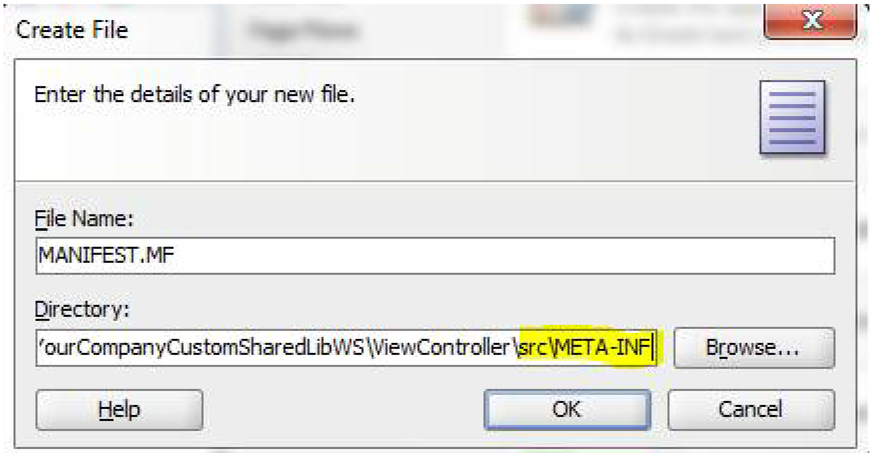
   c. JDeveloper will generate a new workspace with two projects: Model and View-Controller.

3. Add a Manifest file containing the name of the shared library.

   a. Right-click on the **View-Controller** project and choose **New** from the Context menu.

b. The New Gallery dialog appears. Choose the option, **File (General)**, in the All Technologies section of the dialog. Click **OK**.

c. The Create File dialog opens. For the **File Name**, specify MANIFEST.MF. For the **Directory**, the new file **must be added under** the src/META-INF **sub-directory** under the View-Controller project's directory.



Create File

Enter the details of your new file.

File Name:
MANIFEST.MF

Directory:
'ourCompanyCustomSharedLibWS\ViewController\src\META-INF

d. Edit the new MANIFEST.MF file and add the following entries:

```
Manifest-Version: 1.0
Implementation-Vendor: companyName
Implementation-Title: Custom Shared Library for companyName
Implementation-Version: 1.0
Extension-Name: companyname.custom.shared.lib
Specification-Version: 1.0
Created-By: companyName
```

Modify the contents such that meaningful and unique values are used for *Implementation-Vendor*, *Implementation-Title*, *Extension-Name*, and *Created-By*. Example:

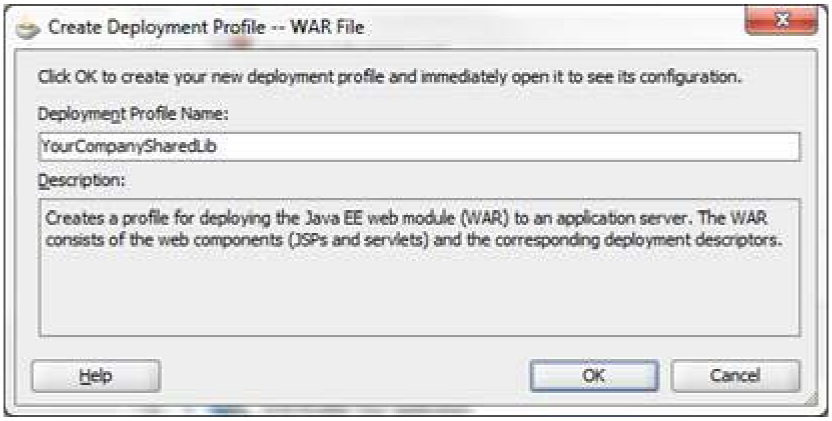
```
Manifest-Version: 1.0
Implementation-Vendor: Acme Retail
Implementation-Title: Custom Shared Library for Acme Retail
Implementation-Version: 1.0
Extension-Name: acmeretail.custom.shared.lib.procurement
Specification-Version: 1.0
Created-By: AcmeRetail
```

4. Create a deployment profile for the shared library.

a. Right-click on the **View-Controller** project and choose **New** from the Context menu.

b. The New Gallery dialog appears. Choose the option, **WARFile (Deployment Profiles)**, in the All Technologies section of the dialog. Click **OK**.

c. Provide a unique and meaningful name for the Deployment Profile and click **OK**.

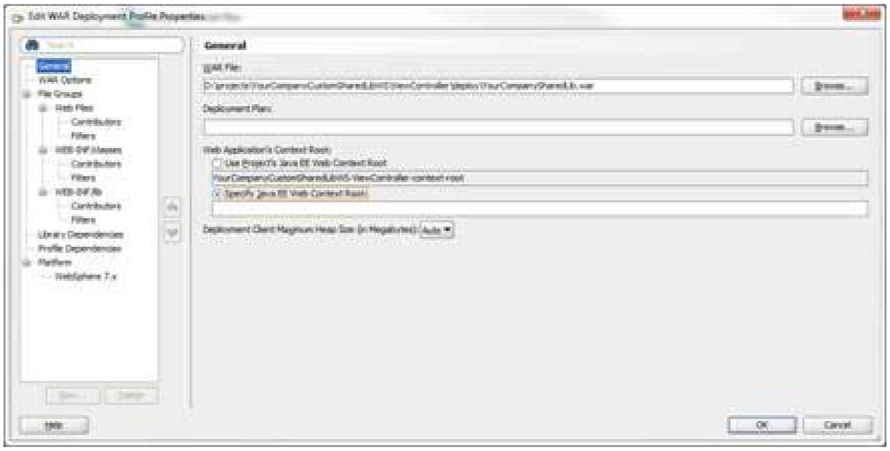


Click OK to create your new deployment profile and immediately open it to see its configuration.

Deployment Profile Name: YourCompanySharedLib

Description: Creates a profile for deploying the Java EE web module (WAR) to an application server. The WAR consists of the web components (JSPs and servlets) and the corresponding deployment descriptors.

d. The Edit WAR Deployment Profile Properties dialog is shown.



General

WAR File: D:\projects\YourCompanyCustomSharedLibWS\ViewController\deploy\YourCompanySharedLib.war

Deployment Plan:

Web Application's Context Root:

☐ Use Project's Java EE Web Context Root

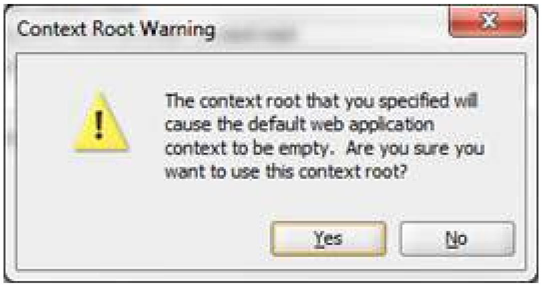
YourCompanyCustomSharedLibWS-ViewController-context-root

☑ Specify Java EE Web Context Root

Deployment Client Maximum Heap Size (in Megabytes): Auto

e. Under the General section, make sure that the **Specify Java EE Web Context Root** is selected without any value.

When you navigate away from this section, you will be prompted to confirm that you really want a blank context root. Click **Yes** to the confirmation dialog.



The context root that you specified will cause the default web application context to be empty. Are you sure you want to use this context root?

f. Under the WAR Options section, **enable** the **Include Manifest File** option and **Add** the MANIFEST.MF file you created under …/View-Controller/src/META-INF.

g. Click **OK**.

Generating and Deploying the Custom Shared Library WAR

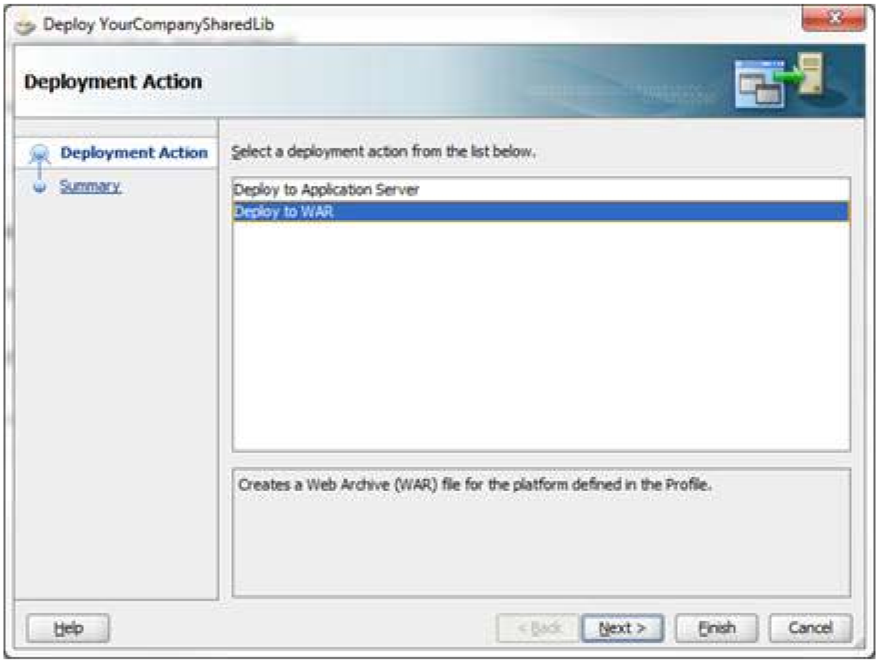
This section describes the steps on how to generate and deploy the Custom Shared Library WAR file:

1. Generate the share library WAR file through JDeveloper.

   a. Open the Custom Shared Library workspace.

   b. Right-click on the **View-Controller** project and choose the shared library WAR deployment profile created previously under the **Deploy** option.

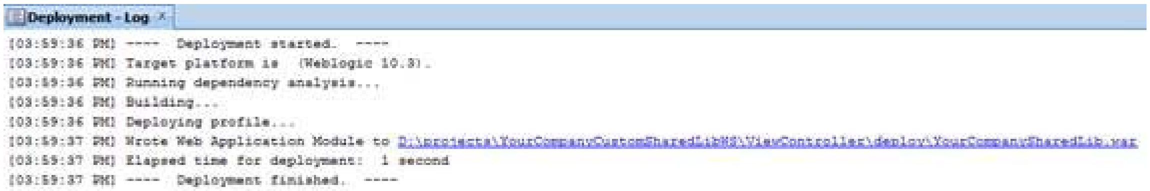
   c. The Deploy dialog opens. Select **Deploy to WAR** and click **Finish**.



Select a deployment action from the list below.

Creates a Web Archive (WAR) file for the platform defined in the Profile.

   d. JDeveloper generates the WAR file into the View-Controller project folder sub-directory, deploy.



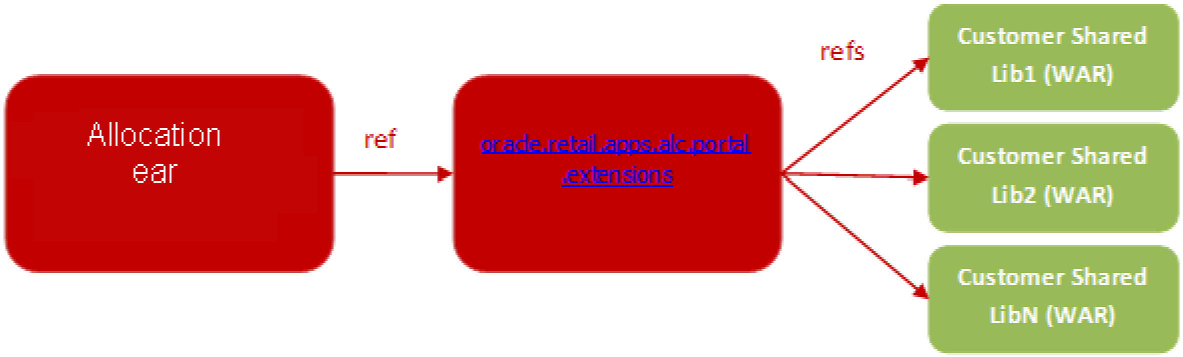
```
[03:59:36 PM] ----  Deployment started.  ----
[03:59:36 PM] Target platform is  (Weblogic 10.3).
[03:59:36 PM] Running dependency analysis...
[03:59:36 PM] Building...
[03:59:36 PM] Deploying profile...
[03:59:37 PM] Wrote Web Application Module to D:\projects\YourCompanyCustomSharedLibWS\ViewController\deploy\YourCompanySharedLib.war
[03:59:37 PM] Elapsed time for deployment:  1 second
[03:59:37 PM] ----  Deployment finished.  ----
```

2. Deploy the generated WAR file to the same managed server as the Retail Application as **a shared library**. Refer to section 6 of the Deploying Applications to Oracle WebLogic Server documentation (`http://docs.oracle.com/cd/E23943_01/web.1111/e13702/deploy.htm`). This task has to be done using the WebLogic Administration Console or Enterprise Manager Fusion Middleware Control with a user having WebLogic administrator permissions.

Referencing the Custom Shared Library from the Retail Application

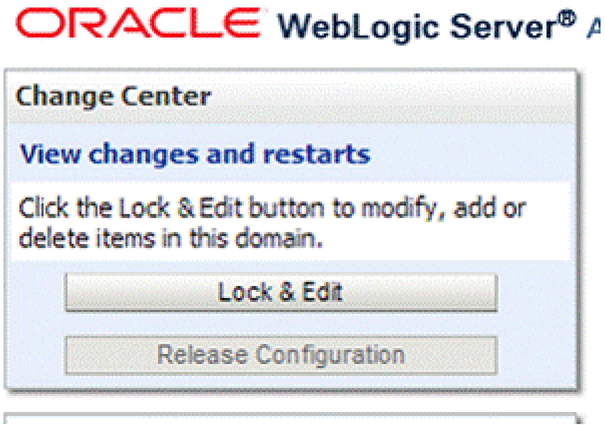
As discussed in the section "Using Custom Shared Library for Adding Custom Content", a customizable Retail Application will include in its deployment an intermediary shared library that will serve as a registry for any other shared libraries the retailer wants to include during runtime of the Retail Application.

Once the retailer has created and deployed its own custom shared library by following the steps in the section "Creating and Deploying a Custom Shared Library", the retailer needs to modify the configuration of the Custom Shared Library Registry to add references to the retailer's shared library.


Allocation ear
ref
oracle.retail.apps.alc.portal.extensions
refs
Customer Shared Lib1 (WAR)
Customer Shared Lib2 (WAR)
Customer Shared LibN (WAR)

To do this:

1. Log in to the WebLogic Administration Console as a user with administrative permissions.
2. If the Administration Console was configured with domain configuration locking, go ahead and click **Lock & Edit** to ensure that other administrators can be prevented from making changes during your edit session.

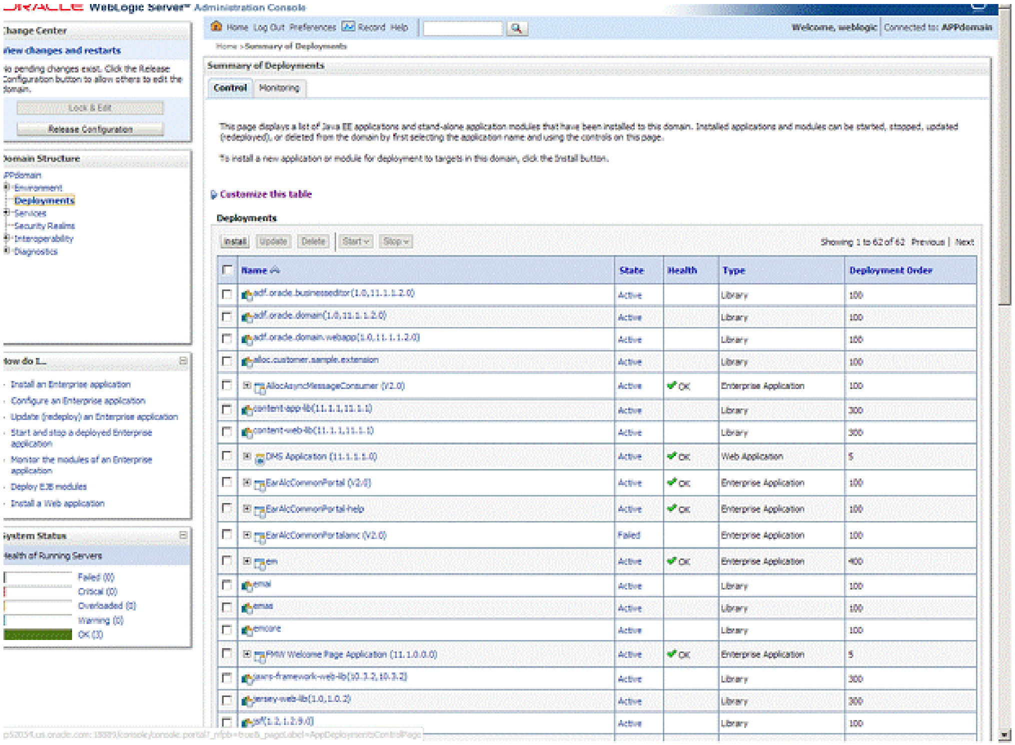


Click the Lock & Edit button to modify, add or delete items in this domain.

oracle.retail.apps.alc.portal.extensions

alloc.customer.sample.extension

3. Navigate to the Deployments section.



Summary of Deployments

This page displays a list of Java EE applications and stand-alone application modules that have been installed to this domain. Installed applications and modules can be started, stopped, updated (redeployed), or deleted from the domain by first selecting the application name and using the controls on this page.

To install a new application or module for deployment to targets in this domain, click the Install button.

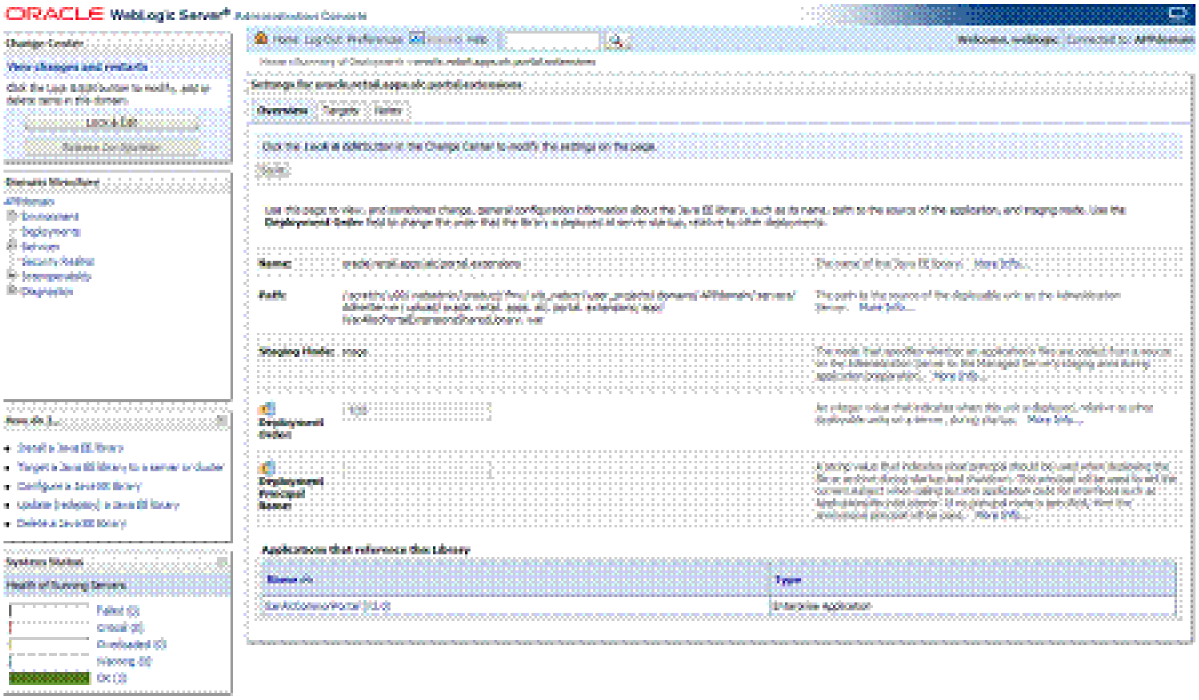
Deployments

Showing 1 to 62 of 62 Previous | Next

Name	State	Health	Type	Deployment Order
adf.oracle.businesseditor(1.0,11.1.1.2.0)	Active		Library	100
adf.oracle.domain(1.0,11.1.1.2.0)	Active		Library	100
adf.oracle.domain.webapp(1.0,11.1.1.2.0)	Active		Library	100
alloc.customer.sample.extension	Active		Library	100
AllocAsyncMessageConsumer (V2.0)	Active	✓ OK	Enterprise Application	100
content-app-lib(11.1.1,11.1.1)	Active		Library	300
content-web-lib(11.1.1,11.1.1)	Active		Library	300
DMS Application (11.1.1.1.0)	Active	✓ OK	Web Application	5
EarAlcCommonPortal (V2.0)	Active	✓ OK	Enterprise Application	100
EarAlcCommonPortal-help	Active	✓ OK	Enterprise Application	100
EarAlcCommonPortalanc (V2.0)	Failed		Enterprise Application	100
em	Active	✓ OK	Enterprise Application	400
emai	Active		Library	100
emas	Active		Library	100
emcore	Active		Library	100
FMW Welcome Page Application (11.1.0.0.0)	Active	✓ OK	Enterprise Application	5
jaxrs-framework-web-lib(10.3.2,10.3.2)	Active		Library	300
jersey-web-lib(1.0,1.0.2)	Active		Library	300
jsf(1.2,1.2.9.0)	Active		Library	100

4. Look for the Retail Application deployment and shut it down. Choose **Force Stop Now** when appropriate. Wait for the shutdown process to complete.

5. Get the deployment location of the Retail Application's custom shared library registry. Under the deployments list, click on the link for the library named `oracle.retail.apps.alc.portal.extensions`. The Settings page for the library appears.



Settings for oracle.retail.apps.alc.portal.extensions

Name: oracle.retail.apps.alc.portal.extensions

Path: [illegible]

Staging Mode: stage

Deployment Order: 100

Applications that reference this Library

Name	Type
EarAlcCommonPortal (V2.0)	Enterprise Application

The Settings page will show the file location of the registry's WAR file under the Path entry.

Make a note of this file location.

6. Using the operating system's file manager application, go to the location of the WAR file. You need read and write permissions to the file system where the WAR file is located.
7. Make a copy of the WAR file as back-up.
8. Open the original WAR file using an archive file manager and update the /WEB-INF/`weblogic.xml` by adding a new <library-ref> entry pointing to your custom shared library.

```
<library-ref>
<library-name>companyname.custom.shared.lib</library-name>
</library-ref>
```

> **Note:** The library-name has to match the Extension-Name you provided in your custom shared library's MANIFEST.MF file.

Once this change is done, you have now linked your custom shared library to the Retail application.

9. Return to the WebLogic Administration Console. Navigate to **Environments' Servers** section. Under the Control tab, select the managed server where the Retail Application is deployed to. Shut it down and start it back up again.

Adding Custom Content into the Custom Shared Library

The Custom ADF and Java based components that add new content or functionality into the Retail Application can be coded or built into the Custom Shared Library workspace. Typically, retailers can include task flows that add new UI workflows into the Retail Application UI.

The WAR for the Custom Shared Library needs to be regenerated when new content is added.

To re-deploy the Custom Shared Library WAR, the Retail Application and its Custom Shared Library Registry must both be shutdown first.

Since Retail Applications are secured web applications, the addition of new ADF UI pages and task flows will need to be secured using the application roles and policies that are recognized by the Retail Application. The provisioning process is done using the Oracle Enterprise Manager Fusion Middleware Control Console.

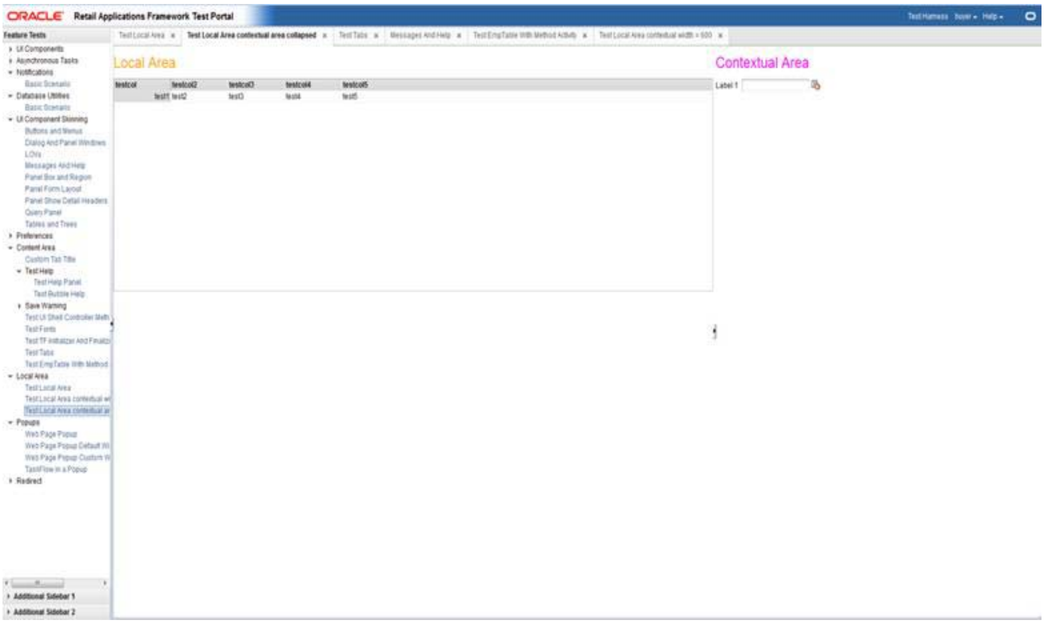
Refer to section 9, Managing the Policy Store, of the *Oracle Fusion Middleware Application Security Guide* for details. (`http://docs.oracle.com/cd/E23943_01/core.1111/e10043.pdf`).

Customizing the Retail Application User Interface

Retail Applications allow retailers to add and display custom content such as ADF task flows and URL to BI reports in the deployed application. These custom contents can be accessed from the Retail Application's user interface (UI).

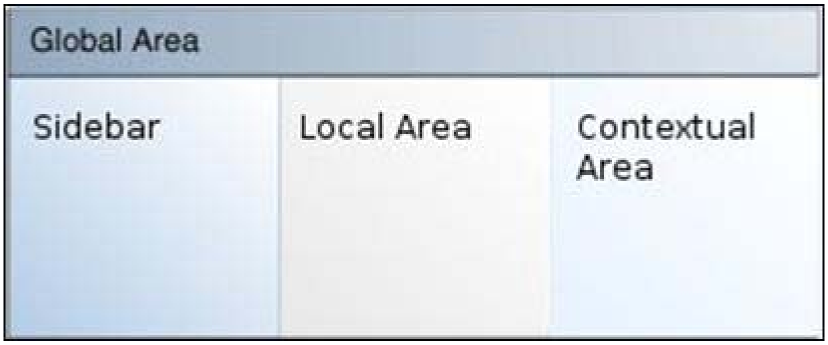
Overview of a Retail Application User Interface

The Retail Application user interface organizes the contents into visual containers that fulfill common layout and navigational requirements in a structured, consistent manner.



testcol	testcol2	testcol3	testcol4	testcol5
test1	test2	test3	test4	test5

The high level containers or areas in the UI are highlighted in the diagram below:



The Global Area presents branding, logged-in user information as well as menus that allow users to switch to other areas of the application. Notifications are also visually represented as an icon in this area.

The Sidebar Area is a collapsible area on the left of the page. A typical content of the sidebar area is the task navigation tree. Each node on the tree presents a business process or task that opens UI workflow screens in the Local Area.

The Local Area is the main content area for the application. The contents change as a result of actions selected in the Sidebar or Global Area. Typically, task flows are opened as tabs within the Local Area.

The Contextual Area is a collapsible area on the right of the page, which provides space to present information that can assist users in completing their tasks. The Contextual Area is presented per Local Area tab. Each task flow in presented in the local area can have their own contextual area.

Adding a Dashboard

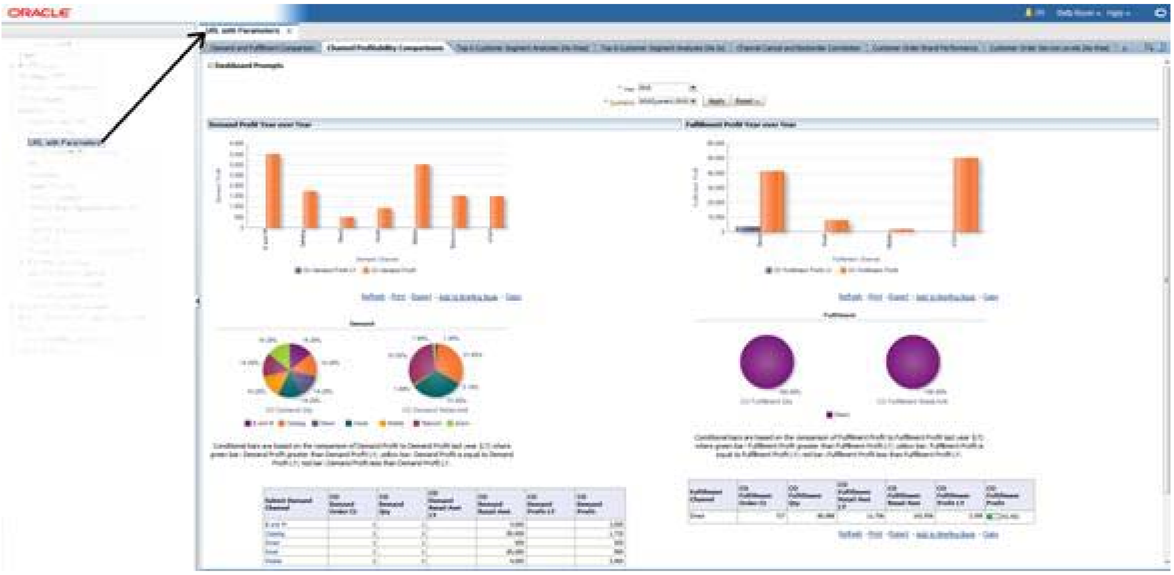
A dashboard is a page within a Retail Application that displays the current status of metrics and key performance indicators relevant to the Retail Application. They are typically tailored for specific roles and they allow you to easily monitor the status of the current data within the application.

For Retail Applications, dashboards are typically built and maintained in a separate BI reporting tool. Oracle Retail recommends using Oracle Business Intelligence Enterprise Edition (OBIEE). Whatever the tool you use, the resulting dashboards must be accessible in a Web browser via a URL.

The dashboards can be added into a Retail Application which a user can launch from the application UI’s sidebar task tree.

The dashboard pages are added into a Retail Application by adding and configuring the application's Sidebar Navigation Model XML file into the Custom Shared Library.

The example below shows the dashboard page as rendered in the local area of a Retail Application's UI.


ORACLE
URL with Parameters
Channel Profitability Comparisons
Dashboard Prompts
Demand Profit Year over Year
Fulfillment Profit Year over Year
URL with Parameters

Preparing the Custom Shared Library for Adding Dashboards

This section describes how to prepare the Custom Shared Library so retailers will be able to add dashboard pages into the Retail Application.

1. Perform the steps described in the section "Creating and Deploying a Custom Shared Library" to create a Custom Shared Library workspace, generate a shared library WAR out of it, deploying the WAR, and associating the library to the Retail Application.

2. Obtain a copy of the application's sidebar navigation model XML file.

   ```
   EarAllocCore.ear\AlcPublicUIViewController.war\WEB-INF\classes\oracle\retail\ap
   ps\framework\uishell\config\custom\HomeSidebarNavigationModel.xml
   ```

3. Using JDeveloper, open the Custom Shared Library workspace in Developer Role.

4. Add the sidebar model XML file in the **View-Controller** project src directory, preferably under a subdirectory called Custom.

Example:

If the sidebar navigation model is named `HomeSidebarNavigationModel.xml`, then that file's path must be `View-Controller/src/custom/HomeSidebarNavigationModel.xml`.

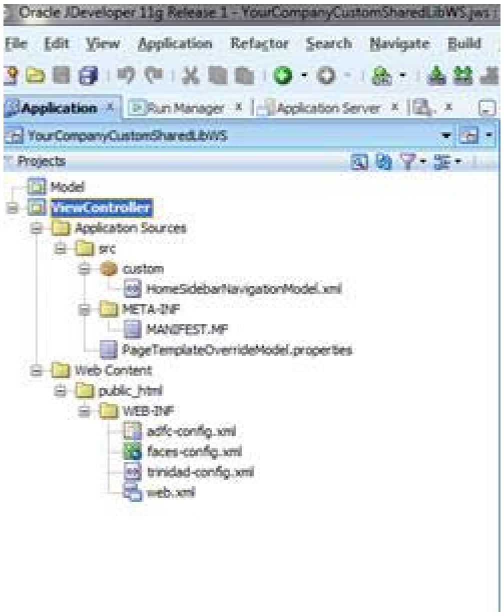
5. Add a new file called `PageTemplateOverrideModel.properties` under the `View-Controller/src` directory. Modify this file and add the following entry:

   ```
   Home.sidebarModel=<path to sidebar model xml within view-controller/src>
   ```

   Example:

   ```
   Home.sidebarModel=/custom/HomeSidebarNavigationModel.xml
   ```

6. Verify the location of the files in the workspace.


Oracle JDeveloper 11g Release 1 - YourCompanyCustomSharedLibWS.jws
File Edit View Application Refactor Search Navigate Build
Application
Run Manager
Application Server
YourCompanyCustomSharedLibWS
Projects
Model
ViewController
Application Sources
src
custom
HomeSidebarNavigationModel.xml
META-INF
MANIFEST.MF
PageTemplateOverrideModel.properties
Web Content
public_html
WEB-INF
adfc-config.xml
faces-config.xml
trinidad-config.xml
web.xml

7. Regenerate the shared library WAR file from the workspace and redeploy the shared library. Shutdown and restart of the Retail Application and its shared library registry is required.

8. Test the Retail Application. The original sidebar contents of the Retail Applications should be accessible.

   The succeeding sections describe the steps to modify the model in order to add dashboard pages into the Retail Application.

Adding a Dashboard into the UI Sidebar

Adding a dashboard page to be accessible from the sidebar area of the Retail Application's UI entails the modification of the Sidebar Navigation Model XML file. The sidebar area renders a collection of links organized or grouped into folders. Each link represents a content that launches into a tab in the UI's local content area. A content is typically a workflow that allows users to accomplish specific tasks in the application. A content can be a dashboard page.

Adding an Oracle Business Intelligence Enterprise Edition based Dashboard into the UI Sidebar

This section describes the steps for adding an Oracle BI EE based Dashboard into the UI Sidebar. Refer to the section, "Oracle Business Intelligence Enterprise Edition Integration in Retail Application" in the Oracle Business Intelligence Enterprise Edition Integration in Retail Applications chapter for more information about Oracle BI EE integration with Retail Applications.

Before adding a dashboard into the UI Sidebar, you must have:

- Built, prepared, deployed, and tested the Custom Shared Library as described in the section "Preparing the Custom Shared Library for Adding Dashboards".
- Created one or more dashboard pages in their BI reporting tool (for example, OBIEE).
  - The Web URL for the dashboard pages must be available in order to proceed with the steps in this section.
  - Any parameters to modify the content of the dashboard must be known and should be accessible as parameters to the dashboard's URL.

Once the above pre-requisites have been satisfied, take the following steps:

1. Obtain the Oracle BI EE connection name from the Retail Application configured to point at the retailer's Oracle BI EE instance. The configuration of this Oracle BI EE connection is performed during the Retail Application installation.

   BI Connection Server name = BIConnection

2. Create dashboard reports in the Oracle BI EE.
3. Build, prepare, deploy, and test the Custom Shared Library as described in the section "Preparing the Custom Shared Library for Adding Dashboards".
4. Open the Custom Shared Library workspace in JDeveloper.
5. Open the application's Sidebar Navigation Model XML file (for example, `View-Controller/src/custom/HomeSidebarNavigationModel.xml`).
6. Go to the bottom of the file and add a new folder into the list. A folder is represented by an <Item>/<Items> pair of XML elements of type "folder". These have to be added within the top most <Items> tag.

   ```
   <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
   <NavigationDefinition … >
   <Items>
   ... (existing contents)
   <!-- Add this paragraph -->
   <Item id="myCustomDashboardFolder" title="My Custom Dashboards" type="folder">
   <Items>
   <!-- add more items here -->
   </Items>
   </Item>
   </Items>
   </NavigationDefinition>
   ```

   Provide a unique identifier to the Item and meaningful values for the title attributes of the Item tag. For this example, we will assume that the folder title is "My Custom Dashboards".

7. Add an <Item> element within the folder that references the dashboard page URL as well as the parameters to control the view of the dashboard if available.

The example below will filter the 'Demand Profit' analysis using the fiscal year as 2012 and fiscal quarters as 2012Quarter1, 2012Quarter2, 2012Quarter3 and 2012Quarter4. The analysis should have prompted filters added for the columns "Business Calendar"."Fiscal Year" and "Business Calendar"."Fiscal Quarter":.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<NavigationDefinition … >
<Items>
... (existing contents)
<Item id="myCustomDashboardsFolder" title="My Custom Dashboards" type="folder">
<Items>
<Item id="myDashboard1"
type="link"
title="Profitability Dashboard">
<url>
<![CDATA[http://mspdv171.us.oracle.com:9704/analytics/saw.dll?SyndicatePages&sy
ndicate=portal&PortalPath=/shared/Customer Order/_portal/Customer
Order&Page=Channel Profitability Comparisons&Action=Navigate]]>
</url>
<Parameters>
<Parameter id="col1">"Business Calendar"."Fiscal Year"</Parameter>
<Parameter id="val1">"2010"</Parameter>
<Parameter id="psa1">"Retail Merchandising Analytics As-Was"</Parameter>
<Parameter id="col2">"Business Calendar"."Fiscal Quarter"</Parameter>
<Parameter id="val2">"2010Quarter1" "2010Quarter2" "2010Quarter3"</Parameter>
<Parameter id="psa2">"Retail Merchandising Analytics As-Was"</Parameter>
</Parameters>
</Item>
</Items>
</Item>
</Items>
</NavigationDefinition>
```

To create this element, note the following attributes and sub-elements:

- The <Item> type must be set as "taskflow".
- The <Item> title must be meaningful.
- The <Item> id attribute must be unique across all the other items in the XML file.
- The <url> sub-element within <Item> indicates thea URL to the dashboard page built in the BI tool. The entire URL must be marked as character data (for example, enclosed in CDATA) to the Retail Applications task flow "BIReportFlow" which allows the integration of the Oracle BI EE dashboard into the Retail Application UI.
- The <Parameters> sub-element within <Item> should list all the parameters to the BIReportFlow task flow including functional parameters to the dashboard being rendered. Each parameter is represented as a <Parameter> element inside <Parameters>.
  - The <Parameter> id should be the actual parameter reference name recognized by the dashboard URL.
  - The value of each <Parameter> is a **string** value. This is the only supported data type.
  - The parameter connectionId refers to the name of the connection configured within the Retail Application to connect to a BI server instance. This is a required parameter of the BIReportFlow task flow.

- The parameter biCatalogPath refers to the path of the BI dashboard within the BI server. This is a required parameter of the BIReportFlow task flow.
- The parameter obipsType specifies if the Oracle BI EE content is an analysis or a dashboard. A value of biDashboardContent is required for dashboard objects.
- The parameter pair parameterNameN and parameterValueN specifies the name and value of the prompt to the report.

More information about the BIReportFlow task flow can be found in the section "Showing Oracle Business Intelligence Enterprise Edition Reports and Dashboards in the Retail Application" in the Oracle Business Intelligence Enterprise Edition Integration in Retail Applications chapter.

8. Re-generate the Custom Shared Library WAR file from the Custom Shared Library workspace. Shutdown the Retail aApplication and its Custom Shared Library Registry, redeploy the Custom Shared Library WAR file, and restart the Retail application components.
9. Test the Retail Application. Log in and verify that a link to the BI dashboard appears in the UI sidebar task tree under a folder "My Custom Dashboards".

Adding a non-Oracle Business Intelligence Enterprise Edition Based Dashboard into the UI Sidebar This section describes the steps for adding non-Oracle BI EE based dashboards into the Retail Application's UI Sidebar. Non-Oracle BI EE based reports such as BI Publisher or Microstrategy can be integrated into the Retail Application by adding the web URL of the dashboard report in the application's sidebar navigation model XML file:

1. Create the dashboard report in the reporting server instance.
2. Obtain the URL to the dashboard report along with any parameters needed to render it.
3. Build, prepare, deploy, and test the Custom Shared Library as described in the section "Preparing the Custom Shared Library for Adding Dashboards".
4. Open the Custom Shared Library workspace in JDeveloper.
5. Open the application's sidebar navigation model XML file (for example, `View-Controller/src/custom/HomeSidebarNavigationModel.xml`).
6. Go to the bottom of the file and add a new folder into the list. A folder is represented by an `<Item>/<Items>` pair of XML elements of type "`folder`". These have to be added within the topmost `<Items>` tag.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<NavigationDefinition … >
  <Items>
    ... (existing contents)

    <!-- Add this paragraph -->
   <Item id="myCustomDashboardFolder" title="My Custom Dashboards"
type="folder">
     <Items>
       <!-- add more items here -->
     </Items>
   </Item>

  </Items>
</NavigationDefinition>
```

Provide unique id to the Item and meaningful values for the title attributes of the Item tag. For this example, we will assume that the folder title is "My Custom Dashboards".

7. Add an `<Item>` element within the folder that references the dashboard page URL as well as parameters to control the view of the dashboard if available.

   The example below shows an `<Item>` element that links to a BI dashboard URL with 6 parameters.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<NavigationDefinition … >
  <Items>
    ... (existing contents)

   <Item id="myCustomDashboardsFolder" title="My Custom Dashboards"
type="folder">
     <Items>

        <Item id="myDashboard1"
           type="link"
           title="Profitability Dashboard">
          <url>
<![CDATA[http://mspdv171.us.oracle.com:9704/analytics/saw.dll?SyndicatePages&sy
ndicate=portal&PortalPath=/shared/Customer Order/_portal/Customer
Order&Page=Channel Profitability Comparisons&Action=Navigate]]>
          </url>
          <Parameters>
            <Parameter id="col1">"Business Calendar"."Fiscal Year"</Parameter>
            <Parameter id="val1">"2010"</Parameter>
            <Parameter id="psa1">"Retail Merchandising Analytics
As-Was"</Parameter>
            <Parameter id="col2">"Business Calendar"."Fiscal
Quarter"</Parameter>
            <Parameter id="val2">"2010Quarter1" "2010Quarter2"
"2010Quarter3"</Parameter>
            <Parameter id="psa2">"Retail Merchandising Analytics
As-Was"</Parameter>
          </Parameters>
        </Item>

     </Items>
   </Item>


  </Items>
</NavigationDefinition>
```

   To create this element, note the following attributes and sub-elements:

   - The <Item> type must be set as "link".
   - The <Item> title must be meaningful.
   - The <Item> id attribute must be unique across all the other items in the XML file.
   - The <url> sub-element within <Item> indicates the URL to the dashboard page built in the BI tool. The entire URL must be marked as character data (for example, enclosed in CDATA).

- The <Parameters> sub-element within <Item> should list all the parameters to the dashboard page if there are any. Each parameter is represented as a <Parameter> element inside <Parameters>.
  - The <Parameter> id should be the actual parameter reference name recognized by the dashboard URL.
  - The value of each <Parameter> is a string value. This is the only supported data type.

8. Re-generate the Custom Shared Library WAR file from the Custom Shared Library workspace. Shutdown the Retail Application and its Custom Shared Library Registry, redeploy the Custom Shared Library WAR file, and restart the Retail Application components.
9. Test the Retail Application. Log-in and verify that a link to the BI dashboard appears in the UI's sidebar task tree under a folder "My Custom Dashboards".

Securing Dashboard Access to Specific Roles

To restrict access to the dashboard added on the sidebar to specific security roles, set the visible property on the <Item> element for the dashboard URL to an Expression Language (EL) expression that calls ADF's securityContext API's isUserInRole method. Example:

```
<Item id="myDashboard1"
type="link"
title="Profitability Dashboard"
visible="#{securityContext.isUserInRole['BUYER_JOB']}" >
```

The parameter to the securityContext.isUserInRole method is a logical security role that is configured for the Retail Application. The API returns true if the user is included in the specified security role. If the user is not authenticated or is not found in the role, the API returns false.

Adding Contextual Reports

Contextual Reports are reports that appear in a task flow's contextual area section. As discussed in the section, "Overview of a Retail Application User Interface", the Contextual Area is a collapsible section to the right of the local area that provides a space to present information that can assist users in completing their tasks.

Since information presented in a contextual area is presented depending on the task or workflow the user is on, contextual areas are associated with task flows and there can be at most one contextual area per task flow.

Within a contextual area, multiple contextual reports can be configured.

Each contextual report can change its contents depending on the action being performed in the user’s current workflow.

For example, Metrics pertaining to an item included in an allocation being maintained by a user can be displayed in the Allocation Maintenance Flow. As the user clicks through the list of items, the corresponding metrics on contextual area change.

Each task flow publishes the contextual business events on key activities taking place in the screen. Contextual reports can listen to those events and change its contents depending on the payload information associated with the event.

Contextual reports are typically built and maintained in a separate BI reporting tool. Oracle Retail recommends using Oracle Business Intelligence Enterprise Edition

(OBIEE). Whatever the tool the retailer users, the resulting reports must be accessible in a web browser via a URL.

The contextual reports are added in a Retail Application by adding and configuring the task flow's contextual area model XML file into the Custom Shared Library.

Retail Applications can also provide you with the list of possible contextual business events each flow generates. Retailers can configure their contextual reports to react to these events. Each event includes information about the event's payload information (for example, the item ID of the item being selected in an Allocation Maintenance screen).

List of Contextual Business Events and Payloads

The following table lists the contextual business events and the payloads:

Event Name	Task Flow	Page	Contextual Area Model XML Location	Generated when...	Payload Values
UpdateContextAwareReportEvent	AllocationMaintenanceFlow	Alloc Maintenance Page	EAR name à war nameà jar name à oracle\retail\apps\framework\uishell\config\custom\contextualarea	Allocation Maintenance, Item Review row change	■ itemId ■ allocId ■ depatmentId ■ classId ■ subclassId ■ diff1 ■ diff2 ■ diff3 ■ wh ■ doc ■ docType ■ Lang ■ locale
itemSourceLocEventBinding	AllocResultsFlow	Alloc Maintenance Page	EAR name à war nameà jar name à oracle\retail\apps\framework\uishell\config\custom\contextualarea	Alloc Maintenance, Item Result row change	■ itemId ■ diff1 ■ diff2 ■ diff3 ■ Location ■ Lang ■ locale
itemforcastEventBinding	AllocResultsFlow	Alloc Maintenance Page	EAR name à war nameà jar name à oracle\retail\apps\framework\uishell\config\custom\contextualarea	Alloc Maintenance, Item Result row change	■ itemId ■ diff1 ■ diff2 ■ diff3 ■ Location ■ Lang ■ locale

Preparing the Custom Shared Library for Adding Contextual Reports

This section describes how to prepare the Custom Shared Library so retailers will be able to add contextual reports in Retail Application task flows.

1. Perform the steps described in the section "Creating and Deploying a Custom Shared Library" to create a Custom Shared Library workspace, generate a shared library WAR out of it, deploying the WAR, and associating the library to the Retail Application.
2. Obtain a copy of the task flow contextual area model XML files where the contextual reports will be added. Refer to section, "List of Contextual Business Events and Payloads".
3. Using JDeveloper, open the Custom Shared Library workspace in a Developer Role.
4. Add the contextual area model XML file in the **View-Controller** project src directory, preferably under a sub-directory called Custom.

   Example:

   If the contextual area model XML for the task flow AllocMaintFlow is called AllocMaintFlowContextualAreaModel.xml, then that file's path must be View-Controller/src/custom/AllocMaintFlowContextualAreModel.xml.
5. Add a new or open the existing file called `PageTemplateOverrideModel.properties` under the `View-Controller/src` directory. Modify this file and add the following entry:

   ```
   Home.sidebarModel=<path to sidebar model xml within view-controller/src>
   <FlowName>.contextualAreaModel=<path to the contextual area model for the flow>
   ```

   For example:

   ```
   AllocMaintFlow.sidebarModel=/custom/AllocMaintFlowContextualAreaModel.xml
   ```
6. Regenerate the shared library WAR file from the workspace and redeploy the shared library. Shutdown and restart of the Retail Application and its shared library registry is required.
7. Test the Retail application. Navigate to the flow and make sure the flow is functional.

Adding a Contextual Report To A Task Flow

Adding a contextual report to a task flow primary entails the modification of the task flow's Contextual Area Model XML file. Each report is rendered in a collapsible panel box.

Before adding a dashboard into the UI Sidebar, the retailer must have:

- Built, prepared, deployed, and tested the Custom Shared Library as described in the section "Preparing the Custom Shared Library for Adding Contextual Reports".
- Obtained information about the Retail Application's list of contextual business events that can be broadcast from various workflows.
- Created one or more contextual BI reports in the BI reporting tool (for example, OBIEE).
  - The web URL for each report must be available in order to proceed with the steps in this section.

- Any parameters to configure the content of the report must be known and should be accessible as parameters to the dashboard's URL.

Once the above pre-requisites have been satisfied, proceed with the following steps:

1. Assume the following example scenario when following the steps:
   - A contextual report called "Item Metrics" showing information about an item is to be added to the Allocation Maintenance Flow's main page. When the user selects an item on the page, the report displays information for the selected item.
   - The Retail Application has provided the following information about the contextual business event:

Event Name	Task Flow	Page	Contextual Area Model XML Location	Generated when...	Payload Values
AlcAllocMaintenanceItemReviewSelectedEvent	AllocationMaintenanceFlow	AllocMaintenancePage		Allocation Maintenance, Item Review row change	■ selectedItem - the item selected by the user. Always generated. ■ selectedItemType - the type of item. Can be "regular" or "pack". Optional. If not supplied, assume item is a regular type.

   - The Item Metrics report was built in the retailer’s BI reporting tool (for example, OBIEE). It was built considering the payload values the contextual business event will generate.
2. Open the Custom Shared Library workspace in JDeveloper.
3. Open the task flow contextual area model XML file (for example, `ViewController/src/custom/AllocMaintFlowContextualAreaModel.xml`).
4. Add an <Item> element within the topmost <Items> element that reference the task flow called `ViewContextAwareReportFlow`. The `ViewContextAwareReportFlow` is a framework for rendering URL based reports that will be aware of contextual business events emanating from the Retail aplication task flows.

   Example:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<NavigationDefinition … >
<Items>
...
...
<Item id="showItemMetric"
type="taskflow"
title="Item Metric">
<url>
```

```
/WEB-INF/oracle/retail/apps/framework/contextawarereport/publicui/flow/ViewCont
extAwareReportFlow.xml#ViewContextAwareReportFlow
</url>
</Item>
</Items>
</NavigationDefinition>
```

Note:

- Make sure that the <Item> id is unique.
- Make sure the <Item> type is "taskflow".
- Provide a meaningful title.

5. Populate the parameters to the ViewContextAwareReportFlow by adding the following <Parameter>/<Parameters> elements:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<NavigationDefinition … >
<Items>
<Item id="showCustomerMetric"
type="taskflow"
title="Customer Metric">
<url>
/WEB-INF/oracle/retail/apps/framework/contextawarereport/publicui/flow/ViewCont
extAwareReportFlow.xml#ViewContextAwareReportFlow
</url>
<Parameters>
<Parameter id="reportDescription">Item Metric</Parameter>
<Parameter id="actionType">AllocMaintItemSelectedEvent</Parameter>
<Parameter id="primaryUrl">
<![CDATA[http://companyobiee.com:9704/report/shared/itemMetric&paramItemId=<sel
ectedItemId>&paramItemType=<selectedItemType:token01>&paramLanguage=<language>]
]>
</Parameter>
<Parameter id="token01">regular</Parameter>
</Parameters>
</Item>
</Items>
</NavigationDefinition>
```

Note:

- The <Parameter id="reportDescription"> element is the title of the contextual area report. Set this to a meaningful value.
- The <Parameter id="actionType"> element indicates the contextual business event name the report listens to.
- The <Parameter id="primaryUrl"> element indicates the URL for the contextual area report in the BI server. The entire URL must be marked as character data (for example, enclosed in CDATA). Note how the parameters to the URL are *tokenized*.
  - The "?paramItemId=<selectedItemId>" portion of the URL instructs the system to pass the contextual business event payload value called selectedItemId into the URL parameter paramItemId when rendering the contextual report.
  - The "?paramItemType=<selectedItemType:token01>" portion of the URL instructs the system to pass the contextual business event payload value called selectedItemType into the URL parameter paramItemType when rendering the contextual report. If that payload value is empty or null at runtime, then a default value of regular is used as referenced in a <Parameter id="token01"> entry.
  - The "?paramLanguage=<language>" portion of the URL instructs the system about your current locale. The "language" identifier is a reference to a value in the contextual event payload. This is a built-in value that exists in all Retail application contextual business event payloads.
- The <Parameter id="token01"> element holds the default value for the URL parameter selectedItemType. Token parameters hold default values and you can define up to 20 default value tokens.

6. Re-generate the Custom Shared Library WAR file from the Custom Shared Library workspace. Shutdown the Retail Application and its Custom Shared Library Registry, redeploy the Custom Shared Library WAR file, and restart the Retail Application components.
7. Test the Retail Application. Go to the flow where the report was added and verify that the report is rendered correctly.

6

Oracle Business Intelligence Enterprise Edition Integration in Retail Application

Oracle Business Intelligence Enterprise Edition is a comprehensive business intelligence platform that delivers a full range of capabilities, including interactive dashboards, ad hoc queries, notifications and alerts, enterprise and financial reporting, scorecard and strategy management, business process invocation, search and collaboration, mobile, integrated systems management and more. Oracle BI EE is based on a proven web service-oriented unified architecture that integrates with an organization's existing information technology infrastructure for the lowest total cost of ownership and highest return on investment.

It is a common requirement to embed the content from Oracle BI EE into ADF applications. Using the Oracle BI EE Presentation Services connection, Retail Applications can embed Oracle BI Presentation Catalog objects such as analysis and dashboard onto their ADF pages.

Showing Oracle Business Intelligence Enterprise Edition Reports and Dashboards in the Retail Application

Oracle BI EE Catalog Objects can be displayed in a Retail Application using the BIReportFlow task flow `(/WEB-INF/oracle/retail/apps/framework/bi/flow/BIReportFlow.xml#BIReportFlow)`. Typically, this task flow is configured to be launched from the application's sidebar task panel to render a dashboard page. Refer to the section, "Adding a non-Oracle Business Intelligence Enterprise Edition Based Dashboard into the UI Sidebar" in the Customizing Retail Applications chapter for more information on adding Oracle BI EE based dashboards into the Retail Application.

The BIReportFlow taskflow takes the following input parameters:

Table 6–1

Input Parameter Name	Type	Required	Description
connectionId	java.lang.String	true	The connection name of the BISoapConnection. Example: BIConnection
biCatalogPath	java.lang.String	true	The path to the Oracle BI report or dashboard. Example: /shared/Customer Order/Demand Profit

Table 6–1 (Cont.)

Input Parameter Name	Type	Required	Description
obipsType	java.lang.String	false	The obipsType input parameter specifies if the Oracle BI EE content is an analysis or a dashboard. The acceptable values for the obipsType are biReportContent or biDashboardContent. If this input parameters is not provided biReportContent is used by default.
subItem	java.lang.String	false	The subItem input parameter specifies the sub item for the Oracle BI EE content. If the Oracle BI EE content is an analysis then a specific view from the analysis can be displayed using the subItem attribute. The valid values are compoundView!1 dvtchart!1 narrativeView!1 tableView!1 titleView!1 viewSelector!1 If the Oracle BI EE content is a dashboard then a specific tab from the dashboard can be displayed using the subItem attribute. The valid value will be the name of the tab in the dashboard.
viewReportLinks	java.lang.String	false	The input parameter viewReportLinks is used to display the links to refresh, print or export an analysis. The valid values are r - refresh p - print d - export Example: rfd
parameterName1	java.lang.String	false	The name of the filter or prompt. Example: "Business Calendar"."Fiscal Quarter" A maximum of ten parameters can be passed to the Oracle BI EE content using the following input parameters to the taskflow. parameterName1, parameterName2, parameterName3, parameterName4, parameterName5, parameterName6, parameterName7, parameterName8, parameterName9, parameterName10.

Table 6–1 (Cont.)

Input Parameter Name	Type	Required	Description
parameterValues1	java.lang.String	false	The value of the filter or prompt to filter the analysis or dashboard. The parameterValues can be EL expression. Example: 2012Quarter1;2012Quarter2;2012Quarter3;2012Quarter4 A maximum of ten parameter values can be passed to the Oracle BI EE content using the following input parameters to the taskflow. parameterValues1, parameterValues2, parameterValues3, parameterValues4, parameterValues5, parameterValues6, parameterValues7, parameterValues8, parameterValues9, parameterValues10. The parameterValues[n] should match parameterName[n]. In the example above, the column "Business Calendar"."Fiscal Quarter" will be filtered with the values 2012Quarter1, 2012Quarter2, 2012Quarter3 and 2012Quarter4. Multiple values should be separated with a semi-colon. Example: 2012Quarter1;2012Quarter2;2012Quarter3;2012Quarter4

Table 6–1 (Cont.)

Input Parameter Name	Type	Required	Description

additionalParameters	java.util.HashMap	false	A map that can be used to specify the parameters for the Oracle BI EE content. This parameter takes an EL expression.
			Example:
			#{TestBIReportFlowBean.demandProfitParametersMap}
			Where the demandProfitParametersMap can be populated as follows
			
			public Map<String, List> getDemandProfitParametersMap() {
			Map<String, List> demandProfitParametersMap = new HashMap<String, List>();
			
			List yearList = new ArrayList();
			yearList.add(2010);
			demandProfitParametersMap.put("\"Business Calendar\".\"Fiscal Year\"", yearList);
			
			List quarterList = new ArrayList();
			quarterList.add("2010Quarter1");
			quarterList.add("2010Quarter2");
			quarterList.add("2010Quarter3");
			quarterList.add("2010Quarter4");
			demandProfitParametersMap.put("\"Business Calendar\".\"Fiscal Quarter\"", quarterList);
			
			return demandProfitParametersMap;
			}

dashboardWidth	java.lang.String	false	The width of the Oracle BI EE dashboard. This attribute is used when the obipsType is biDashboardContent. If not provided it defaults to 100%.
dashboardHeight	java.lang.String	false	The height of the Oracle BI EE dashboard. This attribute is used when the obipsType is biDashboardContent. If not provided it defaults to 100%.

Various Examples of the BIReportFlow Taskflow

The examples below shows how the BIReportFlow taskflow can be used in the SideBarNavigationModel.xml file of the Retail Application to display the Oracle BI EE content in the application tab.

The example below will display the 'Demand Profit' analysis from the Oracle BI EE using the connection BIConnection.

```
<Item id="testBIReportFlow"
```

```
           title="#{adfBundle['RetailAppsJunitUiBundle'].TEST_BI_REPORT_FLOW}"
           type="taskflow">

<url>/WEB-INF/oracle/retail/apps/framework/bi/flow/BIReportFlow.xml#BIReportFlow</
url>
          <Parameters>
            <Parameter id="connectionId">BIConnection</Parameter>
            <Parameter id="biCatalogPath">/shared/Customer Order/Demand
Profit</Parameter>
            <Parameter id="obipsType">biReportContent</Parameter>
          </Parameters>
        </Item>
```

The example below will display the refresh print and export links in the 'Demand Profit' analysis.

```
        <Item id="testBIReportWithRefreshPrintDownloadEnabled"
           title="#{adfBundle['RetailAppsJunitUiBundle'].TEST_BI_REPORT_WITH_
REFRESH_PRINT_EXPORT_ENABLED}"
           type="taskflow">

<url>/WEB-INF/oracle/retail/apps/framework/bi/flow/BIReportFlow.xml#BIReportFlow</
url>
          <Parameters>
            <Parameter id="connectionId">BIConnection</Parameter>
            <Parameter id="biCatalogPath">/shared/Customer Order/Demand
Profit</Parameter>
            <Parameter id="obipsType">biReportContent</Parameter>
            <Parameter id="viewReportLinks">rfd</Parameter>
          </Parameters>
        </Item>
```

The example below will display just the dvtchart view from the 'Demand Profit' analysis.

```
        <Item id="testDVTChartView"
           title="#{adfBundle['RetailAppsJunitUiBundle'].TEST_DVT_CHART_VIEW}"
           type="taskflow">

<url>/WEB-INF/oracle/retail/apps/framework/bi/flow/BIReportFlow.xml#BIReportFlow</
url>
          <Parameters>
            <Parameter id="connectionId">BIConnection</Parameter>
            <Parameter id="biCatalogPath">/shared/Customer Order/Demand
Profit</Parameter>
            <Parameter id="obipsType">biReportContent</Parameter>
            <Parameter id="subItem">dvtchart!1</Parameter>
          </Parameters>
        </Item>
```

The example below will display just the table view from the 'Demand Profit' analysis.

```
        <Item id="testTableView"
           title="#{adfBundle['RetailAppsJunitUiBundle'].TEST_TABLE_VIEW}"
           type="taskflow">

<url>/WEB-INF/oracle/retail/apps/framework/bi/flow/BIReportFlow.xml#BIReportFlow</
url>
          <Parameters>
            <Parameter id="connectionId">BIConnection</Parameter>
            <Parameter id="biCatalogPath">/shared/Customer Order/Demand
```

```
Profit</Parameter>
            <Parameter id="obipsType">biReportContent</Parameter>
            <Parameter id="subItem">tableView!1</Parameter>
          </Parameters>
        </Item>
```

The example below will filter the 'Demand Profit' analysis using the fiscal year as 2012 and fiscal quarters as 2012Quarter1, 2012Quarter2, 2012Quarter3 and 2012Quarter4. The analysis should have prompted filters added for the columns Business Calendar"."Fiscal Year" and "Business Calendar"."Fiscal Quarter".

```
<Item id="testBIReportWithParameters"
           title="#{adfBundle['RetailAppsJunitUiBundle'].TEST_BI_REPORT_WITH_
PARAMETERS}"

           type="taskflow">

<url>/WEB-INF/oracle/retail/apps/framework/bi/flow/BIReportFlow.xml#BIReportFlow</
url>
          <Parameters>
            <Parameter id="connectionId">BIConnection</Parameter>
            <Parameter id="biCatalogPath">/shared/Customer Order/Demand
Profit</Parameter>
            <Parameter id="obipsType">biReportContent</Parameter>
            <Parameter id="viewReportLinks">rfd</Parameter>
            <Parameter id="parameterName1">"Business Calendar"."Fiscal
Year"</Parameter>
            <Parameter id="parameterValues1">2012</Parameter>
            <Parameter id="parameterName2">"Business Calendar"."Fiscal
Quarter"</Parameter>
            <Parameter
id="parameterValues2">2012Quarter1;2012Quarter2;2012Quarter3;2012Quarter4</Paramet
er>
          </Parameters>
        </Item>
```

The example below will filter the 'Demand Profit' analysis using the fiscal year as 2010 and fiscal quarters as 2010Quarter1, 2010Quarter2, 2010Quarter3 and 2010Quarter4. Note that this example does not pass the parameters directly. It uses an EL expression to pass a parameter map with all the parameters.

```
        <Item id="testBIReportWithParametersMap"
           title="#{adfBundle['RetailAppsJunitUiBundle'].TEST_BI_REPORT_WITH_
PARAMETERS_MAP}"
           type="taskflow">

<url>/WEB-INF/oracle/retail/apps/framework/bi/flow/BIReportFlow.xml#BIReportFlow</
url>
          <Parameters>
            <Parameter id="connectionId">BIConnection</Parameter>
            <Parameter id="biCatalogPath">/shared/Customer Order/Demand
Profit</Parameter>
            <Parameter id="obipsType">biReportContent</Parameter>
            <Parameter id="viewReportLinks">rfd</Parameter>
            <Parameter
id="additionalParameters">#{TestBIReportFlowBean.demandProfitParametersMap}</Param
eter>
          </Parameters>
        </Item>
```

The demandProfitParametersMap can be created as follows in a bean.

```
  public Map<String, List> getDemandProfitParametersMap() {
    Map<String, List> demandProfitParametersMap = new HashMap<String, List>();

    List yearList = new ArrayList();
    yearList.add(2010);
    demandProfitParametersMap.put("\"Business Calendar\".\"Fiscal Year\"",
yearList);

    List quarterList = new ArrayList();
    quarterList.add("2010Quarter1");

    quarterList.add("2010Quarter2");
    quarterList.add("2010Quarter3");
    quarterList.add("2010Quarter4");
    demandProfitParametersMap.put("\"Business Calendar\".\"Fiscal Quarter\"",
quarterList);

    return demandProfitParametersMap;
  }
```

The example below will show the 'Customer Order' dashboard with all its tabs. Note that the obipsType has been changed to biDashboardContent. It also passes the dashboard prompt "Business Calendar"."Fiscal Year" as 2011 to filter the results in the dashboard for the year 2011.

```
        <Item id="testBIDashboardAllTabs"
           title="#{adfBundle['RetailAppsJunitUiBundle'].TEST_BI_DASHBOARD_ALL_
TABS}"
           type="taskflow">

<url>/WEB-INF/oracle/retail/apps/framework/bi/flow/BIReportFlow.xml#BIReportFlow</
url>
          <Parameters>
            <Parameter id="connectionId">BIConnection</Parameter>
            <Parameter id="biCatalogPath">/shared/Customer Order/_portal/Customer
Order</Parameter>
            <Parameter id="obipsType">biDashboardContent</Parameter>
            <Parameter id="parameterName1">"Business Calendar"."Fiscal
Year"</Parameter>
            <Parameter id="parameterValues1">2011</Parameter>
          </Parameters>
        </Item>
```

The example below shows how to display one specific tab of the Oracle BI EE dashboard. It displays the 'Channel Profitability Comparisons' tab of the 'Customer Order dashboard. It passes the parameters map as EL expression just like the previous example.

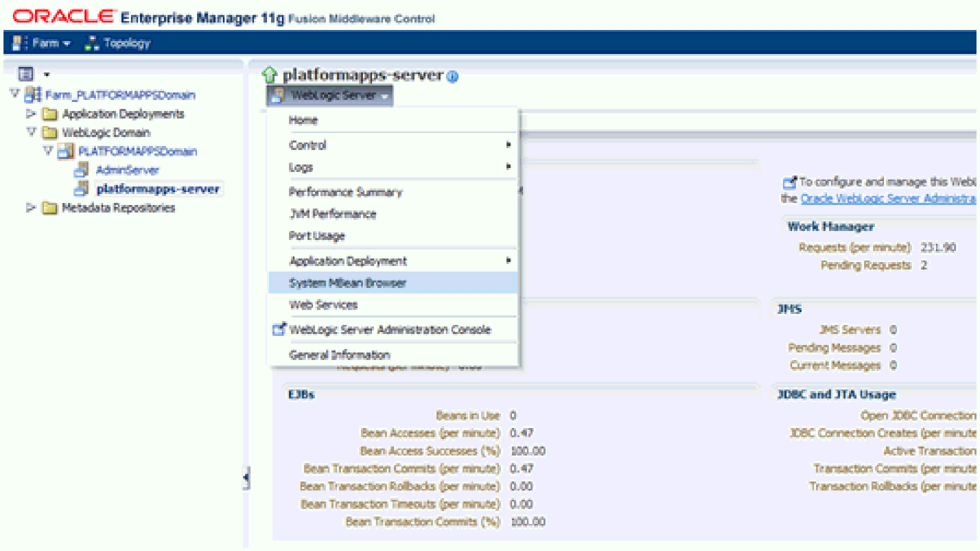
```
        <Item id="testBIDashboardSpecificTab"
           title="#{adfBundle['RetailAppsJunitUiBundle'].TEST_BI_DASHBOARD_
SPECIFIC_TAB}"
           type="taskflow">

<url>/WEB-INF/oracle/retail/apps/framework/bi/flow/BIReportFlow.xml#BIReportFlow</
url>
          <Parameters>
            <Parameter id="connectionId">BIConnection</Parameter>
            <Parameter id="biCatalogPath">/shared/Customer Order/_portal/Customer
Order</Parameter>
            <Parameter id="obipsType">biDashboardContent</Parameter>
            <Parameter id="subItem">Channel Profitability Comparisons</Parameter>
            <Parameter
id="additionalParameters">#{TestBIReportFlowBean.dashboardParametersMap}</Paramete
r>
          </Parameters>
        </Item>
```

Modifying the BIConnection Attributes at Runtime

When the application gets installed on the WebLogic Server, BIConnection is automatically created by the installer. However, it may happen that we may want to change the connection attributes post the application installation. This can be done for troubleshooting the existing BI connection or changing the BI connection to use a different Oracle BI EE server. The BI connection attributes can be modified using the Enterprise Manager using the following steps.

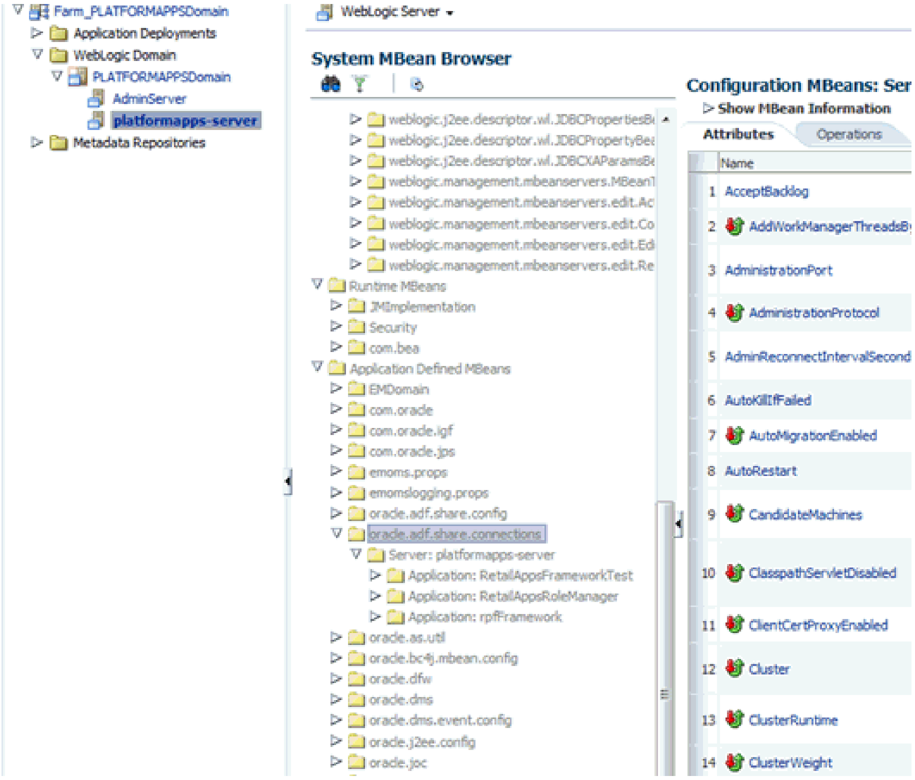
1. Login to the Enterprise Manager of the environment where the Oracle Retail Application is deployed and go to 'System MBean Browser' as shown below.

Figure 6–1 System MBean Browser


ORACLE Enterprise Manager 11g Fusion Middleware Control
Farm
Topology
Farm_PLATFORMAPPSDomain
Application Deployments
WebLogic Domain
PLATFORMAPPSDomain
AdminServer
platformapps-server
Metadata Repositories
platformapps-server
WebLogic Server
Home
Control
Logs
Performance Summary
JVM Performance
Port Usage
Application Deployment
System MBean Browser
Web Services
WebLogic Server Administration Console
General Information
To configure and manage this Webl
the Oracle WebLogic Server Administra
Work Manager
Requests (per minute) 231.90
Pending Requests 2
JMS
JMS Servers 0
Pending Messages 0
Current Messages 0
EJBs
Beans in Use 0
Bean Accesses (per minute) 0.47
Bean Access Successes (%) 100.00
Bean Transaction Commits (per minute) 0.47
Bean Transaction Rollbacks (per minute) 0.00
Bean Transaction Timeouts (per minute) 0.00
Bean Transaction Commits (%) 100.00
JDBC and JTA Usage
Open JDBC Connection
JDBC Connection Creates (per minute
Active Transaction
Transaction Commits (per minute
Transaction Rollbacks (per minute

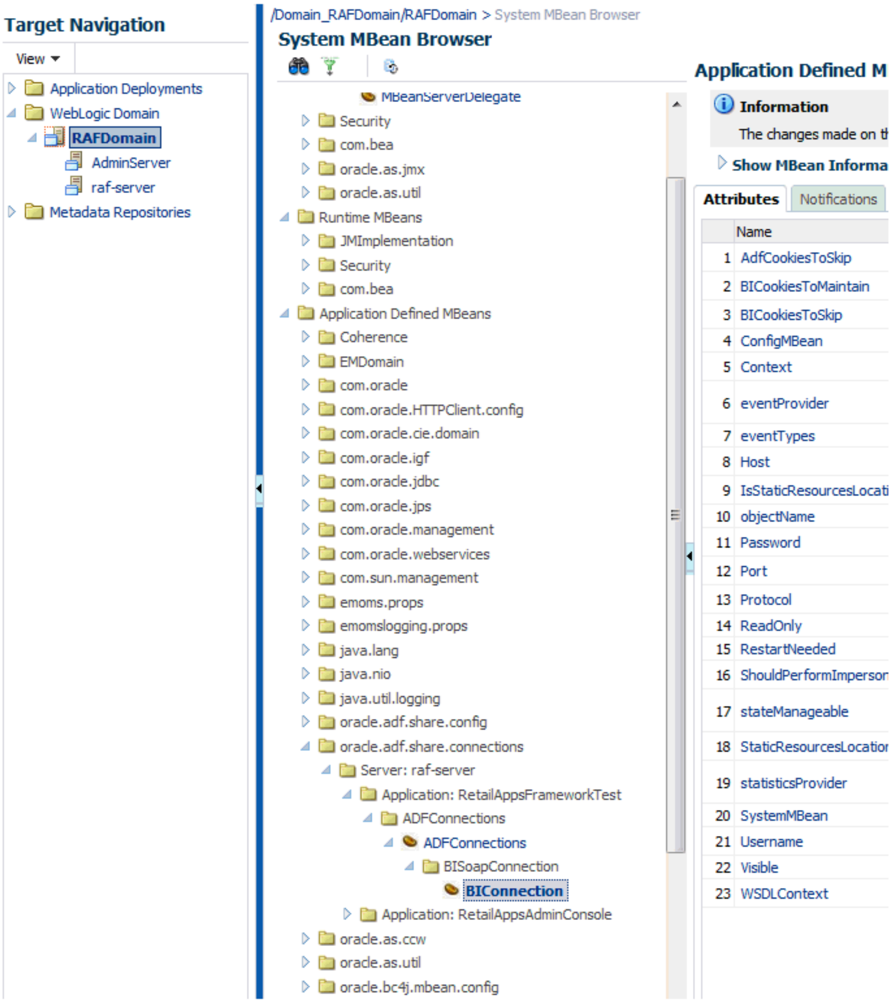
2. Go to 'Application Defined MBeans' and its sub folder 'oracle.adf.share.connections'.

Figure 6–2 Application Defined MBeans



3. Expand the Oracle Retail application and select 'ADFConnections'. If the BIConnection is already created then you would be able to see the BIConnection as shown below.

Figure 6–3 BIConnection

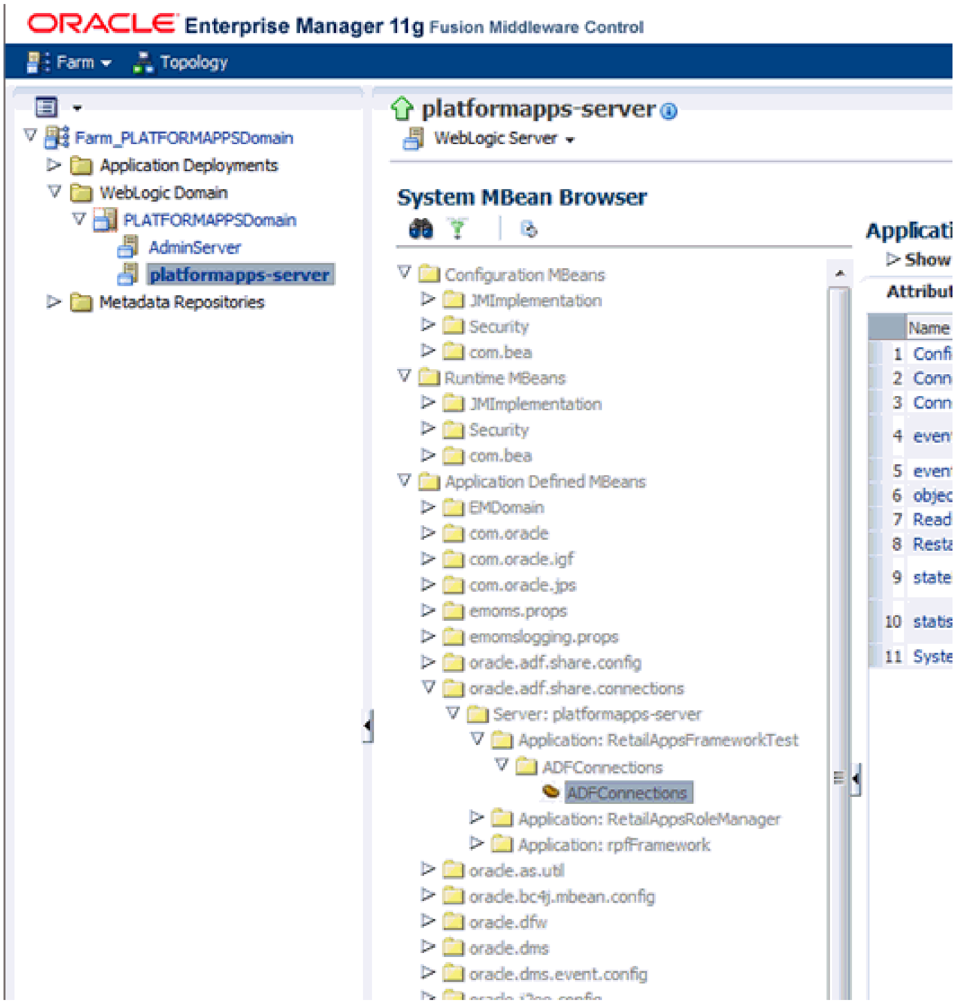


The changes made on th

	Name
1	AdfCookiesToSkip
2	BICookiesToMaintain
3	BICookiesToSkip
4	ConfigMBean
5	Context
6	eventProvider
7	eventTypes
8	Host
9	IsStaticResourcesLocati
10	objectName
11	Password
12	Port
13	Protocol
14	ReadOnly
15	RestartNeeded
16	ShouldPerformImpersor
17	stateManageable
18	StaticResourcesLocatior
19	statisticsProvider
20	SystemMBean
21	Username
22	Visible
23	WSDLContext

If the BIConnection is not available then the 'ADFConnections' will not be expandable as shown below.

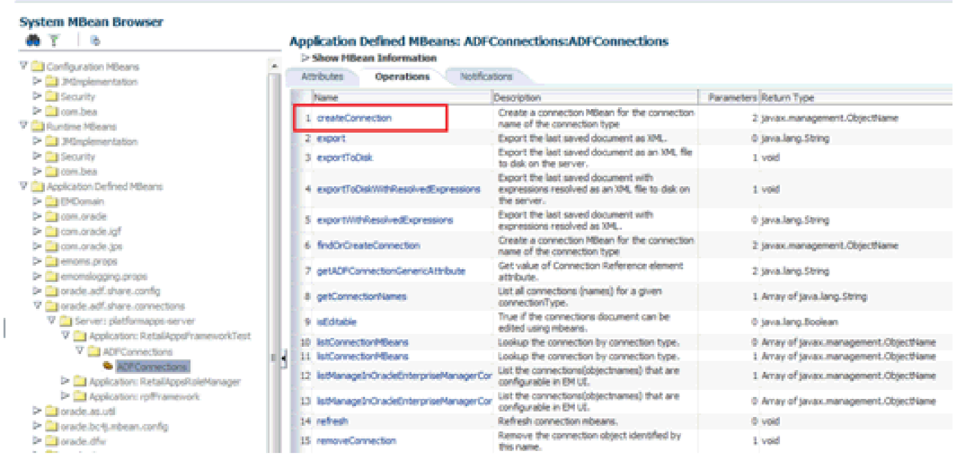
Figure 6–4 ADFConnections



Do the steps below to create a BIConnection if it is not available.

4. Go to the 'Operations' tab and click 'createConnection'.

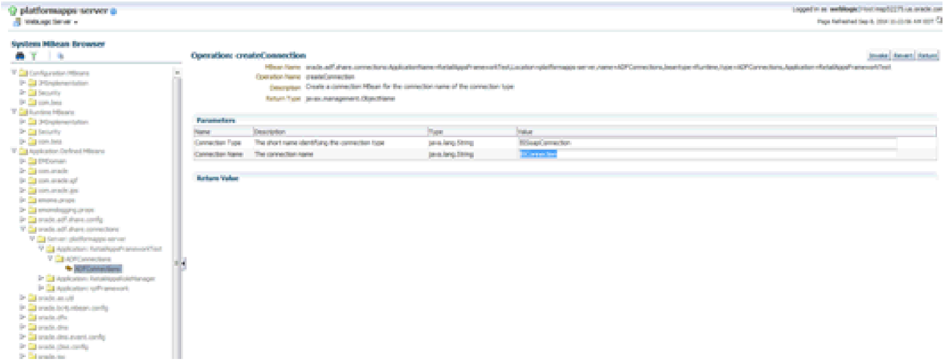
Figure 6–5 Create Connection



	Name	Description	Parameters	Return Type
1	createConnection	Create a connection MBean for the connection name of the connection type	2	javax.management.ObjectName
2	export	Export the last saved document as XML.	0	java.lang.String
3	exportToDisk	Export the last saved document as an XML file to disk on the server.	1	void
4	exportToDiskWithResolvedExpressions	Export the last saved document with expressions resolved as an XML file to disk on the server.	1	void
5	exportWithResolvedExpressions	Export the last saved document with expressions resolved as XML.	0	java.lang.String
6	findOrCreateConnection	Create a connection MBean for the connection name of the connection type	2	javax.management.ObjectName
7	getADFConnectionGenericAttribute	Get value of Connection Reference element attribute.	2	java.lang.String
8	getConnectionNames	List all connections (names) for a given connectionType.	1	Array of java.lang.String
9	isEditable	True if the connections document can be edited using mbeans.	0	java.lang.Boolean
10	listConnectionMBeans	Lookup the connection by connection type.	0	Array of javax.management.ObjectName
11	listConnectionMBeans	Lookup the connection by connection type.	1	Array of javax.management.ObjectName
12	listManageInOracleEnterpriseManagerCor	List the connections(objectnames) that are configurable in EM UI.	1	Array of javax.management.ObjectName
13	listManageInOracleEnterpriseManagerCor	List the connections(objectnames) that are configurable in EM UI.	0	Array of javax.management.ObjectName
14	refresh	Refresh connection mbeans.	0	void
15	removeConnection	Remove the connection object identified by this name.	1	void

5. Enter Connection Type as BISoapConnection and Connection Name as BIConnection. Click **Invoke** on top right.

Figure 6–6 BISoapConnection



Operation: createConnection

MBean Name oracle.adf.share.connections:ApplicationName=RetailAppsFrameworkTest,Location=platformapps-server,name=ADFConnections,beantype=Runtime,type=ADFConnections,Application=RetailAppsFrameworkTest

Operation Name createConnection

Description Create a connection MBean for the connection name of the connection type

Return Type javax.management.ObjectName

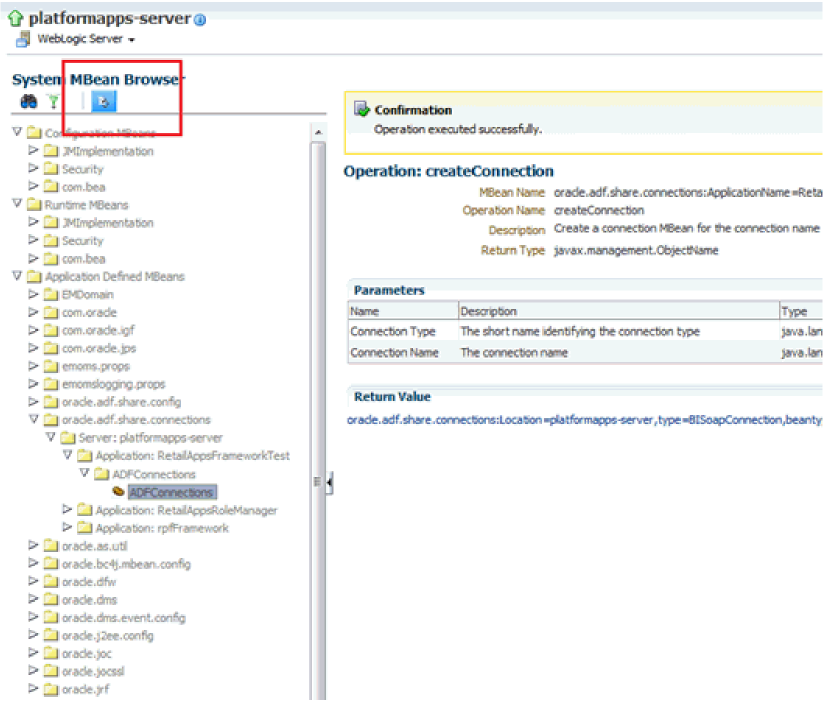
Parameters

Name	Description	Type	Value
Connection Type	The short name identifying the connection type	java.lang.String	BISoapConnection
Connection Name	The connection name	java.lang.String	BIConnection

Return Value

6. Refresh the tree using the button 'Refresh cached tree data'.

Figure 6–7 Refresh Cached Tree Data



Confirmation

Operation executed successfully.

Operation: createConnection

MBean Name oracle.adf.share.connections:ApplicationName=Reta

Operation Name createConnection

Description Create a connection MBean for the connection name

Return Type javax.management.ObjectName

Parameters

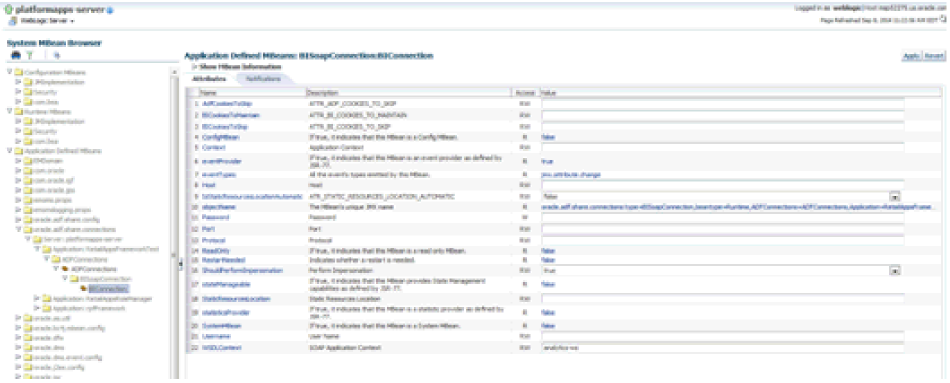
Name	Description	Type
Connection Type	The short name identifying the connection type	java.lan
Connection Name	The connection name	java.lan

Return Value

oracle.adf.share.connections:Location=platformapps-server,type=BISoapConnection,beanty

7. Once the tree is refreshed, you will notice that 'ADFConnections' becomes a folder and it has a new connection 'BIConnection'. Click on the BIConnection to open its properties in the right pane.

Figure 6–8 BIConnection Properties



Application Defined MBeans: BISoapConnection:BIConnection

	Name	Description	Access	Value
1	AdfCookiesToSkip	ATTR_ADF_COOKIES_TO_SKIP	RW	
2	BICookiesToMaintain	ATTR_BI_COOKIES_TO_MAINTAIN	RW	
3	BICookiesToSkip	ATTR_BI_COOKIES_TO_SKIP	RW	
4	ConfigMBean	If true, it indicates that this MBean is a Config MBean.	R	false
5	Context	Application Context	RW	
6	eventProvider	If true, it indicates that this MBean is an event provider as defined by JSR-77.	R	true
7	eventTypes	All the event's types emitted by this MBean.	R	jmx.attribute.change
8	Host	Host	RW	
9	IsStaticResourcesLocationAutomatic	ATTR_STATIC_RESOURCES_LOCATION_AUTOMATIC	RW	false
10	objectName	The MBean's unique JMX name	R	oracle.adf.share.connections:type=BISoapConnection,beantype=Runtime,ADFConnections=ADFConnections,Application=RetailAppsFrame...
11	Password	Password	W	
12	Port	Port	RW	
13	Protocol	Protocol	RW	
14	ReadOnly	If true, it indicates that this MBean is a read only MBean.	R	false
15	RestartNeeded	Indicates whether a restart is needed.	R	false
16	ShouldPerformImpersonation	Perform Impersonation	RW	true
17	stateManageable	If true, it indicates that this MBean provides State Management capabilities as defined by JSR-77.	R	false
18	StaticResourcesLocation	Static Resources Location	RW	
19	statisticsProvider	If true, it indicates that this MBean is a statistic provider as defined by JSR-77.	R	false
20	SystemMBean	If true, it indicates that this MBean is a System MBean.	R	false
21	Username	User Name	RW	
22	WSDLContext	SOAP Application Context	RW	analytics-ws

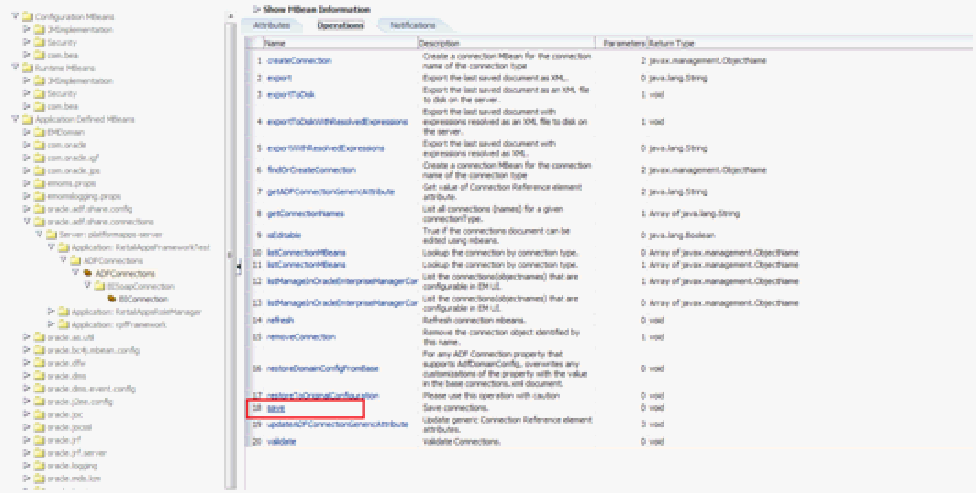
8. Enter the values shown in the table below. Once done, click **Apply** on the top right.

Table 6–2 BIConnection Properties

Value	Description
Context	The context root of the Oracle BI EE installation. Example: analytics.
Host	The host of the Oracle BI EE installation.
IsStaticResourcesLocationAutomatic	The IsStaticResourcesLocationAutomatic attribute can be either true or false. When this attribute is true, the BI Presentation Server will be used to specify the location of the static resources. When this attribute is false, the attribute 'StaticResourcesLocation' should be provided. The attribute StaticResourcesLocation will be used to find the static resources when the attribute IsStaticResourcesLocationAutomatic is false.
Password	The password for the BIImpersonateUser.
Port	The port of the Oracle BI managed server. Example 9704
Protocol	http For production environments, use https.
ShouldPerformImpersonation	true (keep it true, default is true)
StaticResourcesLocation	The path of the location where the static resources are available. If the IsStaticResourcesLocationAutomatic attribute is true then use the following URL for the StaticResourcesLocation attribute. http://<obiee_host>:<obiee_port>/analytics If the IsStaticResourcesLocationAutomatic attribute is false, then use the following URL for the StaticResourcesLocation attribute. http://<web_tier_host>:<web_tier_port> The Oracle BI EE static resources, namely the 'res' and 'analyticsRes' directories should be available from the Oracle Web Tier's htdocs directory when the IsStaticResourcesLocationAutomatic attribute is false.
Username	The username that has been configured on the Oracle BI EE with the impersonate permissions. Example: BIImpersonateUser
WSDLContext	analytics-ws

9. Go to 'ADFConections' to save the changes. Click on '**save**' operation in the Operations tab.

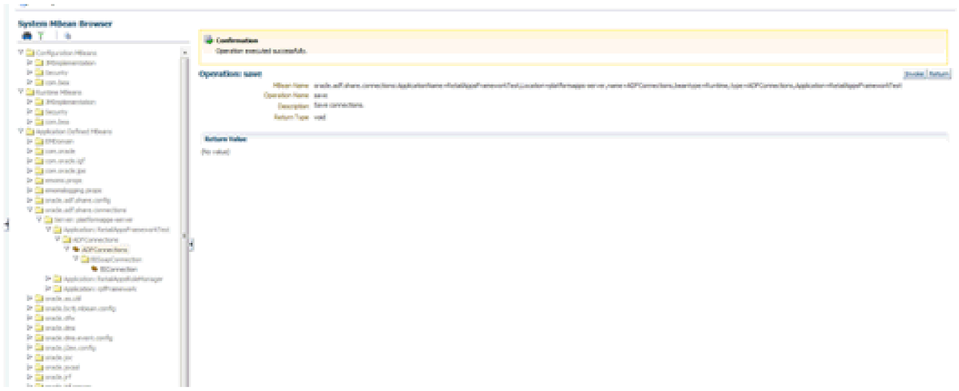
Figure 6–9 Save Operation



	Name	Description	Parameters	Return Type
1	createConnection	Create a connection MBean for the connection name of the connection type	2	javax.management.ObjectName
2	export	Export the last saved document as XML.	0	java.lang.String
3	exportToDisk	Export the last saved document as an XML file to disk on the server.	1	void
4	exportToDiskWithResolvedExpressions	Export the last saved document with expressions resolved as an XML file to disk on the server.	1	void
5	exportWithResolvedExpressions	Export the last saved document with expressions resolved as XML.	0	java.lang.String
6	findOrCreateConnection	Create a connection MBean for the connection name of the connection type	2	javax.management.ObjectName
7	getADFConnectionGenericAttribute	Get value of Connection Reference element attribute.	2	java.lang.String
8	getConnectionNames	List all connections (names) for a given connectionType.	1	Array of java.lang.String
9	isEditable	True if the connections document can be edited using mbeans.	0	java.lang.Boolean
10	listConnectionMBeans	Lookup the connection by connection type.	0	Array of javax.management.ObjectName
11	listConnectionMBeans	Lookup the connection by connection type.	1	Array of javax.management.ObjectName
12	listManageInOracleEnterpriseManagerCor	List the connections(objectnames) that are configurable in EM UI.	1	Array of javax.management.ObjectName
13	listManageInOracleEnterpriseManagerCor	List the connections(objectnames) that are configurable in EM UI.	0	Array of javax.management.ObjectName
14	refresh	Refresh connection mbeans.	0	void
15	removeConnection	Remove the connection object identified by this name.	1	void
16	restoreDomainConfigFromBase	For any ADF Connection property that supports AdfDomainConfig, overwrites any customizations of the property with the value in the base connections.xml document.	0	void
17	restoreToOriginalConfiguration	Please use this operation with caution	0	void
18	save	Save connections.	0	void
19	updateADFConnectionGenericAttribute	Update generic Connection Reference element attributes.	3	void
20	validate	Validate Connections.	0	void

10. Click **Invoke** on the top right to save the changes.

Figure 6–10 Invoke Save



Confirmation

Operation executed successfully.

Operation: save

MBean Name oracle.adf.share.connections:ApplicationName=RetailAppsFrameworkTest,Location=platformapps-server,name=ADFConnections,beantype=Runtime,type=ADFConnections,Application=RetailAppsFrameworkTest

Operation Name save

Description Save connections.

Return Type void

Return Value

(No value)

11. Once the connection attributes for the BIConnection are saved, the changes are available to the Oracle Retail Application. There is no need to restart the server after saving. Simply logging out and logging in the application will suffice.

Enabling In-Context Launch from the Oracle Business Intelligence Enterprise Edition Content

The retail application provides the capability to launch taskflows in a new UIShell tab, from the Oracle BI EE content embedded directly in the application. The data from the Oracle BI EE analysis or dashboard can be passed to the application taskflow enabling in-context launch of the taskflow from the Oracle BI EE analysis or dashboard.

The retail application leverages the Action Framework of Oracle BI EE to deliver actionable business intelligence to user. The Action Framework provides functionality for creating, managing, and invoking actions. Actions are created and managed in the

Oracle BI Presentation Services user interface. The retail application uses the ADF Contextual Event action to execute actions in the application.

More details about the ADF Contextual Event Action can be found in the document.

Oracle Fusion Middleware Developer's Guide for Oracle Business Intelligence Enterprise Edition at the following location:

https://docs.oracle.com/middleware/11119/biee/BIEDV/embedding_adf.htm#BIEDV1214

Enabling the Application Development Framework Contextual Event Action in Oracle Business Intelligence Enterprise Edition

The ADF Contextual event action is not enabled by default in the Oracle BI EE installation. It has to be manually enabled. Edit the following file in your Oracle BI EE installation

```
<OBIEE_HOME>/instances/<INSTANCE_
NAME>/config/OracleBIPresentationServicesComponent/coreapplication_
obips1/instanceconfig.xml
```

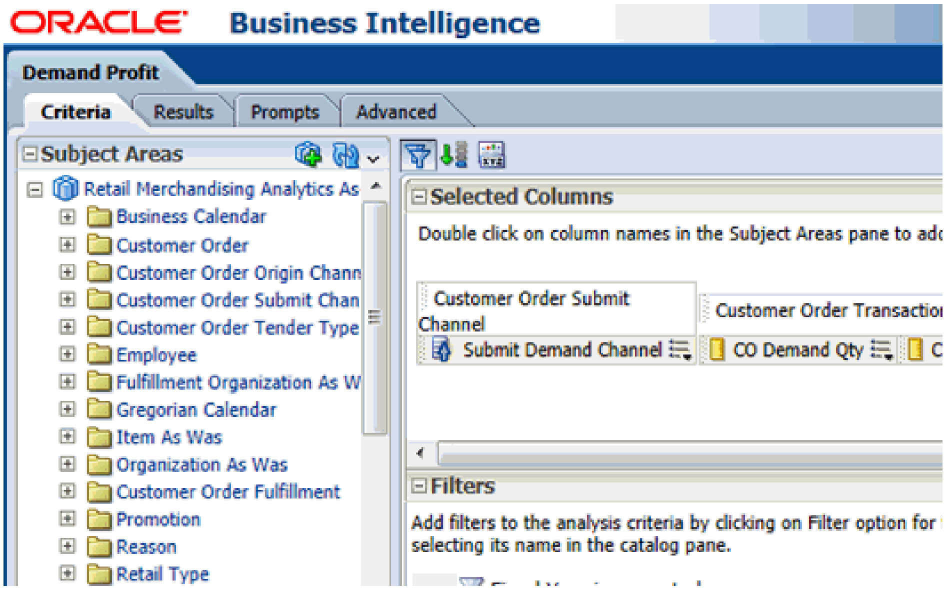
Add the following line in the above file and restart the Oracle BI EE instance.

```
<ActionLinks>
     <EnableADFContextualEvent>true</EnableADFContextualEvent>
 </ActionLinks>
```

This will enable the ADF Contextual Event action in Oracle BI EE. Verify that the ADF Contextual Event is enabled in Oracle BI EE using the following steps.

1. Edit any analysis and go to the Criteria tab.

Figure 6–11 Criteria Tab

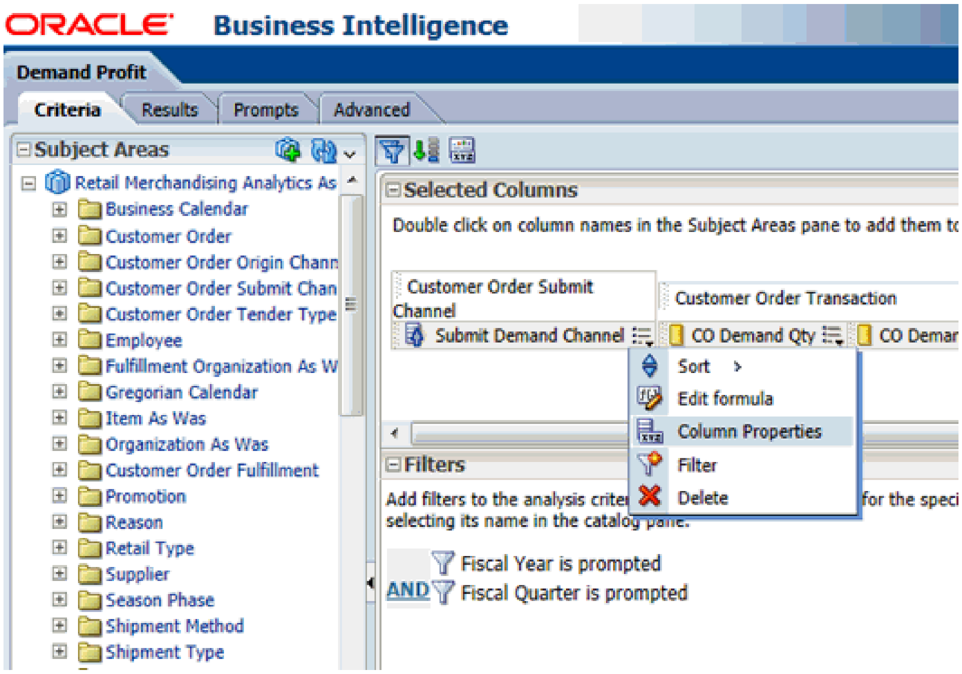

ORACLE Business Intelligence
Demand Profit
Criteria
Results
Prompts
Advanced
Subject Areas
Retail Merchandising Analytics As
Business Calendar
Customer Order
Customer Order Origin Chann
Customer Order Submit Chan
Customer Order Tender Type
Employee
Fulfillment Organization As W
Gregorian Calendar
Item As Was
Organization As Was
Customer Order Fulfillment
Promotion
Reason
Retail Type
Selected Columns
Customer Order Submit Channel
Customer Order Transactio
Submit Demand Channel
CO Demand Qty
Filters

Double click on column names in the Subject Areas pane to adc

Add filters to the analysis criteria by clicking on Filter option for
selecting its name in the catalog pane.

2. Edit the Column Properties of any column and go to the Interaction tab.

Figure 6–12 Column Properties



Double click on column names in the Subject Areas pane to add them tc

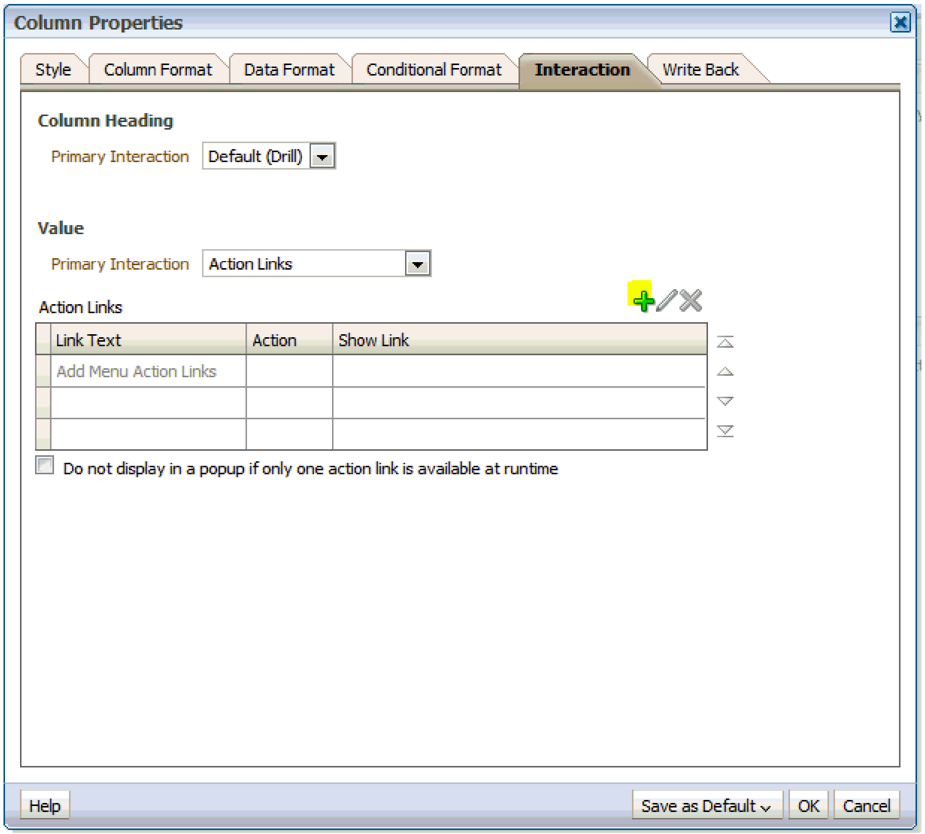
Add filters to the analysis criter[...] for the speci[...] selecting its name in the catalog pane.

Fiscal Year is prompted

AND Fiscal Quarter is prompted

3. In the Interaction tab select the Primary Interaction as Action Links for the Value of the column and click on the + icon.

Figure 6–13 Primary Interaction



Column Heading

Primary Interaction: Default (Drill)

Value

Primary Interaction: Action Links

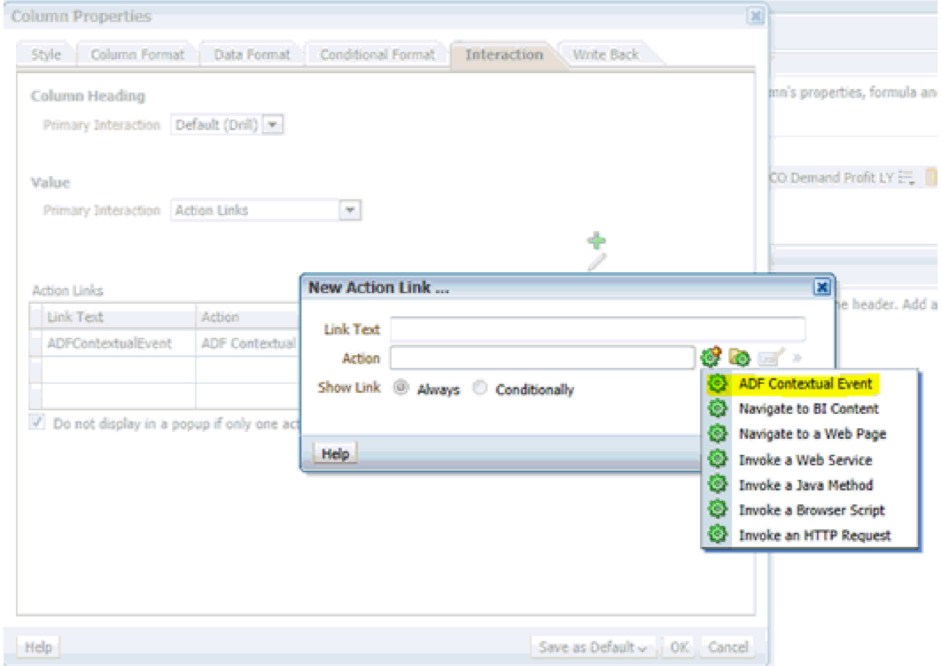
Action Links

Link Text	Action	Show Link
Add Menu Action Links		

☐ Do not display in a popup if only one action link is available at runtime

4. Verify that 'ADF Contextual Event' is available as a new Action Link item.

Figure 6–14 ADF Contextual Event



Column Properties

Column Heading
Primary Interaction: Default (Drill)

Value
Primary Interaction: Action Links

Action Links

Link Text	Action
ADFContextualEvent	ADF Contextual

☑ Do not display in a popup if only one act

New Action Link ...

Link Text:
Action:
Show Link: ● Always ○ Conditionally

Preparing the Custom Shared Library for Contextual Event Model File

This section describes how to prepare the Custom Shared Library so retailers will be able to add in-context launch item that will be launched from the Oracle BI EE report.

1. Perform the steps described in the section "Creating and Deploying a Custom Shared Library" in the Customizing Retail Applications chapter to create a Custom Shared Library workspace, generate a shared library WAR out of it, deploying the WAR, and associating the library to the Retail Application.
2. Obtain a copy of the OBIEEADFContextualEventModel.xml file where the in-context launch taskflows will be added.
3. Using JDeveloper, open the Custom Shared Library workspace in Developer Role.
4. Add the OBIEEADFContextualEventModel.xml file in the View-Controller project src directory, preferably under a subdirectory called custom.

   Example:

   ```
   View-Controller/src/custom/bi/OBIEEADFContextualEventModel.xml
   ```

   The file OBIEEADFContextualEventModel.xml is similar to the other navigation model xml files as it uses the same NavigationModel.xsd schema. The file will be used to define the taskflows that will be launched in-context from the Oracle BI EE reports. The contents of the file can be as follows.

   ```
   <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
   <NavigationDefinition id="Folder_1"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xmlns="urn:www.oracle.com:oracle.retail.apps.framework.navigation"

   xsi:schemaLocation="urn:www.oracle.com:oracle.retail.apps.framework.navigation
   classpath:oracle/retail/apps/framework/uishell/navigation/model/schema/Navigati
   ```

```
onModel.xsd">
  <Items>
    <Item id="biInContextLaunchAction"
       title="BIInContextLaunchFlow"
       shortDesc="This taskflow is an example taskflow which is launched from
an Oracle BI EE report in a new tab in Retail Application."
type="taskflow" visible="#{true}" keysList="#{payload.valueMap['&quot;Customer
Order Submit Channel&quot;.&quot;Submit Demand Channel&quot;']}"
targetTitle="#{adfBundle['RetailAppsJunitUiBundle'].BI_IN_CONTEXT_LAUNCH_FLOW}
- #{payload.valueMap['&quot;Customer Order Submit Channel&quot;.&quot;Submit
Demand Channel&quot;']}">
<url>/WEB-INF/oracle/retail/apps/framework/uishell/navigation/contentarea/flow/
BIInContextLaunchFlow.xml#BIInContextLaunchFlow</url>
      <Parameters>
<Parameter id="parameter1">#{payload.valueMap['"Customer Order Submit
Channel"."Submit Demand Channel"']}</Parameter>
<Parameter id="parameter2">#{payload.valueMap['"Customer Order Transaction"."CO
Demand Order Ct"']}</Parameter>
<Parameter id="parameter3">#{payload.valueMap['"Customer Order Transaction"."CO
Demand Qty"']}</Parameter>
<Parameter id="parameter4">#{payload.valueMap['"Customer Order Transaction"."CO
Demand Retail Amt LY"']}</Parameter>
<Parameter id="parameter5">#{payload.valueMap['"Customer Order Transaction"."CO
Demand Retail Amt"']}</Parameter>
<Parameter id="parameter6">#{payload.valueMap['"Customer Order Transaction"."CO
Demand Profit LY"']}</Parameter>
<Parameter id="parameter7">#{payload.valueMap['"Customer Order Transaction"."CO
Demand Profit"']}</Parameter>
      </Parameters>
    </Item>
  </Items>
</NavigationDefinition>
```

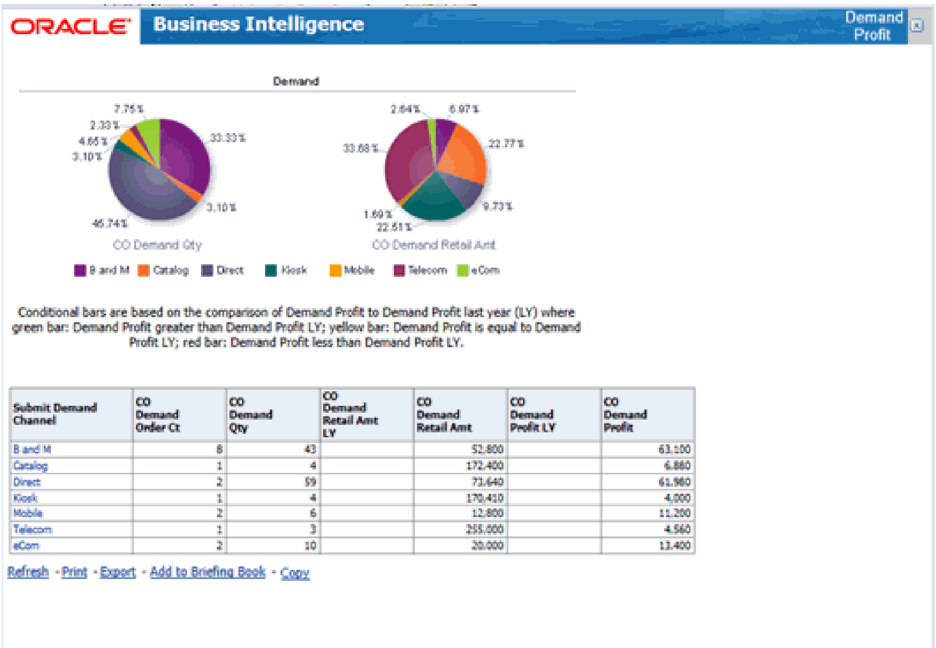
5. Regenerate the shared library WAR file from the workspace and redeploy the shared library. Shutdown and restart of the Retail Application and its shared library registry is required.

Enabling the Application Development Framework Contextual Event Action on the Target Column

In the above step, we deployed a custom navigation model file and defined a navigation item in the model file that will be launched from an action in the Oracle BI EE analysis. Now, let's try to enable the action in the Oracle BI EE analysis.

Let's consider an example. We have an analysis called 'Demand Profit' analysis as shown in the screenshot below. The analysis has a pie chart view and a table view.

Figure 6–15 Analysis Chart



Conditional bars are based on the comparison of Demand Profit to Demand Profit last year (LY) where green bar: Demand Profit greater than Demand Profit LY; yellow bar: Demand Profit is equal to Demand Profit LY; red bar: Demand Profit less than Demand Profit LY.

Submit Demand Channel	CO Demand Order Ct	CO Demand Qty	CO Demand Retail Amt LY	CO Demand Retail Amt	CO Demand Profit LY	CO Demand Profit
B and M	8	43		52,800		63,100
Catalog	1	4		172,400		6,880
Direct	2	59		73,640		61,980
Kiosk	1	4		170,410		4,000
Mobile	2	6		12,800		11,200
Telecom	1	3		255,000		4,560
eCom	2	10		20,000		13,400

When the user clicks on any row of the column 'Submit Demand Channel' in the table view of the above analysis, we want to open the taskflow defined in the file OBIEEADFContextualEventModel.xml. Also, we want to pass data for the user's selection in the analysis to the taskflow as input parameters.

First, step is to enable ADF Contextual Event action for the data of the column 'Submit Demand Channel'. Refer to the section "Enabling the Application Development Framework Contextual Event Action in Oracle Business Intelligence Enterprise Edition" to see how to enable ADF Contextual Event for this column. Note that the Action Link should be added to the Value of the column and not to the Column Heading.

When a user clicks a data cell containing the Oracle BI EE ADF Contextual Event action, the system generates an Oracle BI EE ADF Contextual Event action that has a qualified data reference (QDR) of the cell as its payload. For more details about QDR, please refer to the *Oracle Fusion Middleware Developer's Guide* for Oracle Business Intelligence Enterprise Edition at location

https://docs.oracle.com/middleware/11119/biee/BIEDV/embedding_adf.htm#BIEDV1214

The payload received does not inform the application, which taskflow in the application should be opened for the ADF Contextual Event action from Oracle BI EE. The Retail Application relies on certain hidden columns in the analysis for this information.

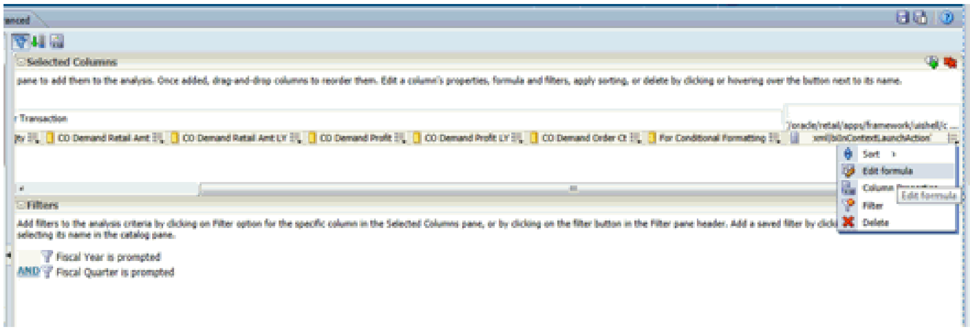
Modifying the Oracle Business Intelligence Enterprise Edition Analysis to Enable In-Context Launch

The analysis in Oracle BI EE should be modified for enabling In-Context Launch. In the example above, we will edit the 'Demand Profit' analysis and add a hidden column as follows:

1. Edit the analysis and go to the Criteria tab.

2. Drag any column and add it to the analysis.
3. Edit the Formula for the column.

Figure 6–16 Edit Formula



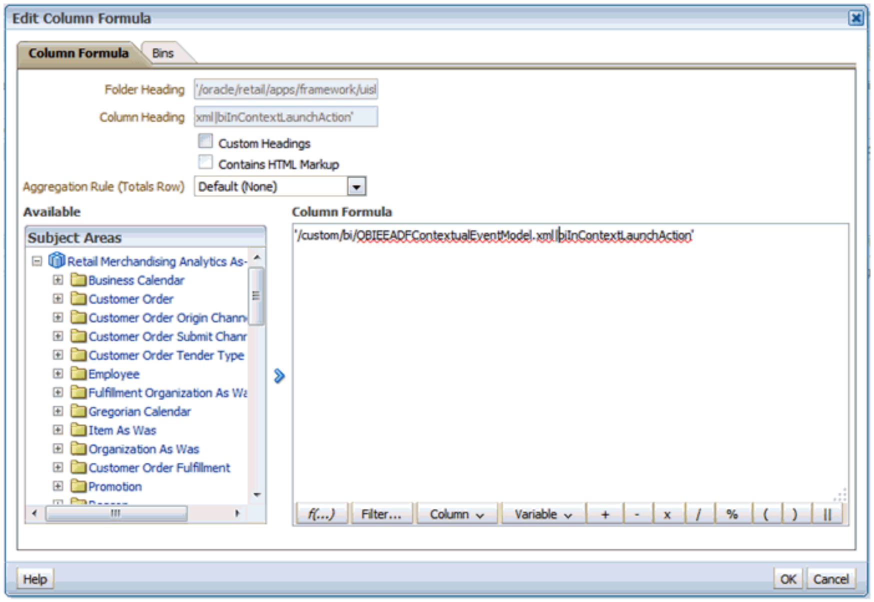
pane to add them to the analysis. Once added, drag-and-drop columns to reorder them. Edit a column's properties, formula and filters, apply sorting, or delete by clicking or hovering over the button next to its name.

Add filters to the analysis criteria by clicking on Filter option for the specific column in the Selected Columns pane, or by clicking on the filter button in the Filter pane header. Add a saved filter by click selecting its name in the catalog pane.

Fiscal Year is prompted
AND Fiscal Quarter is prompted

4. In the Edit Formula window, add a quoted text as follows.

Figure 6–17 Column Formula Text



Folder Heading '/oracle/retail/apps/framework/uisl

Column Heading xml|biInContextLaunchAction'

□ Custom Headings

□ Contains HTML Markup

Aggregation Rule (Totals Row) Default (None)

Column Formula

'/custom/bi/OBIEEADFContextualEventModel.xml|biInContextLaunchAction'

5. The quoted text will tell application, what to open when the action is clicked in the Oracle BI EE analysis.

   The text is in the format

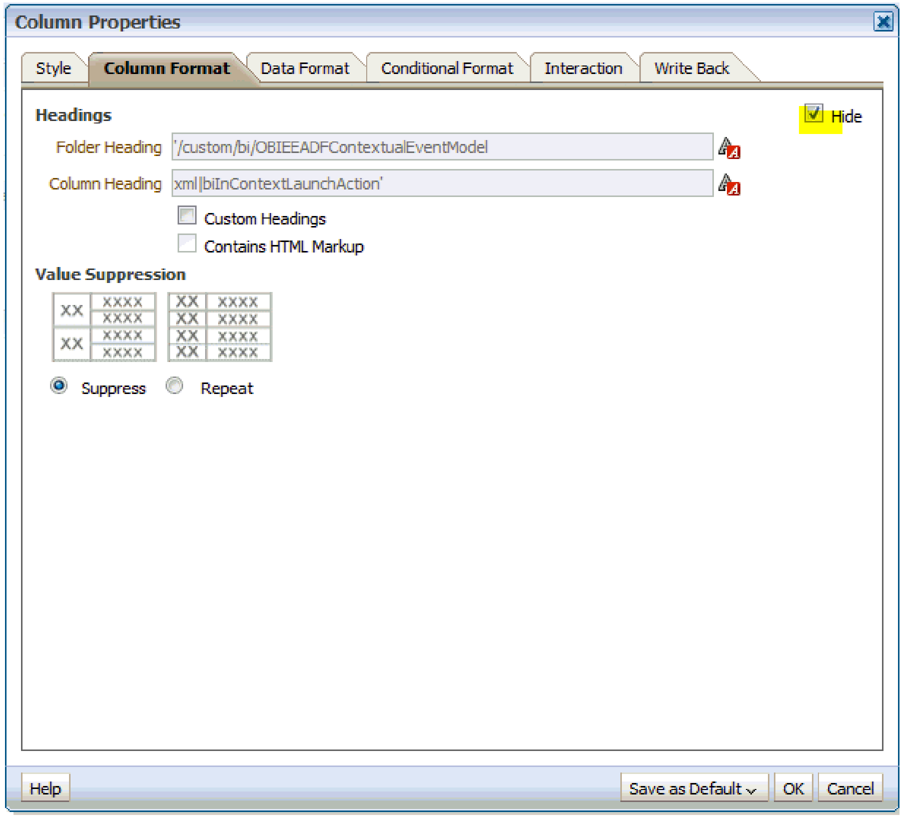
   '<path to the OBIEEADFContextualEventModel file>|<id of the item in the model file>'

   Example:

   '/custom/bi/OBIEEADFContextualEventModel.xml|biInContextLaunchAction'

   The column has to be marked hidden so that it is not displayed when the analysis is run. Edit the Column Properties, go to the Column Format tab and check Hide as shown in the screenshot below.

Figure 6–18 Select Hide



☑ Hide

Folder Heading: '/custom/bi/OBIEEADFContextualEventModel

Column Heading: xml|biInContextLaunchAction'

☐ Custom Headings

☐ Contains HTML Markup

● Suppress ○ Repeat

Now, we have added a column to the analysis which is hidden from display. When the user clicks on any row on the table view of this analysis, the column will be included in the QDR payload. The application will open the content requested in the column.

In the example,

'/custom/bi/OBIEEADFContextualEventModel.xml|biInContextLaunchAction'

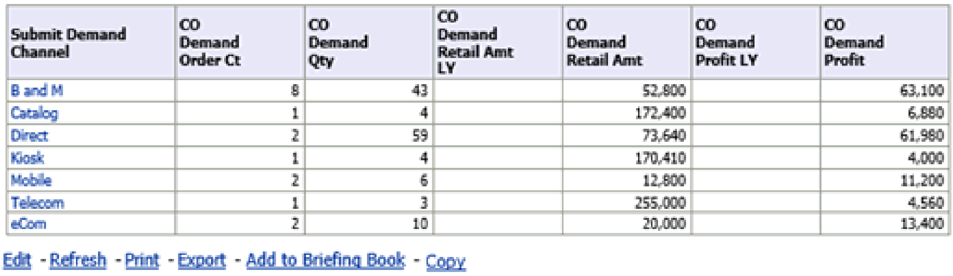
the application will open the navigation item with the Item id as biInContextLaunchAction in the file /custom/bi/OBIEEADFContextualEventModel.xml. The item id biInContextLaunchAction refers to the taskflow /WEB-INF/oracle/retail/apps/framework/uishell/navigation/contentarea/flow/BIInContextLaunchFlow.xml#BIInContextLaunchFlow as shown in the file "Preparing the Custom Shared Library for Contextual Event Model File". Ensure that the user has permissions to view the taskflow that you are trying to open.

Passing Parameters from the Oracle Business Intelligence Enterprise Edition Analysis to the Taskflow

When the user clicks on any column in the Oracle BI EE analysis that has the ADFContextualEvent action enabled, a qualified data reference (QDR) of the cell is received by the application as the payload. The application parses this payload and converts it into a format that could be easily used to populate the input parameters of the taskflow that is being opened in a new application tab.

The application will deliver a RetailAppsContextualEventPayload object whenever the ADFContextualEvent action is executed. The RetailAppsContextualEventPayload object has a map that stores the Oracle BI EE column name and its values which generated the contextual event.

Figure 6–19 Demand Channel



Submit Demand Channel	CO Demand Order Ct	CO Demand Qty	CO Demand Retail Amt LY	CO Demand Retail Amt	CO Demand Profit LY	CO Demand Profit
B and M	8	43		52,800		63,100
Catalog	1	4		172,400		6,880
Direct	2	59		73,640		61,980
Kiosk	1	4		170,410		4,000
Mobile	2	6		12,800		11,200
Telecom	1	3		255,000		4,560
eCom	2	10		20,000		13,400

Edit - Refresh - Print - Export - Add to Briefing Book - Copy

Let's assume the user clicked on 'Kiosk' row of the Submit Demand Channel column. The value Kiosk can be obtained using

#{payload.valueMap['"Customer Order Submit Channel"."Submit Demand Channel"']}

Similarly, the other data can be obtained using the format

#{payload.valueMap['<column formula in the analysis>']}

The value of CO Demand Qty for the row clicked by the user can be passed to the taskflow as input parameters using the following expression in the model file for the Item in the model.

<Parameter id="parameter3">#{payload.valueMap['"Customer Order Transaction"."CO Demand Qty"']}</Parameter>

Different Tabs for Each Set of Data

If you want to open different tabs for each row of data, you can use the keysList attribute to differentiate between tabs. You can use the targetTitle attribute to provide a unique name to each tab.

```
    <Item id="biInContextLaunchAction"
       title="#{adfBundle['RetailAppsJunitUiBundle'].BI_IN_CONTEXT_LAUNCH_FLOW}"
       shortDesc="#{adfBundle['RetailAppsJunitUiBundle'].BI_IN_CONTEXT_LAUNCH_
SHORT_DESC}"
       type="taskflow" visible="#{true}"
keysList="key=#{payload.valueMap['&quot;Customer Order Submit
Channel&quot;.&quot;Submit Demand Channel&quot;']}"
       targetTitle="#{adfBundle['RetailAppsJunitUiBundle'].BI_IN_CONTEXT_LAUNCH_
FLOW} - #{payload.valueMap['&quot;Customer Order Submit Channel&quot;.&quot;Submit
Demand Channel&quot;']}">

<url>/WEB-INF/oracle/retail/apps/framework/uishell/navigation/contentarea/flow/BII
nContextLaunchFlow.xml#BIInContextLaunchFlow</url>
      <Parameters>
        <Parameter id="parameter1">#{payload.valueMap['"Customer Order Submit
Channel"."Submit Demand Channel"']}</Parameter>
        <Parameter id="parameter2">#{payload.valueMap['"Customer Order
Transaction"."CO Demand Order Ct"']}</Parameter>
        <Parameter id="parameter3">#{payload.valueMap['"Customer Order
Transaction"."CO Demand Qty"']}</Parameter>
        <Parameter id="parameter4">#{payload.valueMap['"Customer Order
Transaction"."CO Demand Retail Amt LY"']}</Parameter>
```

```
        <Parameter id="parameter5">#{payload.valueMap['"Customer Order
Transaction"."CO Demand Retail Amt"']}</Parameter>
        <Parameter id="parameter6">#{payload.valueMap['"Customer Order
Transaction"."CO Demand Profit LY"']}</Parameter>
        <Parameter id="parameter7">#{payload.valueMap['"Customer Order
Transaction"."CO Demand Profit"']}</Parameter>
      </Parameters>
    </Item>
```

In the example above, if the user clicks on Kiosk and Mobile, each will open in a new application tab instead of using the same tab. The title of each tab will suffix the value of the column "Customer Order Submit Channel"."Submit Demand Channel" to differentiate between the tabs.

Pre-Processing the Data Before Opening the Content

It may be a requirement to pre-process the data that is received from Oracle BI EE before sending it to the application taskflow. For example, the application taskflow might require the data in certain data type which is different from what is received from Oracle BI EE. The Retail application provides a pre-processing hook for the applications to manipulate the data received from Oracle BI EE before sending it to the application taskflow. The applications can use the contentListener attribute of the Item element in the model file to invoke a pre-processing listener.

```
    <Item id="biInContextLaunchAction"
       title="#{adfBundle['RetailAppsJunitUiBundle'].BI_IN_CONTEXT_LAUNCH_FLOW}"
       shortDesc="#{adfBundle['RetailAppsJunitUiBundle'].BI_IN_CONTEXT_LAUNCH_
SHORT_DESC}"
       type="taskflow" visible="#{true}"
keysList="key=#{payload.valueMap['&quot;Customer Order Submit
Channel&quot;.&quot;Submit Demand Channel&quot;']}"
       targetTitle="#{adfBundle['RetailAppsJunitUiBundle'].BI_IN_CONTEXT_LAUNCH_
FLOW} - #{payload.valueMap['&quot;Customer Order Submit Channel&quot;.&quot;Submit
Demand Channel&quot;']}"

contentListener="oracle.retail.apps.framework.uishell.navigation.contentarea.liste
ner.OBIEEPreProcessListenerImpl">

<url>/WEB-INF/oracle/retail/apps/framework/uishell/navigation/contentarea/flow/BII
nContextLaunchFlow.xml#BIInContextLaunchFlow</url>
      <Parameters>
        <Parameter id="parameter1">#{payload.valueMap['"Customer Order Submit
Channel"."Submit Demand Channel"']}</Parameter>
        <Parameter id="parameter2">#{payload.valueMap['"Customer Order
Transaction"."CO Demand Order Ct"']}</Parameter>
        <Parameter id="parameter3">#{payload.valueMap['"Customer Order
Transaction"."CO Demand Qty"']}</Parameter>
        <Parameter id="parameter4">#{payload.valueMap['"Customer Order
Transaction"."CO Demand Retail Amt LY"']}</Parameter>
        <Parameter id="parameter5">#{payload.valueMap['"Customer Order
Transaction"."CO Demand Retail Amt"']}</Parameter>
        <Parameter id="parameter6">#{payload.valueMap['"Customer Order
Transaction"."CO Demand Profit LY"']}</Parameter>
        <Parameter id="parameter7">#{payload.valueMap['"Customer Order
Transaction"."CO Demand Profit"']}</Parameter>
      </Parameters>
    </Item>
```

The listener implementation class should implement the `oracle.retail.apps.framework.uishell.navigation.contentarea.listener.ContentListener` interface and provide an implementation to the execute() method of this interface. The execute method will be invoked before the application taskflow is opened. In the example below, the data type of `"Customer Order Transaction"."CO Demand Profit"` is being changed from data type Integer to String as the application taskflow expects this value to be a String type.

```
package oracle.retail.apps.framework.uishell.navigation.contentarea.listener;

import java.io.Serializable;

import java.util.HashMap;
import java.util.Map;

import
oracle.retail.apps.framework.publicui.contextualevent.RetailAppsContextualEventPay
load;
import oracle.retail.apps.framework.uicomponents.util.ADFFacesUtil;

public class OBIEEPreProcessListenerImpl implements ContentListener, Serializable
{
  @SuppressWarnings("compatibility")
  private static final long serialVersionUID = 1L;

  public OBIEEPreProcessListenerImpl() {
    super();
  }

  @Override
  public void execute() {
    //get the original payload from the request
    RetailAppsContextualEventPayload payload =

(RetailAppsContextualEventPayload)ADFFacesUtil.getRequestAttribute("payload");
    //original value is of type Integer
    Integer demandProfit =
      (Integer)payload.getValue("\"Customer Order Transaction\".\"CO Demand
Profit\"");

    //modify the value to be of type String
    Map valueMap = payload.getValueMap();
    Map modifiedValueMap = new HashMap();
    modifiedValueMap.putAll(valueMap);
    modifiedValueMap.put("\"Customer Order Transaction\".\"CO Demand Profit\"",
              String.valueOf(demandProfit));

    //put the new payload object in the request
    RetailAppsContextualEventPayload modifiedPayload =
      new RetailAppsContextualEventPayload(modifiedValueMap, null);
    ADFFacesUtil.setRequestAttribute("payload", modifiedPayload);
  }
}
```

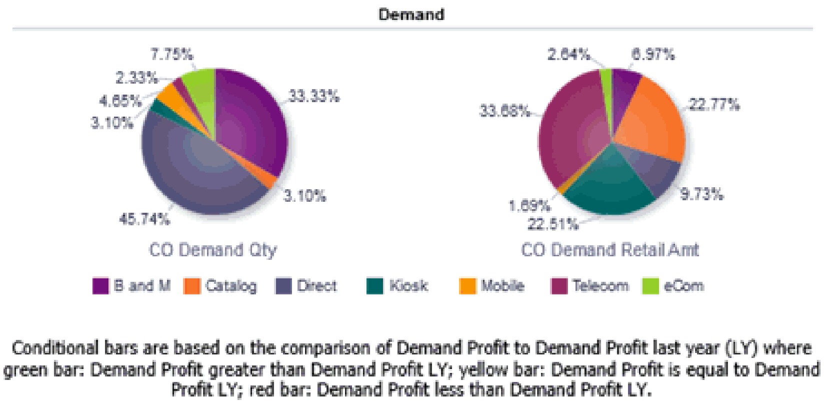
In-Context Launch from the Chart View of the Analysis

In-Context Launch from the chart view of the analysis is different from the table view. When the hidden column is added as described in the section "Modifying the Oracle

Business Intelligence Enterprise Edition Analysis to Enable In-Context Launch", it becomes available in the QDR payload in case the analysis has a table view, which enables the Retail Applications Framework to launch the content requested in the hidden column.

If the analysis has a chart view, then the hidden column has to be explicitly added in the chart view to be included in the payload.

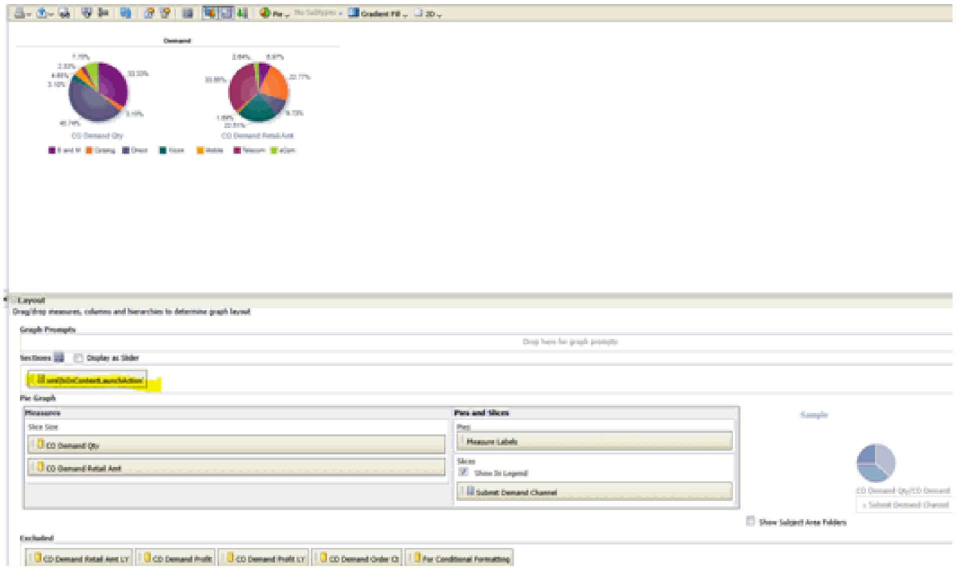
Figure 6–20 Demand Chart View



Conditional bars are based on the comparison of Demand Profit to Demand Profit last year (LY) where green bar: Demand Profit greater than Demand Profit LY; yellow bar: Demand Profit is equal to Demand Profit LY; red bar: Demand Profit less than Demand Profit LY.

In the pie chart above, if the user clicks on B and M, the payload will not include the hidden column by default. In order include the hidden column in the payload, we will edit the chart and add the column to Sections in the chart.

Figure 6–21 Include Hidden Column



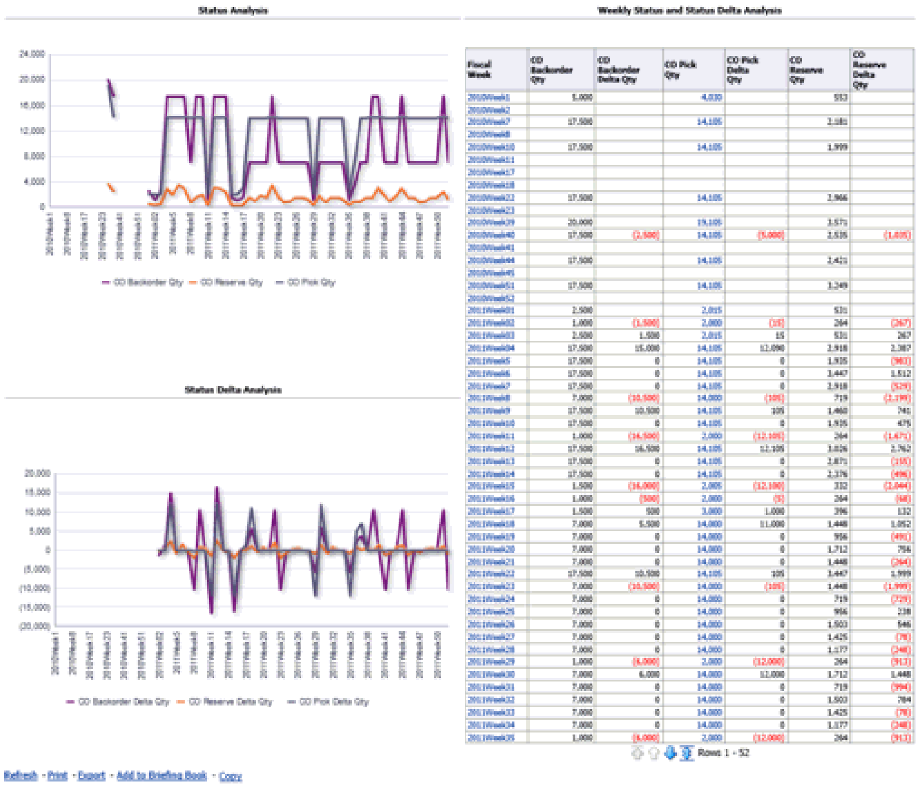
Adding this column to the Sections of the chart will not have any impact on the chart as the column has the same data for each row. However, this will include this column in the payload received from the chart, enabling the Retail Application to open the content requested in the payload.

There is an issue related to the ADFContextualEvent action generated from the chart view of an analysis. The payload only includes the dimensions and not the measures values. In the example above, only the dimension Submit Demand Channel will be included in the payload. The measure CO Demand Qty or CO Demand Retail Amt will not be included. So, the measure values cannot be passed to the taskflow as input parameters, if the ADFContextualEvent is generated from an Oracle BI EE chart.

Opening Different Taskflows from the Same Analysis

It may be a requirement that you want to open one taskflow from one column in the analysis and a different taskflow from a different column. Let's consider an example.

Figure 6–22 Taskflow Column



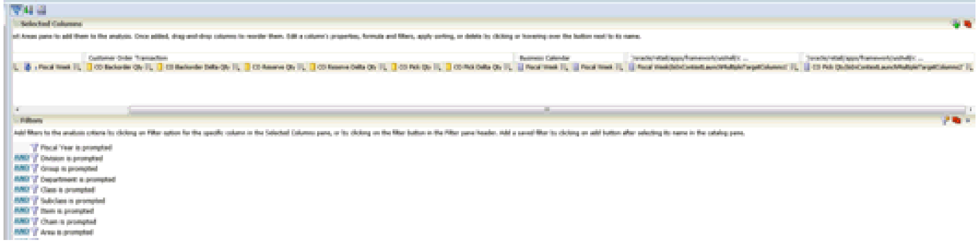
Weekly Status and Status Delta Analysis

Fiscal Week	CO Backorder Qty	CO Backorder Delta Qty	CO Pick Qty	CO Pick Delta Qty	CO Reserve Qty	CO Reserve Delta Qty
2010Week1	5,000		4,030		553	
2010Week2						
2010Week7	17,500		14,105		2,181	
2010Week8						
2010Week10	17,500		14,105		1,999	
2010Week11						
2010Week17						
2010Week18						
2010Week22	17,500		14,105		2,966	
2010Week23						
2010Week39	20,000		19,105		3,571	
2010Week40	17,500	(2,500)	14,105	(5,000)	2,535	(1,035)
2010Week41						
2010Week44	17,500		14,105		2,421	
2010Week45						
2010Week51	17,500		14,105		3,249	
2010Week52						
2011Week01	2,500		2,015		531	
2011Week02	1,000	(1,500)	2,000	(15)	264	(267)
2011Week03	2,500	1,500	2,015	15	531	267
2011Week04	17,500	15,000	14,105	12,090	2,918	2,387
2011Week5	17,500	0	14,105	0	1,935	(983)
2011Week6	17,500	0	14,105	0	3,447	1,512
2011Week7	17,500	0	14,105	0	2,918	(529)
2011Week8	7,000	(10,500)	14,000	(105)	719	(2,199)
2011Week9	17,500	10,500	14,105	105	1,460	741
2011Week10	17,500	0	14,105	0	1,935	475
2011Week11	1,000	(16,500)	2,000	(12,105)	264	(1,671)
2011Week12	17,500	16,500	14,105	12,105	3,026	2,762
2011Week13	17,500	0	14,105	0	2,871	(155)
2011Week14	17,500	0	14,105	0	2,376	(496)
2011Week15	1,500	(16,000)	2,005	(12,100)	332	(2,044)
2011Week16	1,000	(500)	2,000	(5)	264	(68)
2011Week17	1,500	500	3,000	1,000	396	132
2011Week18	7,000	5,500	14,000	11,000	1,448	1,052
2011Week19	7,000	0	14,000	0	956	(491)
2011Week20	7,000	0	14,000	0	1,712	756
2011Week21	7,000	0	14,000	0	1,448	(264)
2011Week22	17,500	10,500	14,105	105	3,447	1,999
2011Week23	7,000	(10,500)	14,000	(105)	1,448	(1,999)
2011Week24	7,000	0	14,000	0	719	(729)
2011Week25	7,000	0	14,000	0	956	238
2011Week26	7,000	0	14,000	0	1,503	546
2011Week27	7,000	0	14,000	0	1,425	(78)
2011Week28	7,000	0	14,000	0	1,177	(248)
2011Week29	1,000	(6,000)	2,000	(12,000)	264	(913)
2011Week30	7,000	6,000	14,000	12,000	1,712	1,448
2011Week31	7,000	0	14,000	0	719	(994)
2011Week32	7,000	0	14,000	0	1,503	784
2011Week33	7,000	0	14,000	0	1,425	(78)
2011Week34	7,000	0	14,000	0	1,177	(248)
2011Week35	1,000	(6,000)	2,000	(12,000)	264	(913)

If the user clicks on any Fiscal Week value in the table above we want to open a taskflow say Taskflow1 in a new application tab. If the user clicks on CO Pick Qty, we want to open another taskflow say Taskflow2 in a new application tab.

The Retail Application provides this capability to open different content from different columns in the analysis by using multiple hidden columns. In this example, we will use two hidden columns as shown in the screenshot below.

Figure 6–23 Two Hidden Columns



Fiscal Year is prompted
AND Division is prompted
AND Group is prompted
AND Department is prompted
AND Class is prompted
AND Subclass is prompted
AND Item is prompted
AND Chain is prompted
AND Area is prompted

The first hidden column is

'/custom/bi/OBIEEADFContextualEventModel.xml|biInContextLaunchAction|"Business Calendar"."Fiscal Week"|biInContextLaunchMultipleTargetColumns1'

The second hidden column is

'/custom/bi/OBIEEADFContextualEventModel.xml|"Customer Order Transaction"."CO Pick Qty"|biInContextLaunchMultipleTargetColumns2'

When the user clicks either the Fiscal Week or the CO Pick Qty, the ADFContextualEvent action will generate the QDR of the cell as a payload. The payload will include both hidden columns that we added to the analysis. The Retail Application will check which column is clicked; it will then verify in the list of hidden columns which action to invoke for that column. So, in the above example, if the user clicks `2011Week32` as the Fiscal Week, the Retail Application will open the content defined for the Item id `biInContextLaunchMultipleTargetColumns1` in the model file `/custom/bi/OBIEEADFContextualEventModel.xml`.

The format for defining hidden column in such a case is

```
'<path to the OBIEEADFContextualEventModel file>|<Column name in the
analysis>|<id of the item in the model file>'
```

7

Troubleshooting

This chapter covers troubleshooting.

Oracle Business Intelligence Enterprise Edition Integration

This section covers the Oracle BI EE integration.

The Oracle Business Intelligence Enterprise Edition Dashboard Displays an Error Message 'An error occurred retrieving the Business Intelligence content. Please contact your System Administrator'

Go through the following steps to troubleshoot this issue.

1. Login directly to Oracle BI EE and verify if it is up and running. If the Oracle BI EE is down you may encounter the above error.
2. The Retail Application uses the BI Presentation Services connection to connect to Oracle BI EE. Ensure that Oracle BI EE is prepared for the Presentation Services connection. Refer to the chapter Embedding Business Intelligence Objects in ADF Applications of the Fusion Middleware Developer's Guide for Oracle Business Intelligence Enterprise Edition.

   The link to the *Fusion Middleware Developer's Guide for Oracle Business Intelligence Enterprise Edition* is

   `https://docs.oracle.com/middleware/11119/biee/BIEDV/embedding_adf.htm`
3. Ensure that the BIConnection attributes in the Retail Application to connect to Oracle BI EE are correct. Refer to the section "Modifying the BIConnection Attributes at Runtime" in the Oracle Business Intelligence Enterprise Edition Integration in Retail Applications chapter to change the BIConnection attributes post install.
4. Ensure that the user who logs into the application exists in Oracle BI EE and has permissions to view the Oracle BI EE content requested.

Troubleshooting In-Context Launch

If the In-Context launch does not work after enabling the ADF Contextual Event action in the Oracle BI EE analysis and adding the hidden column as documented above, then you should enable FINER logging for oracle.retail.apps.framework.bi package.

The log file should log the QDR payload received from Oracle BI EE and the actions executed by the Retail Application code based on the QDR.

Contextual Area Displays a White Screen

The Contextual Area displays the Oracle BI EE content using inline frames (IFrames). Ensure that IFrames are enabled in Oracle BI EE.

8
Functional Design

This chapter discusses the various functional aspects of Oracle Retail Allocation. The chapter provides the following:

- A functional overview of the system, along with its features and corresponding functional assumptions.
- The sources of data used by rules to determine gross need.
- A description of the three possible methods of closing allocations.

Functional Features and Assumptions

This section covers what Allocation does, different item sources, and the functional assumptions that are made during the creation of ASN-based allocations, item-location combination validation, calculation multiple, what-if allocations, weight and date selection, and proportional allocations.

> **Note:** Oracle Retail Allocation does not offer logic to support the following:
>
> - Fashion items set up as grandparent > parent > children (items)
> - Buyer packs where the packs are not inventoried and only the components are inventoried

Overview: What Does Oracle Retail Allocation Do?

An Allocation application should enable retailers to make important decisions as close as possible to the time the product must be sent to the destination stores or warehouses. A critical link in the supply chain process, the allocation process presents the final chance to distribute products successfully. The challenges facing retailers for allocating product are the same, whether they sell fashion items, groceries or hard lines. Merchants want an efficient, accurate method of translating their merchandise plans into location level allocations. Effectively allocating products is a critical step in product life cycle management and the last chance the retailer has to get the right product to the right location in the right quantity.

Oracle Retail Allocation enables retailers to take advantage of the most current, up-to-date sales and inventory information. The application also has the flexibility to allow allocations to be calculated months in advance for vendor commitment purposes.

Oracle Retail Allocation has been designed to address the following challenges (among others) related to the correct allocation of product:

- How to put a variety of merchandise plans into action.
- How to allocate products to support diverse marketing efforts and selling profiles.
- How to effectively and accurately allocate products without increasing headcount while continuing to grow the business.
- How to streamline the training process for allocators.

If these challenges are not met, the wrong product can be sent to the wrong location in the incorrect quantity at the wrong time. The net result is higher markdowns and lower profits.

Oracle Retail Allocation allows multiple parameters to be selected when creating an allocation. The system determines store or warehouse level need based on metrics that fit the product, location characteristics and product life cycle. The result is allocations based on individual location need, the key to maximizing sales and profits.

In Oracle Retail Allocation, the retailer has the option of executing a sales plan, receipt plan, history or forecast at any level of the hierarchy. The retailer can allocate a collection of products using a class plan or allocate one item using its history. Oracle Retail Allocation has the functionality to create and reuse 'templates' to save time and produce consistent results.

Oracle Retail Allocation can react to current trends. The system has sophisticated rules that can compare current selling to a plan and create a forecast on which to base an allocation. Oracle Retail Allocation provides the ability to both allocate in advance to give vendor commitments and allocate at the last minute, utilizing up-to-date sales and inventory information to determine individual location need. Some key features of the application include:

- Any PO that a user creates can be previewed through the front end.
- In order to increase the efficiency of the allocation process, Oracle Retail Allocation has the ability to split allocations. By splitting an allocation, the user has the option of selecting the product hierarchy and location combinations that the user would like to remove from the original allocation.
- Oracle Retail Allocation can automatically respect the existence of an item location record in RMS.
- The user is allowed to copy an existing allocation, resulting in the creation of a new one with the same item/location combinations and other parameters as the source.

Item Sources

Item sources represent the physical inventory that Oracle Retail Allocation can allocate. The system allows the retailer to allocate based upon the following:

- Advanced shipping notices (ASN) or Advanced Shipment Notifications (ASN) are batch communications that inform a retailer when to expect a certain quantity of order inventory. ASN quantities are received closer to the time of arrival at the distribution center. Allocations based upon ASN quantities allow retailers to account for purchase order shortages or overages as a part of their Oracle Retail Allocation process.
- Transfers
- Bills of lading (BOL)
- Purchase orders
- Warehouse-sourced inventory

- Approved allocations to a warehouse

The user is given more access to and control of existing transactions because of these item sources, which increases supply chain efficiencies. As a result, the next inventory movement can be communicated to the distribution center before the inventory arrives and is put on the shelf as warehouse stock.

How Need is Determined

The logic of Oracle Retail Allocation is based on establishing need at the item/location level. The user determines need by choosing a rule and rule modifiers and then setting optional quantity limits. The system accesses gross need values provided. The application applies constraints to the data, such as on-hands and on-order, and determines the net need. At this point, the algorithm determines how to spread the available inventory across all of the net needs for each location, and the allocation is created.

When attempting to allocate with a store that has a zero need, the system uses the sister-store data instead of allocating no product. The Sister Store Setup system option setting file allows the retailer to determine whether to use the sister store need whenever the need is calculated as zero or only when no need records exist for each location.

Oracle Retail Allocation retrieves most data in real-time from the Oracle Retail Merchandising System (RMS). Oracle Retail Allocation only requires visibility to approved items and purchase orders and transfers. See Functional and Technical Integration in the Functional and Technical Integration chapter for integration information.

In sum, Oracle Retail Allocation determines the needs of each individual store or warehouse at the item-location level through the following capabilities:

- The application sorts through large quantities of data, such as sales history, current on-hand, and store volume groups.
- The application applies user-established rules, rule modifiers, and optional quantity limits.
- The application utilizes complex algorithms that can determine gross need for large volumes of locations and products, using real time data.
- Presentation Minimums at the item location level can be respected.

Due to the importance of balancing the need to maintain appropriate store/warehouse presentation level and effectively allocate to stores' or warehouses’ inventory needs, Oracle Retail Allocation can access item/location planogram data and dictate to the algorithm the smallest amount to allocate a given item/location.

Weight and Date User Selection

> **Note:** In the Allocation application and in this document, the terms 'rules' and 'policies' are used interchangeably.

The premise of Oracle Retail Allocation is to establish need at the item/location level via policies and policy modifiers. The following eight different rules are available:

- Sales history
- Forecast

- Plan
- Receipt plan
- History and plan
- Plan re-project
- Corporate
- Manual

Although these rules are detailed, occasionally the user needs to base allocations upon like items. The User Merchandise Level Selection screen allows retailers to select any combination of like data on which to base allocations. The user may choose a merchandise hierarchy level, a combination of merchandise hierarchy levels, individual items or merchandise hierarchy levels combined with individual items. Each combination of data may be weighted. For example, an allocation may require the input of Subclass Z's sales history to be weighted at 50% and item A's sales history to be weighted at 75%. The values selected by the user are applied to each item on the allocation.

'What if' On Hand

The 'what if' source allows the user to create hypothetical allocations. 'What if' allocations provide users the flexibility to create purchase orders based upon the allocation calculated quantities. The process is exactly the same as a regular allocation except that the user starts with an infinite available number of the selected product or a user specified quantity.

Purchase Order Addition

In Oracle Retail Allocation, the user has the ability to allocate many items on a single allocation. This ability is dependent on what rules are utilized when gathering an item, or set of items' need values. The user has the ability to add quantities to existing purchase orders (POs) from the front end screen dedicated to 'what if' summary data. Valid items added to the PO must not previously exist on the PO. If the item already exists on the PO, the allocator should modify the quantities within the ordering window.

The purchase order addition functionality is achieved by calling an RMS stored procedure that handles the PO Creation Request. The Oracle Retail Merchandising System owns this code and the standards it enforces regarding PO creation. The current API validation includes: location exists, item exists, supplier exists, item/supplier exists, item/supp/country exists, item is orderable and checks for dept_level_orders which is a system option in RMS.

Holdback Quantity

The holdback quantity is the quantity that the user does not want to allocate. The holdback quantity is subtracted from the available quantity of each item to calculate the available percent to total quantity. The available percent of the total quantity is applied to the gross need to determine the final allocated quantity. The following example illustrates applying the holdback quantity to a sellable staple simple pack in spread demand mode:

Table 8–1 Holdback Quantity Example

Dept	1	Hardware
Class	12	Tools

Table 8–1 (Cont.) Holdback Quantity Example

Subclass	123	Hand Tools

Table 8–2 Different Packs

Pack	Description	Warehouse Quantity (Pack)	Holdback Qty	Net Available Warehouse Qty	Percent of Total
1	Hammer (5 units)	500	75	425	425/900=47.2222%
2	Screwdriver (10 units)	350	75	275	225/900=25%
3	Handsaw (15 units)	250	0	250	250/900=27.7778%
Total quantity on hand--------->		1100		900	

For sellable packs, the transaction level is the sellable unit, so sales history is recorded at the pack level as shown in the Dept Sales column of the table below. For sellable packs, the store owns the product at component level. When a pack is sold, the components are deducted from the stock on hand.

Table 8–3 Dept Sales and On Hands for Sellable Packs

Rule		History	
Level		Dept 1	
Date Ranges		Last 4 Weeks	
Available Quantity		1500	
RLOH		N/A	
Store	Pack	Dept Sales	On Hands
999	1	3967	125
	2	3967	140
	3	3967	130
	Dept Totals	3967	

On hand for Sellable Packs are not recorded at the pack level through RMS, only the pack components have on hand. Thus, on hand for sellable packs = 0. Since the final quantity is pack quantity, the calculations do not go through any pack optimization process in the calculation engine.

Table 8–4 The Final Calculations

Store No	Pack	Gross Need	OH (total of the level)	Net Need	Adjusted Need
999	Pack 1	3967	0	3967	3967*47.2222%= 1873
	Pack 2	3967	0	3967	3967*25%=992
	Pack 3	3967	0	3967	3967*27.7778%=1102

Allocation Approval Process

The process of approving an allocation is asynchronous. The functionality is consistent with the calculation process and allows the user to work on other allocations while one is being submitted, reserved or approved. In addition, the validation ensures that all inventory quantities are up to date at the time approval occurs, including the item source quantities. This validation prevents the system from creating an allocation against quantities that were available at the time of calculation but have since been claimed by another allocation, process or system.

Functional Assumptions

This section covers assumptions made during ASN-based allocations, item-location combination, calculation multiple, what-if allocations, weight and date selection, and proportional allocations.

Advanced Shipping Notice-Based Allocation Assumptions

Oracle Retail Allocation item sources are not individually selectable. A previously generated allocation may be selected indirectly by being included in a BOL portion of inventory.

Item-location Assumptions

The ITEM_LOC table is where item location relationships are held and maintained from the Oracle Retail Merchandising System.

Calculation Multiple Assumptions

- The system assumes that all of the items under a parent have the same Store Order Multiple (SOM) as the parent. In other words, the SOM cannot differ by item under a parent.
- The ability to calculate an allocation based upon one multiple and create a PO based upon another multiple may create a discrepancy between the total amount the allocation calculated initially, and the amount of inventory included in the purchase order.
- Promotional buy rounding issue: The use of the order case size as the case multiple for PO sourced allocations creates a scenario where rounding at the inner causes packaging discrepancies. Note that when a purchase order is the source of an allocation, the suppliers case pack size may be different than the pack size defined in the RMS Item data model. Because the RMS Purchase Order screen does not hold a defined value for an inner size, the inner defined for the item may not evenly round into the case on the purchase order.

What if

- Front-end PO location selection only has an effect on the PO to location when the user is creating a PO with a PO type of warehouse or cross dock, or updating a PO with a PO type of warehouse. For all other types of PO actions, the value has no relevance.

> **Note:** In a typical Cross Dock scenario, if the default warehouse for a store is also present as a destination warehouse on the What If allocation, the user will receive a popup while trying to raise the PO that the same warehouse cannot be specified as a source and a destination warehouse in the same allocation and hence such a cross dock allocation cannot be created. The PO creation will be carried out. The user will then need to manually create an allocation using this PO as its source.

- The major assumption associated with this functional area is that basing the allocation quantity on future available on hand amounts assumes that the inventory that is expected within the warehouse arrives and that the business user creates a separate allocation to move the inventory to the stores.
- 'What if' allocations utilize the primary supplier's primary origin countries' inner, case and pallet size only. If a retailer wants to use the window to create a PO to a supplier that does not have the same multiple as the primary supplier, the functional recommendation is to calculate the allocation and create the order using an 'Each' multiple. The multiple can then be adjusted within the merchandising system.

Weight and Date User Selection Assumptions

- The many-to-one functionality is available to users when using automatic rule types, except for corporate rules and manual.
- The system allows the user to add the same item or merchandise hierarchy level twice. This is intentional because various weights may be applied. Users are responsible for adjusting the weight appropriately.

Proportional Allocation Assumption

The system assumes that a proportional allocation does not contain more than 10,000 units going to one store. The 10,000-unit value is a hard limit, and if exceeded, an infeasible solution arises in the system's algorithm.

For both Simple and Spread Demand, the application should consider the threshold value as the first validation for allocating quantity to the store or warehouse. This is irrespective of whether the MIN or MAX values are mentioned.

> **Note:** If the "Need is" set to proportional, and the threshold value has been mentioned, allocation happens only if the store's net need is greater than the threshold value. If the store's net need is less than the threshold value the store is not allocated any quantity. After these validations occur, the allocation passes through the calc engine and the allocated quantity may be greater than the threshold value.

Scheduled Allocation Constraints

Some of the constraints the user must follow while scheduling allocations are:

- The parent/children relationship is one parent to many children of the same parent. The parent allocation cannot be a child allocation for another parent; a parent allocation cannot be a parent to another parent allocation.
- A parent allocation can only be deleted if all of the following is satisfied:

  - The current date is at least one day after the scheduled end date.
  - All child allocations of the parent must be closed or deleted.
  - If there is at least one child allocation that is not closed or deleted or if the end date is not met, the user cannot delete the parent allocation. The following error message is displayed "Allocation cannot be deleted until after the scheduled End Date and all its Child Allocation(s) are 'Deleted' and/or 'Closed'." In order to delete this parent, the user must go into the parent allocation to change the end date and also change the status of the child or children allocation(s) to 'deleted.'

- It is recommended to have no more than 10 items in a scheduled allocation.

Batch Job Considerations for Scheduled Allocations

The user has to consider the following points while running the batch for scheduled allocations:

- Batches can be run by external scheduling system such as APPWORX or a simple UNIX CRON job.
- If the server is down and restarted outside the client-designated window, scheduled allocations for that day are not created.
- The Schedule Allocation batch must be run after RMS updates the sales and on hand data.
- The Daily Cleanup batch process deletes the data from the temporary tables on a daily basis. Run this batch immediately after you run the Schedule Allocation batch.
- There should be a two hour window in which the child allocations can be generated. The user has to define this two hour window and this is against the system (server) date and timestamp that would apply to all users regardless of time zone.
- Whatever may be the status of the parent allocation, the batch process runs if the parent allocation has to generate a child allocation on that day. Despite having a status of 'Schedule Error' the parent allocation runs because the error may have occurred in the past (when the previous batch was run), and that situation may not be applicable for subsequent batches. The error may have occurred because of insufficient inventory, but by the next batch run, the warehouse may have received additional goods to meet inventory criteria, so the previous error is no longer valid.
- If the user tries to run the batch process a second time on the same day that the user deleted the earlier created child allocation, the batch process does not pick up the parent allocation, and creates the second child. The user has to either manually create an allocation or wait for the next scheduled batch run.

Additional Validations for Scheduled Allocation

Children allocations are not created if the parent allocation has errors. The child allocation is not created for the parent allocation, if the following validations fail:

- At least one item has a valid item/store or item/warehouse association. For example: If two items and two stores are selected in the Parent Allocation, then at least one item should have association with at least one store.
- At least one store is not closed and has a valid Warehouse-Store relationship when the “Enforce Supply Chain” check box is selected.

- If the Parent Allocation contains fashion items, then all the fashion items should have Size Profile information available for all the stores if the 'Enable Size Profile' = True in the Properties File on the System Properties tab of the system options screen.
- If at least one item has the warehouse available quantity greater than the minimum available quantity specified on the “Item Review” screen of the application, the child allocation is created with the status as 'Submitted'. This status overrides the status specified by the user. If all items do not have the warehouse available quantity greater than the minimum available quantity, then no child allocation is created for that parent on that day. The status of the parent allocation will be set to 'Schedule Error'.
- For a warehouse to warehouse ranging checks for the Enforce Supply Chain checkbox is in the checked state. The destination warehouse has s source method set to Warehouse and the source warehouse of the allocation set of items being allocated in order to receive goods from the warehouse location. If that is not set, then the default warehouse column will be checked next. If there is no source warehouse defaulted for the destination warehouse in either of these places, the warehouse would be listed at the item/location level in the Item Location Exclusions screen with the Reason Code set to Default Warehouse Missing.

Allocation Status

Significant functionality is attached to the status numbers in the system. The tables below reflect the status column on the ALC_ALLOC table.

Table 8–5 Visible Through the User Interface

Status	Description	Status #
Worksheet	The allocation is in the initial stages of creation or it is being updated by the user.	0
Submitted	The allocation has been submitted successfully.	1
Approved	The allocation has been approved successfully. The allocation records have been sent to the merchandising system and the other downstream applications.	2
Processed	The warehouse system has started executing this allocation. The allocation cannot be updated.	3
Closed	The allocation has been executed by the warehouse. The inventory movement is complete.	4
Cancelled	The allocation has been cancelled by an external system. The allocation cannot be updated.	5
Reserved	The allocation has been reserved successfully. The allocation records have been sent to the merchandising system.	6

Table 8–5 (Cont.) Visible Through the User Interface

Status	Description	Status #
PO Created	This status exists for 'what if' allocations only. A purchase order has been created within this 'what if' allocation. The user cannot edit anything within the allocation except creating or updating additional purchase orders for other items in the allocation for which no order has been created so far.	10
Scheduled	The allocation is scheduled. This status is not dependant on whether the parent allocation is successfully validated or not. It is a static field and does not get updated. Child allocations display the status selected in the Auto Schedule screen of the parent allocation.	11

Table 8–6 Not Visible Through the GUI

Status	Description	Status #
Deleted	Allocations have been set to be deleted from the Allocation application tables by the user selecting the Delete button in the front end screen that holds 'what if' summary data. The next time the allocation deletion logic is run, the record is removed from the tables.	7
Approval In Process	This status is used when the system is writing an approval request to the queue.	8
Reservation In Process	This status is used when the system is writing a reservation request to the queue.	9

Allocation Process Status

Significant functionality is attached to the status numbers in the system. The tables below reflect the “process_status” column on the ALC_ALLOC table.

Table 8–7 Process Status Column

Status	Description	Status #
Not Calculated	The allocation has not been calculated.	1
Calculation Waiting	The allocation is waiting to be processed by the algorithm. The user cannot access an allocation with this status.	2
Calculating	The system is in the process of calculating this allocation. The user cannot access an allocation with this status.	3
Calculated	The allocation has been calculated successfully.	4
Calculation Error	An error was encountered during calculation.	6
Size Profile Calculation Error	The size profile was not found for all parent/diff combinations on the allocation. Users must adjust their size profiles and recalculate the allocation.	7
Ideal Weeks of Supply (IWOS) Calculation Error	This error occurs if the allocation weeks of supply data are not sufficient for the algorithm requirements for calculation.	8

Table 8–7 (Cont.) Process Status Column

Status	Description	Status #
Quantity Limits Conflict	This error occurs if quantity limits are defined for a pack and non-sellable pack containing the same item. The system requires the user to resolve the conflict manually.	9
Status Error	The system encountered an error when submitting, reserving or approving this allocation.	10
Status Waiting	The system is in the process of submitting, reserving or approving this allocation. The user cannot enter an allocation with this status	11
Status Processing	The system is in the process of submitting, reserving or approving the allocation. The user cannot enter an allocation with this status	12
Status Processed	The allocation has been approved, reserved or submitted successfully.	13
Available Inventory Error	This error occurs if the inventory quantities that the allocation was based upon has increased or decreased by another part of the system since the time of calculation. The user must recalculate and approve based on the current inventory.	14
Item Source Conflict	This error occurs if an allocation is created using a purchase order in one allocation, and then a user attempts to create another allocation using an associated advance shipping notice (ASN). Once the purchase order allocation is approved, entering the ASN allocation prompts a question to the user stating that he or she must delete any associated purchase order allocations to continue using the ASN allocation. If the user selects ‘Cancel’, the ASN allocation is set to this new status, ‘Item Source Conflict.’ The allocation is then ‘locked down’ until either the user deletes the purchase order allocation or deletes the ASN allocation.	17
Scheduled	This scheduled allocation has been created successfully.	18
Schedule Error	This error occurs if one or more errors occur when scheduled allocation was created.	19

Sources of Data Used by Rules to Determine Gross Need

Gross Need is the need of the rule level selected. To accurately determine individual gross need, retailers want the flexibility to choose forecast data, plan data, receipt plan data, sales history data, or combinations of this data.

Through the front end, retailers select a rule based on a portion of this data that accurately gathers gross need. The source of the data used by each rule is described in this section.

To determine the net need at the store or warehouse level, the system takes the gross need and subtracts from it the stock-on-hand at the store or warehouse level. The equations that Oracle Retail Allocation uses to determine the stock-on-hand at the store or warehouse is described later in this chapter.

> **Note:** For a description of how the following rules use the data to determine gross need, see the *Oracle Retail Allocation User Guide*.

History Data Sources

For this rule, data is gathered primarily from the following tables:

Table 8–8 Data Tables (History Sources)

RMS 13.2.5	Legacy System
DEPT_SALES_HIST	This table contains one row for each dept-location-week-sales type combination. Sales history information about each combination is held here.
CLASS_SALES_HIST	This table contains one row for each class-location-week-sales type combination. Sales history information about each combination is held here.
SUBCLASS_SALES_HIST	This table contains one row for each subclass-location-week-sales type combination. Sales history information about each combination is held here.
ITEM_LOC_HIST	This table contains one row for each item-location-week-sales type combination. Sales history information about each combination may be held here.

Forecast Data Sources

For this rule, data is gathered from the following tables:

Table 8–9 Data Table (Forecast Data Sources)

RMS 13.2.5	Legacy System
DEPT_SALES_FORECAST	This table holds the forecast information summed to the department-location-eow_date.
CLASS_SALES_FORECAST	This table holds the forecast information summed to the class-location-eow_date and should be partitioned by domain_id, as well. Thus, if only a portion of the domains is forecasted, then the rebuild is done by domain_id.
SUBCLASS_SALES_FORECAST	This table holds the forecast information summed to the subclass-location-eow_date and should be partitioned by domain, as well. Thus, if only a portion of the domains is forecasted, then the rebuild is done by domain_id.
ITEM_FORECAST	This table holds the item level forecasted information from the RDF extractions. This table holds all item types. This table should be partitioned according to the domain level.

Plan Data Sources

For this rule, data is gathered from the following table:

ALC_PLAN

For a more detailed description of this table, see the section, Planning Table in Oracle Retail Allocation, in the Functional and Technical Integration chapter.

Receipt Plan Data Sources

For this rule, data is gathered from the following table:

ALC_RECEIPT_PLAN

For a more detailed description of this table, see the section, "Receipt Plan Data Sources", in the Functional and Technical Integration chapter.

History and Plan Data Sources

For this rule, the plan portion of data is gathered from the following plan table in Oracle Retail Allocation:

ALC_PLAN

The history portion of data is gathered from the following tables:

Table 8–10 Data Table (History and Plan Data Source)

RMS 13.2.5	Legacy System
DEPT_SALES_HIST	This table contains one row for each dept-location-week-sales type combination. Sales history information about each combination is held.
CLASS_SALES_HIST	This table contains one row for each class-location-week-sales type combination. Sales history information about each combination is held.
SUBCLASS_SALES_HIST	This table contains one row for each subclass-location-week-sales type combination. Sales history information about each combination is held.
ITEM_LOC_HIST	This table contains one row for each item-location-week-sales type combination. Sales history information about each combination may be held here.

Plan Re-project Data Sources

> **Note:** For a description of the Bayesian algorithm that is used in this section, see "Allocation Calculations" in the Allocations Calculations chapter.

For this rule, the historical and future plan data is gathered from the following table in Oracle Retail Allocation:

ALC_PLAN

The history portion of data is gathered from the following tables:

Table 8–11 Data table (History Source)

RMS 13.2.5	Legacy System
DEPT_SALES_HIST	This table contains one row for each dept-location-week-sales type combination. Sales history information about each combination is held here.
CLASS_SALES_HIST	This table contains one row for each class-location-week-sales type combination. Sales history information about each combination is held here.
SUBCLASS_SALES_HIST	This table contains one row for each subclass-location-week-sales type combination. Sales history information about each combination is held here.
ITEM_LOC_HIST	This table contains one row for each item-location-week-sales type combination. Sales history information about each combination may be held here.

Corporate Rules

For this rule, data is gathered from the selected column of the following tables in Oracle Retail Allocation:

- ALC_CORPORATE_RULE_HEAD
- ALC_CORPORATE_RULE_DETAIL

The column selection is based on which corporate rule is picked by the user.

> **Note:** If the retailer plans ideal weeks of supply (IWOS) by product-location, the corporate table can be accessed to create different ideal weeks of supply by store. If the retailer does not plan IWOS, a field can be created that contains the same IWOS for every store.

Quantity Limits

Quantity Limits allows the user to limit the allocation. This feature allows the user to set parameters that affect different stages of the allocation for the product-stores or product-warehouses where they are entered. The values for each applicable quantity limits selection are held on the applicable column of the ALC_QUANTITY_LIMITS table.

Using Quantity Limits, the value that gets allocated can be altered according to the user's preference. There are six constraints for quantity limits: Min, Max, Threshold, Week Of Supply (WOS), Trend, and Min Need.

For example, if the user wants to allocate at least 100 units of merchandise irrespective of the need, a minimum constraint of 100 can be applied. In this case, 100 units get allocated (if there is enough inventory to support it). Though the quantity limit gets accounted at the lowest level entity (that is, item/store combination), the retailer can also supply the quantity limits values at a higher level such as group of stores or warehouses.

Quantity Limits should work in the same manner for staple items, sellable and non-sellable staple packs since all these types of items/packs inventory are maintained at the same unit (item or pack) that gets allocated. For fashion items, though the inventory is maintained at the SKU level, allocation occurs for the parent/diff. However, you may choose to de-aggregate the parent/diff within allocation and distribute only those sizes/components which are available to allocate. Two more quantity limits - "Minimum Pack" and "Maximum Pack" can be applied in an allocation using the 'Pack Distribution' mode. These can also be applied in the 'Simple' mode but in specific cases where the allocation contains only pack items that have been selected to be allocated as a single entity.

Stop Ship

A 'stop ship' is a product/location combination that prevents an item from shipping to that location. Allocation refers to the 'Store Close Date' and 'Stop Order Day' in the STORE table in order to determine if a store is open for receiving inbound shipments.The system then looks at the release dates entered on the product window and compares them with STORE table records entered via the merchandising system. The Store Close Date minus the Stop Order Days will indicate the last date on which inbound shipments can be received at a store. If the release date is on or after the stop ship dates, the system inserts '0' into the min and max columns of quantity limits for this store-item. Stop shipment records appear as item location exceptions.

The release date is on the ALC_ITEM_LOC table and is represented by the column, release_date. See the *Oracle Retail Allocation User Guide* for more information.

Net Need at Store Level Calculation

On a fundamental level, net need is gross need minus the on-hand at the store or warehouse.

> **Note:** Although quantity limits also affect net need, they are not addressed in the calculation illustrated by this section.

To determine the gross need, Oracle Retail Allocation gathers the information based upon one of the rules selected by the retailer through the front end. Oracle Retail Allocation uses the following equation to determine the on-hand at the location that is subtracted from the gross need result.

For a store location,

On Hand = (Stock-on-hand at the store+ Stock in transit+ Stock on order [stock that is expected by the on order commit date]+ Transfers of stock expected + Stock on allocation) - (Outgoing transfers + Return to vendor stock+ Unavailable stock+ Transfers on reserve) = (SOH +In Transit + On Order + TSF Expected + On Alloc) - (TSF Out + RTV + Unavailable + TSF Reserved)

For a warehouse location, On Hand =

(Stock-on-hand at the warehouse + Stock in transit + Stock on order [stock that is expected by the on order commit date] + Transfers of stock expected + Stock on allocation) - (Outgoing transfers + Return to vendor stock+ Unavailable stock + Transfers on reserve + Outbound allocations) = (SOH + In Transit + On Order + TSF Expected + On Alloc) - (TSF Out + RTV + Unavailable + TSF Reserved + Alloc Out). For determining the on hand quantity, we need to consider all these different inventory buckets, some of which are present in Allocation while the rest are derived from RMS forms/tables/views.

Allocation side

- SOH at the store
- Stock in transit
- Stock on order
- Stock On Alloc
- Back orders

RMS side

- TSF Expected
- TSF Reserved
- TSF Out
- RTV
- Unavailable
- Back orders
- Alloc Out (only for warehouses)

See the Selecting Policies section in the Creating Standard Allocations chapter of the *Oracle Retail Allocations User Guide* for additional information.

Closing Allocations

This section addresses the three possible methods of closing allocations. Note that the closure of the allocation in Oracle Retail Allocation entails 'all or nothing' processing logic.

- Warehouse and Store initiated Closures: The majority of RMS allocation records should be closed as part of the retailer's warehouse management system and the retailer's store inventory management system integration with RMS.
- For purchase orders closed via batch functionality: RMS allocation records attached to these closed purchase orders are closed, if the allocation meets RMS validations. RMS cancels the associated quantities on the RMS allocation records and closes the RMS allocation records. All the quantities remaining for related RMS allocation records are cancelled, and the RMS allocation records are closed if no quantities are in transit. If the RMS allocation records cannot be closed, there is no further action.
- For purchase orders closed manually online: If RMS allocation records exist when the user attempts to either cancel an item or cancel all items, a message offers the user an option to cancel the associated RMS allocation records or not. If the user selects to close the allocations, all the quantities remaining for related RMS allocation records are cancelled, and the RMS allocation records are closed if no quantities are in transit. If the user selects to not close the allocations, there is no further action.

Note: The Oracle Retail Allocation application is updated through the table triggers when actions occur on RMS allocation records.

9

Functional and Technical Integration

This chapter discusses the integration among Oracle Retail Allocation and other systems and it provides the following:

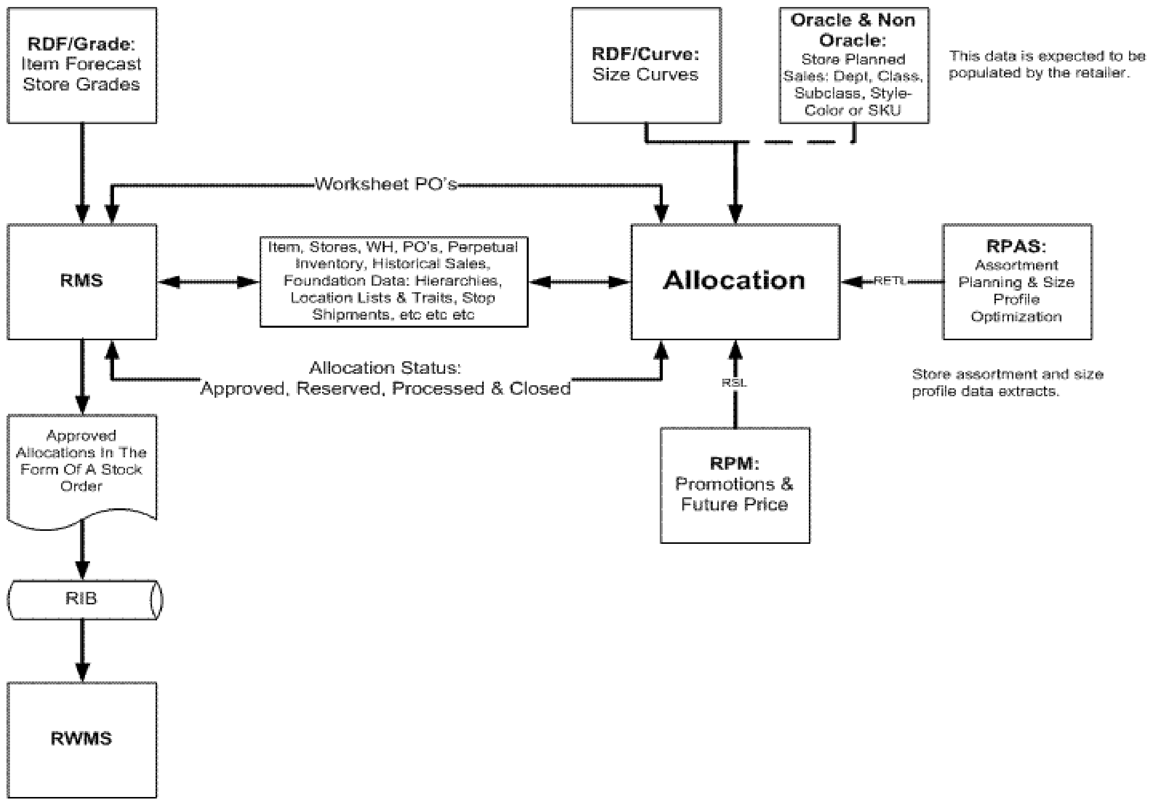
- An integration interface allocation-related dataflow across the enterprise.
- The tables and triggers that are in external systems or related to external systems that Oracle Retail Allocation uses (for example, RMS).
- A functional description of RMS dependencies and assumptions that affect Oracle Retail Allocation.
- Information necessary to integrate Oracle Retail Allocation and Oracle Retail Workspace.

Integration Interface Allocation-Related Dataflow

This section provides an overview as to how Oracle Retail Allocation is functionally integrated with other systems (including other Oracle Retail systems). The discussion primarily concerns the flow of allocation-related business data across the enterprise.

> **Note:** Symbol denotes tables held on the Merchandising table. Oracle Retail allocation pulls the data from these Merchandising tables through the use of the JDBC connection.

Figure 9–1 Oracle Retail Allocation-Related Dataflow Across the Enterprise


RDF/Grade:
Item Forecast
Store Grades
RDF/Curve:
Size Curves
Oracle & Non Oracle:
Store Planned Sales: Dept, Class, Subclass, Style-Color or SKU
This data is expected to be populated by the retailer.
Worksheet PO's
RMS
Item, Stores, WH, PO's, Perpetual Inventory, Historical Sales, Foundation Data: Hierarchies, Location Lists & Traits, Stop Shipments, etc etc etc
Allocation
RETL
RPAS:
Assortment Planning & Size Profile Optimization
Allocation Status:
Approved, Reserved, Processed & Closed
RSL
Store assortment and size profile data extracts.
Approved Allocations In The Form Of A Stock Order
RPM:
Promotions & Future Price
RIB
RWMS

Note: Oracle Retail allocation pulls the data from the Merchandising tables in RMS using the JDBC connection.

The diagram above shows the overall direction of the dataflow among the products. The accompanying explanations are written from a system-to-system perspective, illustrating the movement of data.

From Oracle Retail Demand Forecasting System to Oracle Retail Allocation via Merchandising System

The history data is subjected to processing that yields data that is sent back to the merchandising system. From there, Oracle Retail Allocation pulls the following data:

- Forecasting data: Oracle Retail Allocation accesses forecasting data that originates in the Oracle Retail Demand Forecasting (RDF) system. RDF is Oracle Retail's statistical and causal forecasting solution. It uses state-of-the-art modeling techniques to produce high quality forecasts with minimal human intervention. RDF is an application that resides on the Oracle Retail Predictive Application Server (RPAS). Oracle Retail Allocation uses forecasting data as a basis for calculating gross need and can access the following five levels of forecasting data: department, class, subclass, style-color and item.
- Store grade group data: Oracle Retail Allocation accesses store grade group data that originates in Grade. Grade is Oracle Retail's application that groups store locations together intelligently, based on similarities in performance, customer type, geography, or some other factor that allows the stores within each group to

be treated as one unit. Grade is an application that is part of the Oracle Retail Predictive Application Server (RPAS). Internally, Oracle Retail Allocation also updates its store grade groups data groups based on the most current definitions. This update plays an important role when many months pass between initial and final allocations.

From Oracle Retail Planning Application to Oracle Retail Allocation

Plan Data

Oracle Retail Allocation accesses plan data that originates in the planning application (including Oracle Retail's planning applications that reside on the RPAS server). The RPAS products are applications that provide functionality for developing, reconciling, and approving plans. When interfacing with Oracle Retail planning applications, Oracle Retail Allocation accesses department, class, subclass, parent/diff, or SKU plan data at the store-week level. Oracle Retail Allocation can be used as a tool to verify the final product-store plans and to initiate a PO to execute the plan. In other words, Oracle Retail Allocation can take the retailer's plan or forecast and execute it. Both the Oracle Retail and the legacy planning applications populate a planning table, ALC_PLAN, which resides within Oracle Retail Allocation. See the section, Planning Table in Oracle Retail Allocation, later in this chapter.

Note:

- Oracle Retail Allocation interfaces with Oracle Retail Assortment Planning by way of an output file from Assortment Planning through Oracle Retail Extract, Transform, and Load (RETL) to access only SKU, style-color data at the store-week level. For more information, see the chapter, "Oracle Retail Extract, Transform, and Load (RETL) Batch Processing".
- RMS is the system of record for Oracle Retail Allocation. Hence Allocation inherits the merchandise hierarchy from RMS. RMS and Assortment Planning (AP) have different merchandise hierarchies. Users of RMS/AP/Allocation who wish to export information from AP to Allocation must ensure that the AP merchandise hierarchy is compatible with that of RMS/Allocation.

From Size Profile Optimization to Oracle Retail Allocation

Size Profile Data

Oracle Retail Allocation uses size profile data from Oracle Retail Size Profile Optimization (SPO) system. SPO creates optimal profiles of size distribution by both merchandise category and by store. The size profile data is extracted at the following levels: department level, class level, sub-class level and item level. The data for all the levels are extracted in a single file. For more information, see the section, Size Profile Logic in the Allocation Calculations chapter, and see the Oracle Retail Extract, Transform, and Load (RETL) Batch Processing chapter.

From Retail Demand Forecasting/Curve to Oracle Retail Allocation

Size Profile Data

Curve data becomes size profile data once it's integrated into Retail Allocation. If allocations are made at the style level, Retail Allocation utilizes the Curve data to get to the SKU level. For more information, see the section, Size Profile Logic in the Allocation Calculations chapter, and see the Oracle Retail Extract, Transform, and Load (RETL) Batch Processing chapter.

From Oracle Retail Warehouse Management System to Oracle Retail Allocation via Oracle Retail Merchandising System

- Appointment data

  Appointment data is one source that identifies item(s) to be allocated.
- Warehouse inventory position data
- ASN, BOL, and Transfer information

From Oracle Retail Promotion Management to Oracle Retail Allocation

RPM provides the following to Allocation:

- **Future Retail Price Data** - Oracle Retail Allocation has the ability to get a real time price from RPM as it is integrated directly with RPM. Allocation uses this data to provide you with the future retail price value of the entire allocation (based on its quantities). In addition, you can access future retail price values by location and by item.

From Oracle Retail Merchandising System to Oracle Retail Allocation

Note for RMS users only: Item, purchase order, supplier, sales and other data are accessed directly from the RMS tables, with no need to interface data via batch modules.

- Item data Oracle Retail Allocation can allocate at the item, style-color, pack, or item list level. Styles, items, and packs can be mixed on a single allocation.
- PO data
- Hierarchy data
- Sales history data (for items, user-defined attributes (UDA), warehouses, stores, and so on)
- Foundation data (supplier data, shipping tables, and so on)

From Oracle Retail Allocation to Oracle Retail Merchandising System

Oracle Retail Allocation calculates the allocation based on the information it has received from the merchandising system. Once the retailer reviews and approves the allocation, Oracle Retail Allocation sends the following information back to the merchandising system:

- Approved or reserved allocation data

- Worksheet status POs that contain product, supplier and quantity information (the only remaining actions to be taken in the merchandising system are to approve the PO and, if desired, to truck scale the PO.) These worksheet status purchase orders may be created or updated from within the Oracle Retail Allocation front end.

From Oracle Retail Merchandising System to Oracle Retail Warehouse Management System

Note for RMS users only: Oracle Retail Allocation utilizes the existing integration between RMS and RWMS. This interface currently passes purchase order, item, location, and allocation information from RMS to RWMS.

Based upon the approved allocation information from Oracle Retail Allocation, the merchandising system sends the following information to the distribution management system:

- Approved allocation data represents the store quantity instructions for allocating a specific quantity of stock at the store level.

From Oracle Retail Active Retail Intelligence to Oracle Retail Allocation

Oracle Retail Active Retail Intelligence (ARI) is an exception management and resolution system driven by custom business rules. Depending upon ARI's configuration, an ARI user could receive an alert that includes a link to Oracle Retail Allocation in the form of a URL address. The user could then log on to Oracle Retail Allocation in order to address the contents of the ARI alert.

Persistence Layer Integration

This section addresses Oracle Retail Allocation's persistence layer method of integration:

Persistence Layer Integration (Including Tables and Triggers)

Tables Populated by External Systems

The following tables are owned by Oracle Retail Allocation. The data within them is populated by external systems. For descriptions of each table and its columns, see the *Oracle Retail Allocation Data Model*.

- ALC_CORPORATE_RULE_DETAIL
- ALC_CORPORATE_RULE_HEAD
- ALC_IDEAL_WEEKS_OF_SUPPLY
- ALC_PLAN
- ALC_RECEIPT_PLAN
- ALC_SIZE_PROFILE
  - Can also be populated through size profile setup via the front end of the application.

Planning Table in Oracle Retail Allocation

Planning applications populate a planning table, ALC_PLAN, that resides within Oracle Retail Allocation. This table includes the following columns:

- Plan ID
- Loc
- EOW.date
- Department
- Class
- Subclass
- Item
- Diff1_id, Diff2_id, Diff3_id, Diff4_id
- Quantity

A record can thus exist at any of the following levels by week-store-quantity:

- Department
- Department-class
- Department-class-subclass
- Item-color
- Item

The ALC_RECEIPT_PLAN table includes the following columns:

- Plan ID
- Loc
- EOW.date
- Department
- Class
- Subclass
- Item
- Diff1_id, Diff2_id, Diff3_id, Diff4_id
- Quantity

A record can thus exist at any of the following levels by week-store-quantity:

- Department
- Department-class
- Department-class-subclass
- Item-color
- Item
- Pack

Merchandising Interface Tables

Oracle Retail Allocation and RMS share certain database tables and processing logic. This integration provides the following two important benefits:

- The number of interface points that need to be maintained is minimized.
- The amount of redundant data (required if the rest of the Oracle Retail product suite is installed) is limited.

Oracle Retail Allocation exchanges data and processing with RMS in four ways:

- By reading directly from RMS tables.
- By directly calling RMS packages.
- By reading Oracle Retail Allocation views based on RMS tables.
- Oracle Retail Allocation triggers reside in RMS tables. These triggers cause actions (create, delete, update) on RMS tables based on Oracle Retail Allocation business rules.

Oracle Retail Merchandising System Tables (for Retailers with Oracle Retail Merchandising System only) used by Oracle Retail Allocation

The following table illustrates the tables from which Oracle Retail Allocation gets its data from RMS.

Table 9–1 RMS Tables Used by Allocation

RMS Tables	
Functional Area	Associated Tables
Item data	SUB_ITEMS_HEAD
	SUB_ITEMS_DETAIL
	ITEM_MASTER
	ITEM_SUPP_COUNTRY
	ITEM_SUPPLIER
	ITEM_LOC
	ITEM_LOC_HIST
	ITEM_LOC_SOH
	ITEM_PARENT_LOC_HIST
Skulist data	SKULIST_HEAD
	SKULIST_DETAIL
Pack data	PACKITEM
	ITEM_MASTER
	ITEM_LOC
Order data	ORDHEAD
	ORDLOC_WKSHT
	ORDLOC
	ORDSKU
	ALLOC_HEADER
	ALLOC_DETAIL
	SHIPMENT

Table 9–1 (Cont.) RMS Tables Used by Allocation

RMS Tables	
Supplier data	SUPS
	ITEM_SUPPLIER
Location list data	LOC_LIST_HEAD
	LOC_LIST_DETAIL
	LOC_LIST_CRITERIA
Merchandise hierarchy data	DEPS
	CLASS
	SUBCLASS
	ITEM_PARENT
	DIFF
	SKU
Organizational hierarchy data	STORE
	WH
	WH_STORE_ASSIGN
Shipment data	SHIPMENT
	SHIPSKU
Store grade data	STORE_GRADE_GROUP
	STORE_GRADE
	STORE
	BUYER
	STORE_GRADE_STORE
Location traits data	LOC_TRAITS
	LOC_TRAITS_MATRIX
	LOC_AREA_TRAITS
	LOC_REGION_TRAITS
	LOC_DISTRICT_TRAITS
Transfer data	TSFHEAD
	TSFDETAIL
User defined attribute (UDA) data	UDA
	UDA_VALUES
	UDA_ITEM_LOV
Forecast data	DEPT_SALES_FORECAST
	CLASS_SALES_FORECAST
	SUBCLASS_SALES_FORECAST
	ITEM_FORECAST

Table 9–1 (Cont.) RMS Tables Used by Allocation

RMS Tables	
Sales data	DEPT_SALES_HIST
	CLASS_SALES_HIST
	SUBCLASS_SALES_HIST
	ITEM_LOC_HIST
	ITEM_PARENT_LOC_HIST
Appointment data	APPT_HEAD
	APPT_DETAIL
Auto Quantity Limits	DEP
	CLASS
	SUBCLASSS
	ITEM_PARENT
	DIFF
	SKU
	GROUP_TYPE
	GROUP_VALUE
	MINIMUM_NET_NEED
	MAXIMUM_NET_NEED
	THRESHOLD
	TREND
	WEEKS_OF_SUPPLY
	MINIMUM_GROSS_NEED
	MINIMUM_PACK
	MAXIMUM_PACK
	START_DATE
	END_DATE

Oracle Retail Allocation-Owned Triggers Residing on Oracle Retail Merchandising System Tables

Table 9–2 Triggers

Triggers	Details
ALC_TABLE_ALD_AUR - 1 - 4	This trigger is involved in the following processing: Whenever a portion of an allocation order is worked on by the distribution center by selecting, distributing, transferring or receiving inventory, the allocation within Oracle Retail Allocation is placed into a 'Processed' status. The user can no longer change that allocation in Oracle Retail Allocation.
ALLOC_STATUS_TRIGGER	This trigger is on the RMS table ALLOC_HEADER. The trigger updates the status in Oracle Retail Allocation table ALC_ALLOC to 4 (closed). This trigger is fired only if the status on RMS table ALLOC_HEADER is updated to 'C' (closed).

Table 9–2 (Cont.) Triggers

Triggers	Details
ALLOC_STATUS_ TRIGGER_AU	The closure logic within the Oracle Retail Allocation application accounts for the multiplicity between ALLOC_HEADER records and the Oracle Retail Allocation (ALC_XXX) tables. The table triggers only set a Oracle Retail Allocation allocation number to closed if all ALLOC_HEADER records have been closed.

Oracle Retail Merchandising System Functional Dependencies and Assumptions

This section describes the functional dependencies of Oracle Retail Allocation on RMS.

Oracle Retail Merchandising System Differentiator Setup

The RMS item structure allows multiple item levels and multiple differentiators. To structure item setup for use with Oracle Retail Allocation, the retailer must understand the implications of the Item Aggregate Indicator and the Aggregate Indicators that exist at the differentiator level.

The following section describes how an item family must be structured to enable the Oracle Retail Allocation product to differentiate the items among fashion, staple and pack items.

Fashion Item

Note: In the Allocation application and this document, the terms 'style/color' and 'parent/diff' are used interchangeably.

RMS allows for the potential of three item levels. For a customer who allocates based on the concept of parent/diff, the style can be translated to RMS item setup as being the level one item in the item family. The SKU can be translated to RMS item setup as being the transaction level item (this could be level one, two or three). There is no requirement within RMS that forces a 'fashion' item to be multi-level.

An item is viewed as a fashion item only if the Item Aggregate Indicator in the Attributes section of the Item Master Window is selected for the style (level one item) in the item family.

Once the item aggregate indicator has been selected, the user needs to indicate which differentiator should be curved by allocations. Each item may contain up to four differentiators.

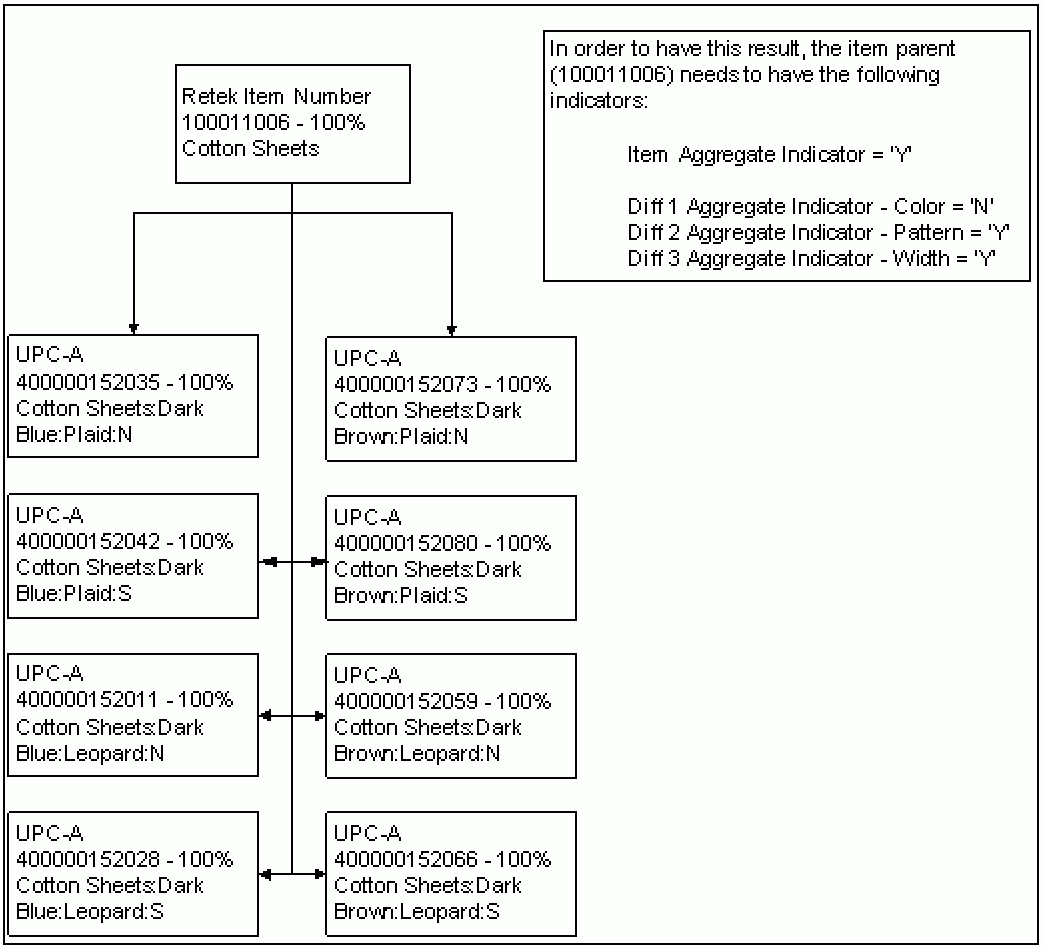
The Aggregate check box is enabled when more than one differentiator is being created for an item where the Item Aggregate Indicator has been selected. The differentiator that the customer wants to be curved by Oracle Retail Allocation must be the only differentiator that is *not* indicated on the Item Master Window.

Below is an example of a fashion item, its indicators within RMS, and what is visible.

Item 100011006 has three differentiators associated.

- Color/pattern/width

Figure 9–2 RMS Differentiator Setup


Retek Item Number
100011006 - 100%
Cotton Sheets
In order to have this result, the item parent (100011006) needs to have the following indicators:
Item Aggregate Indicator = 'Y'
Diff 1 Aggregate Indicator - Color = 'N'
Diff 2 Aggregate Indicator - Pattern = 'Y'
Diff 3 Aggregate Indicator - Width = 'Y'
UPC-A
400000152035 - 100%
Cotton Sheets:Dark Blue:Plaid:N
UPC-A
400000152073 - 100%
Cotton Sheets:Dark Brown:Plaid:N
UPC-A
400000152042 - 100%
Cotton Sheets:Dark Blue:Plaid:S
UPC-A
400000152080 - 100%
Cotton Sheets:Dark Brown:Plaid:S
UPC-A
400000152011 - 100%
Cotton Sheets:Dark Blue:Leopard:N
UPC-A
400000152059 - 100%
Cotton Sheets:Dark Brown:Leopard:N
UPC-A
400000152028 - 100%
Cotton Sheets:Dark Blue:Leopard:S
UPC-A
400000152066 - 100%
Cotton Sheets:Dark Brown:Leopard:S

The retailer wants to have Oracle Retail Allocation apply the curve to Color. Therefore, it sees information within the Oracle Retail Allocation screens based upon the pattern and width differentiators.

All of the transaction level children have their item and differentiator aggregate indicators set to 'N'. These values are only maintained for the level one item. All other items in the system (including packs) have those indicators defaulted to 'N'.

In this scenario, if the retailer is creating an allocation for the parent item (100011006), it has visibility to four different levels of the 'style'.

- 100011006 - 100% Cotton Sheets Plaid:N
- 100011006 - 100% Cotton Sheets Plaid:S
- 100011006 - 100% Cotton Sheets Leopard:N
- 100011006 - 100% Cotton Sheets Leopard:S

Staple Item

A staple item is every item in the system where the level one item in the item family does not have the Item Aggregate Indicator selected. In this scenario, the Oracle Retail Allocation retailer has visibility to the transaction level item only. There is no roll up of item information. The retailer also has visibility to the non-sellable packs that contain the component staple item and is able to include or exclude those packs from the allocation.

Pack Item

There are multiple types of packs that may be set up within RMS. The key criteria for Oracle Retail Allocation is whether the pack is sellable or non-sellable, whether the pack contains multiple component items and whether or not those multiple components items are of one type (for example, fashion as opposed to staple).

When creating your packs, consider the following pack assumptions made by Oracle Retail Allocation:

- Oracle Retail Allocation does not have the ability to allocate packs that contain fashion and staple items.

Summary of Items and How Oracle Retail Allocation Handles Them

- **Single staple item**:

  These items are individually allocated and can be selected from item LOV search criteria.

- **Single fashion item**:

  These items are allocated as part of their style/color. They may also be individually selected from the worksheet to distribute only those sizes/components that are available to allocate. Single fashion items may also optionally be included in a Fashion Group Allocation.

  Note: Fashion packs cannot be de-aggregated.

- **Style/color**:

  The transaction level (item) is allocated as visible in the View Assortment window. However, the allocation is created at the item level one/differentiator (style/color) level. The item level one/differentiator (style/color) level is where retailers work with the allocation.

- **Simple sellable staple pack and complex sellable staple pack**:

  These types of packs are included in an allocation when they are individually allocated.

- **Simple non-sellable staple pack and complex non-sellable staple pack**:

  These types of packs are included in an allocation when the component of the pack item is allocated or when the non-sellable pack itself is allocated.

- **Simple sellable fashion packs and complex sellable fashion packs**:

  These types of packs are included in an allocation when they are individually allocated. They are not being automatically included in any fashion items allocation.

- **Simple non-sellable fashion packs and single color complex non-sellable fashion packs**:

  These packs can be allocated as part of a style/color or they may also be allocated individually (components must stay within the pack). They could also be allocated as part of a Fashion Group allocation.

- **Multi-color complex non-sellable fashion packs**:

  These packs can be allocated individually or they can be allocated as part of a Fashion Group Allocation.

Oracle Retail Allocation Functional Assumptions Related to Oracle Retail Merchandising System

- When fashion items are individually selected for an allocation (rather the selecting a style/color), the items are allocated as staple items.
- A single allocation cannot have both fashion item(s) and staple item(s).
- Non-sellable fashion packs are visible on the trans level view of the Assortment View screen.
- The list of values on the search screen displays staple items, sellable/non-sellable staple packs, and sellable simple/complex fashion packs.
- The stop shipment record for a non-sellable staple pack must be at the component item level for the stop shipment to be recognized by Oracle Retail Allocation. A record for the non-sellable staple pack itself has no effect.

10

Oracle Retail Extract, Transform, and Load (RETL) Batch Processing

The module works in conjunction with the Oracle Retail Extract Transform and Load (RETL) framework. This architecture optimizes a high performance data processing tool that allows database batch processes to take advantage of parallel processing capabilities.

The RETL framework runs and parses through the valid operators composed in XML scripts.

This chapter provides an overview of Oracle Retail Allocation RETL processing and defines the export file from Curve/Plan to Oracle Retail Allocation that is used when exporting Curve/Financial Plan values. For more information on RETL, see the product's Programmer's Guide.

Note: In this chapter, some examples refer to RETL programs that are not related to Oracle Retail Allocation. References to these programs are included for illustration purposes only.

Note: The RETL loads into Allocation are point to point integration between Oracle Retail product, and are not designed to support generic uploads from other systems.

Functional Overview

The extracts from RETL may contain up to four levels of plan/profile. They consist of the department level, class level, subclass level and item level. All of these levels are contained in a single normalized file. Each record in the Curve file has a dedicated 'space' and distinct position for department, class, subclass, item, store, diff1, diff2, diff3, diff4 and size profile quantity values and each record in the Plan file has a dedicated 'space' and distinct position for department, class, subclass, item, store, diff1, diff2, diff3, diff4, EOW date and plan quantity values. It is crucial that the records are mapped using the correct positions and space/padding rules for each data value.

- Regardless of the level of financial plan/profile, each record must include a store, diff value in one of the four diff value fields and a quantity value (including an EOW date for Plan only).

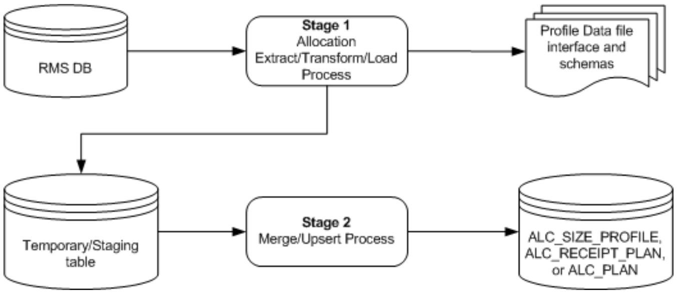
- Department-level financial plan or profiles include a department data value in the dedicated department field. The class, subclass and item fields do not contain any values. They remain empty.
- Class-level financial plan or profiles include a department and class data value in the dedicated department and class fields. The subclass and item fields do not contain any values. They remain empty.
- Subclass-level financial plan or profiles include a department, class and subclass data value in the dedicated department, class and subclass fields. The item fields do not contain any values. They remain empty.
- All of the department, class and subclass record exports contain only the non-aggregate diff values mapped from the specific diff value in ITEM_MASTER to the corresponding diff value in the export file. It is crucial that the non-aggregate diffs are mapped to the correct diff_id in the export file.
- Item-level profiles include aggregate level IDs in the dedicated item field. The department, class and subclass fields do not contain any values. They remain empty. The item level export records contains both the aggregate and non-aggregate diff values mapped from the specific diff id in ITEM_MASTER to the associated diff position in the export file.
- Item-Level financial plan or profiles include item data value in the dedicated item field. The department, class and subclass fields do not contain any values. They remain empty.
- All the data values must start in the beginning of the corresponding field, and padding comes after the data to fill all the dedicated space for that data field.

Oracle Retail Extract, Transform, and Load Batch Processing Architecture

The diagram below illustrates the extraction processing architecture. The plan/size profile architecture adheres to what is shown in the diagram:

The architecture relies upon two distinct stages, each of which is described in the passages that follow.

Figure 10–1 RETL Batch Processing Architecture



Processing Stage 1

Stage 1 involves importing plan/profile data and looking up required information in the RMS ITEM_MASTER table (item-level plans/profiles only). The resulting output from this stage is a temporary table that contains any item-level and department/class/subclass-level plans/profiles.

The detailed flow is as follows:

1. Insert dept-level plans/profiles directly into the staging table.
2. Insert class-level plans/profiles directly into the staging table.
3. Insert subclass-level plans/profiles directly into the staging table.
4. The item-level plans/profiles require lookups with the ITEM_MASTER table. The processing logic for transaction-level items is to do a three-way left outerjoin on the ITEM_MASTER table to retrieve each parent and grandparent item aggregate indicators. The Item file then lookups its item id in the item master table and uses the STYLE set as the parent or grandparent item id whose ITEM_AGGREGATE_IND = 'Y'. These item level plans/profiles are then inserted into the staging table.

Error Handling

Any item records that do not have a parent and grandparent are flagged as warnings. Any items in the incoming data file that do not match an item in the ITEM_MASTER table are flagged as errors.

Processing Stage 2

Stage 2 involves inserting and updating the plan or profile records into the final destination ALC_PLAN, ALC_RECEIPT_PLAN, or ALC_SIZE_PROFILE table respectively.

The detailed processing is as follows:

1. Update quantity when matched department, class, subclass, style, store, size1, size2, size3, size4.
2. Otherwise, insert record.

Configuration

This section covers configuration.

Oracle Retail Extract, Transform, and Load

Before trying to configure and run Oracle Retail Allocation RETL, install RETL version 13.2, which is required to run Oracle Retail Allocation RETL. Run the 'verify_retl' script (included as part of the RETL installation) to ensure that RETL is working properly before proceeding.

Oracle Retail Extract, Transform, and Load User and Permissions

Oracle Retail Allocation RETL should be installed and run as the RETL user.

Additionally, the permissions should be set up as per the *RETL Programmer's Guide*. Oracle Retail Allocation RETL reads, creates, deletes and updates data for tables. If these permissions are not set up properly, processing fails.

Environment Variables

See the *RETL Programmer's Guide* for RETL environment variables that must be set up for your version of RETL. You need to set ALCHOME to your base directory for Oracle Retail Allocation RETL. This is the top level directory that you selected during the installation process (see Oracle Retail Allocation Installation Guide) in your .kshrc, you should add a line such as the following:

```
export ALCHOME=<base directory path for  ALLOCATION RETL>
```

Execute the setup-security-credential.sh script after the installation from RFX_HOME/bin directory. This script provides the options for adding/updating the database credentials. Enter the values for the following parameters:

- dbuseralias
- username

A secure wallet file (cwallet.sso) is created under "RFX_HOME/etc/security" directory of RETL installation.

alc_config.env Settings

On the Oracle Retail Allocation side, make sure to review the environmental parameters in the alc_config.env file before executing the batch module. Depending upon your local settings, the variables may need to be changed.

Configure Oracle Retail Extract, Transform and Load

1. Log in to the UNIX server with a UNIX account that runs the RETL scripts.
2. Change directory to $ALCHOME/rfx/etc.
3. Modify the alc_config.env script:
   - Change the DBNAME variable to the name of the Oracle Retail Allocation database. For example:

     ```
     export DBNAME=int9i
     ```

   - Set the user alias fields such as RETL_WALLET_ALIAS and ORACLE_WALLET_ALIAS.
   - Set the other variables namely: RFX_HOME, RFX_TMP, TNS_ADMIN, ALCHOME, ORACLE_HOME, JAVA_HOME, PATH, LD_LIBRARY_PATH, SPO_GID_FILENAME.

Update the rfx.conf, vdate.txt files with the DB information and other local settings necessary.

Running the Module

This section covers running the module.

Schema File

RETL uses a schema file to specify the format of an incoming or outgoing dataset. The schema file defines each column's data type and format, which is then used within RETL to format/handle the data. Schema file names are hard-coded within each module because they do not change on a day-to-day basis. All schema files end with '.schema' and are placed in the 'rfx/schema' directory. For more information about schema files, see the latest *RETL Programmer's Guide*.

The input data schema file for the Oracle Retail Allocation module is named as alcl_plan.schema for Plan and as alcl_size_profile.schema for profile and is shown later in this chapter.

Mandatory Multi-Threading and Command Line Parameters

In contrast to the way in which multi-threading is defined in UNIX, Oracle Retail Allocation uses 'multi-threading' to refer to the running of a single RETL program multiple times on separate groups of data simultaneously. Multi-threading can reduce the total amount of processing time.

For this Oracle Retail Allocation module, multi-threading is *mandatory*, and the file-based module has to be run once for each input file.

- The alcl_pan / alcl_size_profile module requires the following two input parameters:
- The uniquely named thread number. Note that the thread number is used internally and is not related to any output file or table name.
- The following example illustrates a scenario in which the retailer runs the alcl_size_profile.ksh module three times for three input files:

```
The load batches, example alcl_size_profile.ksh, loads its data from $ALC_
HOME/data
alcl_size_profile.ksh profile_01.dat 1
alcl_size_profile.ksh profile_03.dat 3
```

The transform batches alct_plan and alct_size_profile do not take any input parameters. They execute the flat files in the path `$ALC_HOME/data alcl_size_profile.ksh profile_02.dat 2`. Running only `alct_size_profile.ksh` at the prompt executes the transform scripts.

Program Features

The extraction programs are written in the RETL framework and include the following features:

- Business virtual date
- Program return code
- Program status control files
- Restart and recovery
- Message logging
- Program error file
- Reject files

Business Virtual Date

The business virtual date must be placed in the vdate.txt file in the $ALCHOME/rfx/etc directory prior to running the RETL module.

Program Return Code

RETL programs use one return code to indicate successful completion. If the program successfully runs, a zero (0) is returned. If the program fails, a non-zero is returned.

Program Status Control Files

To prevent a program from running while the same program is already running against the same set of data, the Oracle Retail Allocation RETL code utilizes a program status control file. At the beginning of each module, alc_config.env is run. It checks for the existence of the program status control file. If the file exists, then a message stating, '${PROGRAM_NAME} has already started', is logged and the module exits. If the file does not exist, a program status control file is created and the module executes.

If the module fails at any point, the program status control file is not removed, and the user is responsible for removing the control file before re-running the module.

File Naming Conventions

The naming convention of the program status control file allows a program whose input is a text file to be run multiple times at the same time against different files.

The name and directory of the program status control file is set in the configuration file (alc_config.env). The directory defaults to $ALCHOME/error. The naming convention for the program status control file itself defaults to the following dot separated file name:

- The program name
- The input filename
- 'status'
- The business virtual date for which the module was run

For example, the program status control file for the alcl_size_profile.ksh program (with an input file name of alcl_size_profile_01.dat specified as the first argument on the command line) would be named as follows for the batch run of January 5, 2001:

```
$ALCHOME/error/alcl_size_profile.alcl_size_profile_01.dat.status.20010105
```

Oracle Retail Allocation Oracle Retail Extract, Transform, and Load Restart and Recovery

The Oracle Retail Allocation RETL module imports data from a flat file, performs transformations if necessary and then loads the data into the applicable Oracle Retail Allocation table.

This module uses a single RETL flow and does not require the use of restart and recovery. If the extraction process fails for any reason, the problem can be fixed, and the entire process can be run from the beginning without the loss of data. For a module that takes a text file as its input, the following two choices are available that enable the module to be re-run from the beginning:

1. Re-run the module with the entire input file.
2. Re-run the module with only the records that were not processed successfully the first time.

Message Logging

Message logs are written while the module is running in a format described in this section.

Daily Log File

Every RETL program writes a message to the daily log file when it starts, while it is running, and when it finishes. The name and directory of the daily log file is set in the configuration file (alc_config.env). The directory defaults to $ALCHOME/log. All log files are encoded UTF-8.

The naming convention of the daily log file defaults to the following 'dot' separated file name:

- The business virtual date for which the module is run.
- '.log'

For example, the location and the name of the log file for the business virtual date of January 5, 2001 would be the following:

```
$ALCHOME/log/20010105.log
```

Format

As the following examples illustrate, every message written to a log file has the name of the program, a timestamp, and either an informational or error message:

```
alct_size_profile 16:06:30: Program started.Number of input files processed = 1.
Number of output files produced = 1
alcl_size_profile 13:20:01: Program started for thread 1…
alcl_size_profile 13:20:05: Analyzing temp table rmsint1201.alcl_size_profile_
temp_1
alcl_size_profile 13:20:13: Merging into alc_size_profile
alcl_size_profile 13:20:27: Program completed successfully for thread 1.
alct_size_profile 16:06:30: Program completed without errors
```

If a program finishes unsuccessfully, an error file is usually written that indicates where the problem occurred in the process. There are some error messages written to the log file, such as 'No output file specified', that require no further explanation written to the error file.

Program Error File

In addition to the daily log file, each program also writes its own detail flow and error messages. Rather than clutter the daily log file with these messages, each program writes out its errors to a separate error file unique to each execution.

The name and directory of the program error file is set in the configuration file (alc_config.env). The directory defaults to $ALCHOME/error. All errors and all routine processing messages for a given program and input file on a given day go into this error file (for example, it contains both the stderr and stdout from the call to RETL). All error files are encoded UTF-8.

The naming convention for the program's error file defaults to the following 'dot' separated file name:

- The program name
- The business virtual date for which the module was run

For example, all errors and detail log information for the alcl_size_profile program would be placed in the following file for the batch run of January 5, 2001:

```
$ALCHOME/error/alcl_size_profile.alcl_size_profile_01.dat.20010105
```

The error file for the transform batch contains the name of the files processed with exit status of each file. If any of the file has bad record like width of some field exceeding the maximum allowed, this error file shows the bad records also.

The naming convention for the error file of the transform batch is

Program name_err.business virtual date for which the scripts were run.

```
For example,
alct_size_profile_err.20061005
```

Oracle Retail Allocation Reject Files

The Oracle Retail Allocation module may produce a reject file if it encounters data related problems, such as the inability to find data on required lookup tables. A given module tries to process all data and then indicates that records were rejected. All data problems are thus identified in one pass and corrected. The module can then be re-run to successful completion. If a module does reject records, the reject file is not removed. The user is responsible for removing the reject file before re-running the module.

Typical Run and Debugging Situations

The following examples illustrate typical run and debugging situations for each type of program. The file names referenced in the example below (log, error, and so on) assume that the module is run on the business virtual date of March 9, 2001.

```
Example for running Transform batch
Run alct_size_profile.ksh
```

1. Change the directory to $ALC_HOME/src
2. In the prompt enter,

$alct_size_profile.ksh

If the module runs successfully, the following results,

```
alct_size_profile.ksh: 16:37:45 Data transform (awk) starting. Input file:
/home/pachaia/RPAS12_Agal/data/d1clss.01
alct_size_profile.ksh: 16:37:45 Transform completed. File: d1clss.01
d1clss.01 processing completed. Exit status: 0
alct_size_profile.ksh: 16:37:45 Data transform (awk) starting. Input file:
/home/pachaia/RPAS12_Agal/data/d1itpt.01
alct_size_profile.ksh: 16:37:45 Transform completed. File: d1itpt.01
d1itpt.01 processing completed. Exit status: 0
alct_size_profile completed with 1 ERRORS: Thu Oct  5 16:37:45 CDT 2006
If the module does not run successfully, the following results,
```

****** STARTING alct_size_profile: Thu Oct 5 16:37:45 CDT 2006 ******

```
******  STARTING alct_size_profile: Thu Oct  5 16:37:45 CDT 2006  ******
alct_size_profile.ksh: 16:37:45 Transform completed. File: d1clss.01
d1clss.01 processing completed. Exit status: 0
alct_size_profile.ksh: 16:37:45 Data transform (awk) starting. Input file:
/home/pachaia/RPAS12_Agal/data/d1dept.01
 ERROR - too many DIFF IDs in current record of Diff Profile File.
```

alct_size_profile.ksh: 16:37:45 Data transform (awk) starting. Input file: /home/pachaia/RPAS12_Agal/data/d1clss.01

```
Diff Profile file name: d1dept.01
 Number of Diffs found:  5
 Maximum number allowed: 4
 DIFF ID string: <  _hh1dif_jh2dif_kh3dif_lh4dif _mh5dif       000000006>
Bad Record:
```

Record number 2

```
1414                     1000000001             _hh1dif_jh2dif_kh3dif_lh4dif_
mh5dif        000000006000
alct_size_profile.ksh: 16:37:45 Transform completed. File: d1dept.01
d1dept.01 processing completed. Exit status: 2
alct_size_profile.ksh: 16:37:45 Data transform (awk) starting. Input file:
/home/pachaia/RPAS12_Agal/data/d1itpt.01
alct_size_profile.ksh: 16:37:45 Transform completed. File: d1itpt.01
d1itpt.01 processing completed. Exit status: 0
alct_size_profile completed with 1 ERRORS: Thu Oct  5 16:37:45 CDT 2006
```

Example for Running Load Batch

Run alcl_size_profile.ksh:

1. Change directory to $ALCHOME/rfx/src.
2. At a UNIX prompt, enter:

```
alcl_size_profile.ksh <input datafile 1> <thread #>
...
```

If the module runs successfully, the following results:

- Log file: Today's log file, 20010309.log, contains the messages described above in the 'Format' passage of the 'Daily log file' section.
- Data: The ALC_SIZE_PROFILE table exists in the Oracle Retail Allocation database and contains the extract records.
- Error File: The program's error file, alcl_size_profile.20010309, contains the standard RETL flow (ending with "All threads complete" and "Flow ran successfully") and no error messages.
- Program status control: The program status control file, alcl_size_profile.status.20010309, does not exist.
- Reject File: No reject files exist.

If the module does not run successfully, the following results:

- Log file: Today's log file, 20010309.log, may not contain the "Program completed successfully…" message.
- Data: The ALC_SIZE_PROFILE table exists in the Oracle Retail Allocation database but may not contain all the records from the profile file interface.
- Error file: The program's error file, alcl_size_profile.20010309, may contain an error message.
- Program status control: The program status control file, alcl_size_profile.status.20010309, may exist.
- Reject file: The reject file, items_not_found.dat, may exist in the $ALCHOME/data directory.
- Bookmark file: The bookmark file, alcl_size_profile.bkm.20010309, does not exist because this module does not utilize restart and recovery.

Re-run the Module:

1. Determine and fix the problem causing the error.
2. Remove the program's status control file.
3. Remove the reject file (if it exists) from the $ALCHOME/data directory.
4. Change directory to $ALCHOME/rfx/src. At a UNIX prompt, enter:

   ```
   % alcl_size_profile.ksh <input datafile 1> <thread #>
   ```

Oracle Retail Allocation Program Reference

This section serves as a reference to the Oracle Retail Allocation program.

By reviewing this section and the section, 'API flat file specification', the retailer should be able to track down to the table and column level, all the extraction data that flows into Oracle Retail Allocation.

Table 10–1 Extraction Data

Program Name	Tables /Files Extracted	Fields Extracted	Target File or Table	Target Field	Field Type	Field Length	NOTES
alcl_size_ profile.ksh	ITEM_ MASTER	item	alc_size_ profile	style	VAR CHAR2 (25)	25	
	profile file	dept		dept	NUMBER (4)	4	
		class		class	NUMBER (4)	4	
		subclass		subclass	NUMBER (4)	4	
		store		store	NUMBER (10)	10	
		size1		size1	VAR CHAR2 (10)	10	
		size2		size2	VAR CHAR2 (10)	10	
		size3		size3	VAR CHAR2 (10)	10	
		size4		size4	VAR CHAR2 (10)	10	
		qty		qty	NUMBER (12,4)	17	

Table 10–1 (Cont.) Extraction Data

Program Name	Tables /Files Extracted	Fields Extracted	Target File or Table	Target Field	Field Type	Field Length	NOTES
ALC_ PLAN.KSH	ITEM_ MASTER	item	ALC_ PLAN	style	VAR CHAR2 (25)	25	
		item_ parent					Use item_ parent as style if Item_ parent_ aggregate_ ind= 'Y'
		item_ grandparent					Use item_ grandparent as Style if item_ grandparent_ aggregate_ ind = 'Y'
		item_ aggregate_ ind					
		item_ parent_ aggregate_ ind					
		item_ grandparent_ aggregate_ ind					
	profile file	dept		dept	NUMBER (4)	4	
		class		class	NUMBER (4)	4	
		subclass		subclass	NUMBER (4)	4	
		store		store	NUMBER (10)	20	
		size1		size1	VAR CHAR2 (10)	48	
		size2		size2	VAR CHAR2 (10)	48	
		size3		size3	VAR CHAR2 (10)	48	

Table 10–1 (Cont.) Extraction Data

Program Name	Tables /Files Extracted	Fields Extracted	Target File or Table	Target Field	Field Type	Field Length	NOTES
		size4		size4	VAR CHAR2 (10)	48	
		qty		qty	NUMBER (12,4)	12	
ALC_RECEIPT_PLAN.KSH	ITEM_MASTER	item	ALC_PLAN	style	VAR CHAR2 (25)	25	
		item_parent					Use item_parent as style if Item_parent_aggregate_ind= 'Y'
		item_grand parent					Use item_grand parent as Style if item_grandparent_aggregate_ind = 'Y'
		item_aggregate_ind					
		item_parent_aggregate_ind					
		item_grandparent_aggregate_ind					
	profile file	dept		dept	NUMBER (4)	4	
		class		class	NUMBER (4)	4	
		subclass		subclass	NUMBER (4)	4	
		store		store	NUMBER (10)	20	
		size1		size1	VAR CHAR2 (10)	48	
		size2		size2	VAR CHAR2 (10)	48	

Table 10–1 (Cont.) Extraction Data

Program Name	Tables /Files Extracted	Fields Extracted	Target File or Table	Target Field	Field Type	Field Length	NOTES
		size3		size3	VAR CHAR2 (10)	48	
		size4		size4	VAR CHAR2 (10)	48	
		qty		qty	NUMBER (12,4)	12	

Application Programming Interface (API) Specification

File Layout

Plan Data File Layout

Plan data is required in Allocation for allocating items to the store based on plan requirements at the location. The data is extracted by the predictive application server and provided to the allocation interfaces as a flat file.

This segment describes the file format that Predictive Application System provides for uploading plan data into RMS database for Allocation.

File Name: p1*prodlevel.NN*

Example: p1scls.01 (subclass, domain 01)

- 1234567890123456789012345678901234567890123456789012345678901234567890123456 7890123456789012345678901234567890123
- 141410001000 1000000014 _CCOLOR31_SMEDIUM
- 20051225000000137500

Table 10–2 Plan Data File Layout

Field Name	Start Position	Width	Format	Content
Product ID	1	25 char	Alpha	141410001000
Location ID	26	20 char	Alpha	10000000014
Diff IDs	46	48 char	Alpha	_CCOLOR31_ SMEDIUM
EOW Date	94	8 char	Alpha	20051225
Quantity	102	12 char	Numeric	000000137500

- The file name should start with p1 followed by four characters for product level and the domain number. The four product level acceptable are:
  - - itpt - for item
  - - scls - for subclass
  - - clss - for class
  - - dept - for department

- The domain ID should be numeric.
- The Product ID, Location ID and Diff IDs fields are left justified and blank filled.
- The number of separate Diffs in the Diff IDs field is in the range: 0-4. The first character of each Diff is an "_" (underscore) and the second character is the Diff Type. No underscore characters are present in the Diff ID field other than the character that immediately precedes each separate Diff Type within the field. Each Diff in the Diff IDs field is lesser than 12 characters in length, including the leading underscore character and the Diff Type.
- The EOW Date field is in the format YYYYMMDD.
- The Quantity field is a right-justified, zero-padded numeric and the decimal point is omitted, but the quantity has a 4-digit decimal fraction part (e.g. 13.75 would appear in the record as 000000137500).
- Total length of each record is 113.
- When uploading data the system updates the quantity if the record exists for the hierarchy/location/Diff_id/EOW data combination or it appends the record into the tables.

Receipt Plan Data File Layout

Receipt plan data is required in Allocation for allocating items to the store based on receipt plan requirements at the location. The data is extracted by the predictive application server and provided to the allocation interfaces as a flat file.

This segment describes the file format that Predictive Application System provides for uploading receipt plan data into RMS database for Allocation.

File Name: p1*prodlevel.NN*

Example: p1scls.01 (subclass, domain 01)

- 12345678901234567890123456789012345678901234567890123456789012345678901234567890123456
  7890123456789012345678901234567890123
- 141410001000 1000000014 _CCOLOR31_SMEDIUM
- 20051225000000137500

Table 10–3 Receipt Plan Data File Layout

Field Name	Start Position	Width	Format	Content
Product ID	1	25 char	Alpha	141410001000
Location ID	26	20 char	Alpha	10000000014
Diff IDs	46	48 char	Alpha	_CCOLOR31_ SMEDIUM
EOW Date	94	8 char	Alpha	20051225
Quantity	102	12 char	Numeric	000000137500

- The file name should start with p1 followed by four characters for product level and the domain number. The four product level acceptable are:
  - – - itpt - for item
  - – - scls - for subclass
  - – - clss - for class

  - - dept - for department

- The domain ID should be numeric.
- The Product ID, Location ID and Diff IDs fields are left justified and blank filled.
- The number of separate Diffs in the Diff IDs field is in the range: 0-4. The first character of each Diff is an "_" (underscore) and the second character is the Diff Type. No underscore characters are present in the Diff ID field other than the character that immediately precedes each separate Diff Type within the field. Each Diff in the Diff IDs field is lesser than 12 characters in length, including the leading underscore character and the Diff Type.
- The EOW Date field is in the format YYYYMMDD.
- The Quantity field is a right-justified, zero-padded numeric and the decimal point is omitted, but the quantity has a 4-digit decimal fraction part (e.g. 13.75 would appear in the record as 000000137500).
- Total length of each record is 113.
- When uploading data the system updates the quantity if the record exists for the hierarchy/location/Diff_id/EOW data combination or it appends the record into the tables.

Size Curve File Layout

Allocation application allocates quantities at the style-color level for fashion merchandise like Blue t-shirts; Dark Blue pants etc and then distributes the allocated quantity according to the size based on the size curve for the location. The size curve data can be inputted into the application directly using the GUI interface in the application or through upload from data provided by the predictive application server.

This section describes in detail the file format structure for the upload.

File Name: d*Xprodlevel.NN*

Example: File Name: d1scls.01 (diff 1, subclass, domain 01)

Record:

- 12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345
- 100045078 1000000002 CCOLOR01_S30x32 000000137500

Table 10–4 Size Curve File Layout

Field Name	Start Position	Width	Format	Content
Product ID	1	25 char	Alpha	100045078
Location ID	26	20 char	Alpha	1000000002
Diff IDs	46	48 char	Alpha	_ CCOLOR01_ S30x32
Quantity	94	12 char	Numeric	000000137500

- The file name should start with letter d, X is diff number being sent followed by four characters for product level and the domain number. The four product level acceptable are:

  - - itpt - for item
  - - scls - for subclass
  - - clss - for class
  - - dept - for department
- The domain ID should be numeric.
- The Product ID, Location ID and Diff IDs fields are left justified and blank filled.
- The number of separate Diffs in the Diff IDs field is in the range: 0-4. The first character of each Diff is an "_" (underscore) and the second character is the Diff Type. No underscore characters is present in the Diff ID field other than the character that immediately precedes each separate Diff Type within the field. Each Diff in the Diff IDs field is lesser than 12 characters in length, including the leading underscore character and the Diff Type.
- The Quantity field is a right-justified, zero-padded numeric and the decimal point is omitted, but the quantity has a 4-digit decimal fraction part (e.g. 13.75 would appear in the record as 000000137500).
- Total length of each record is 105.
- When uploading data the system updates the quantity if the record exists for the hierarchy/location/Diff_id/EOW data combination or it appends the record into the tables.

Schema file (alcl_size_profile.schema)

This section describes the RETL schema file (alcl_size_profile.schema) used in the RETL script that loads the Curve export file into Oracle Retail.

Allocation's ALC_SIZE_PROFILE table:

```
<RECORD type="fixed" len="115" final_delimiter="0x0A">
<!-- start pos 5  --> <FIELD name="CLASS" len="4" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 9  --> <FIELD name="SUBCLASS" len="4" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 13 --> <FIELD name="ITEM" len="25" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 38 --> <FIELD name="STORE" len="20" datatype="string"
nullable="false"/>
<!-- start pos 58 --> <FIELD name="DIFF_TYPE_1" len="1" datatype="string"
nullable="false" nullvalue=""/>
<!-- start pos 59 --> <FIELD name="DIFF1" len="10" datatype="string"
nullable="false" nullvalue=""/>
<!-- start pos 69 --> <FIELD name="DIFF_TYPE_2" len="1" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 70 --> <FIELD name="DIFF2" len="10" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 80 --> <FIELD name="DIFF_TYPE_3" len="1" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 81 --> <FIELD name="DIFF3" len="10" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 91 --> <FIELD name="DIFF_TYPE_4" len="1" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 92 --> <FIELD name="DIFF4" len="10" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 102 --> <FIELD name="QTY" len="14" datatype="dfloat"
nullable="false"/>
<!-- end pos 114  -->
```

```
</RECORD>
<!-- start pos 1 --> <FIELD name="DEPT" len="4" datatype="string" nullable="true"
nullvalue=""/>
```

Schema file (alcl_plan.schema)

This section describes the RETL schema file (alcl_plan.schema) used in the RETL script that loads the plan export file into Oracle Retail.

Allocation's ALC_PLAN table.

```
<RECORD type="fixed" len="123" final_delimiter="0x0A">
<!-- start pos 5  --> <FIELD name="CLASS" len="4" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 9  --> <FIELD name="SUBCLASS" len="4" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 13 --> <FIELD name="ITEM_ID" len="25" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 38 --> <FIELD name="STORE" len="20" datatype="string"
nullable="false"/>
<!-- start pos 58 --> <FIELD name="DIFF_TYPE_1" len="1" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 59 --> <FIELD name="DIFF1_ID" len="10" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 69 --> <FIELD name="DIFF_TYPE_2" len="1" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 70 --> <FIELD name="DIFF2_ID" len="10" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 80 --> <FIELD name="DIFF_TYPE_3" len="1" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 81 --> <FIELD name="DIFF3_ID" len="10" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 91 --> <FIELD name="DIFF_TYPE_4" len="1" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 92 --> <FIELD name="DIFF4_ID" len="10" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 102 --> <FIELD name="EOW_DATE" len="8" datatype="date"
nullable="false"/>
<!-- start pos 110 --> <FIELD name="QTY" len="14" datatype="dfloat"
nullable="false"/>
<!-- end pos 122  -->
</RECORD>
<!-- start pos 1 --> <FIELD name="DEPT" len="4" datatype="string" nullable="true"
nullvalue=""/>
```

Schema File (alcl_receipt_plan.schema)

This section describes the RETL schema file (alcl_receipt_plan.schema) used in the RETL script that loads the receipt plan export file into Oracle Retail.

Allocation's ALC_RECEIPT_PLAN table:

```
<RECORD type="fixed" len="123" final_delimiter="0x0A">
<!-- start pos 5  --> <FIELD name="CLASS" len="4" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 9  --> <FIELD name="SUBCLASS" len="4" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 13 --> <FIELD name="ITEM_ID" len="25" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 38 --> <FIELD name="STORE" len="20" datatype="string"
nullable="false"/>
<!-- start pos 58 --> <FIELD name="DIFF_TYPE_1" len="1" datatype="string"
```

```
nullable="true" nullvalue=""/>
<!-- start pos 59 --> <FIELD name="DIFF1_ID" len="10" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 69 --> <FIELD name="DIFF_TYPE_2" len="1" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 70 --> <FIELD name="DIFF2_ID" len="10" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 80 --> <FIELD name="DIFF_TYPE_3" len="1" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 81 --> <FIELD name="DIFF3_ID" len="10" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 91 --> <FIELD name="DIFF_TYPE_4" len="1" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 92 --> <FIELD name="DIFF4_ID" len="10" datatype="string"
nullable="true" nullvalue=""/>
<!-- start pos 102 --> <FIELD name="EOW_DATE" len="8" datatype="date"
nullable="false"/>
<!-- start pos 110 --> <FIELD name="QTY" len="14" datatype="dfloat"
nullable="false"/>
<!-- end pos 122  -->
</RECORD>
<!-- start pos 1 --> <FIELD name="DEPT" len="4" datatype="string" nullable="true"
nullvalue=""/>
```

Oracle Retail Extract, Transform, and Load for Receipt and Plan

RETL scripts are required to support receipt plan logic (what the store is expected to own) at the store/week level. This rule provides fashion retailers the option to choose different plans coming from different source data feeds.

Large size fashion retailers can use Assortment Planning data as the pre-season source to determine store or warehouse needs for seasonal items. The data file sent from the Assortment Planning system is used to generate the gross need in the Allocation system in order to create pre-allocations.

Once the selling season begins, retailers have the option to switch to a different rule such as Plan, Forecast or Historical data to generate store demand.

Script Names

- alct_receipt_plan.ksh
- alct_receipl_plan.ksh

Table Name

- alc_receipt_plan

Oracle Retail Extract, Transform, and Load for Size Profile Optimization Data

Allocation users have the option to select a specified store size profile to be used for the Allocation. Using the RPAS Store Size Profile Optimization application, users have the capability to create seasonal store size profiles and multiple store size profiles created in SPO (called GIDs). These are displayed to the Allocation user as options to be used.

- Depending on what is being allocated and expected arrival date in the stores, the Allocation user has the option to view and select the desired store size profile date to be used.
- All item and locations use the same store size profile data per allocation. There cannot be unique buy item/location records within a single allocation.

- SPO assigns a numeric generation ID number (GID) to specifically created store size profile data. This ID, along with a user defined name should be displayed in the Allocation user interface.
- Only those GIDs populated from SPO to Allocation are displayed in the user interface.
- The retailer is responsible for updating the Allocation table on a frequent basis or as needed.

SPO GID text files (spo_gid_label.txt) are passed along with the batch of Size Profile Hierarchy dat file. The text file is used as the GID for that batch of dat file. Running RETL for SPO imports data to three tables after extraction.

The RETL for SPO data file format is as follows:

<Beginning of file>

<GID>

<GID_DESC>

<End of File>

The following are examples:

GID1

Winter 2014

Table 10–5 RETL for SPO Data Physical Tables

Table Head	Description
ALC_GID_HEADER	The ALC_GID_HEADER table holds all generation ID descriptions.
ALC_GID_PROFILE	The ALC_GID_PROFILE table holds all generation ID profile IDs.
ALC_SIZE_PROFILE	The ALC_SIZE_PROFILE table holds all size profiles at Style/Color, Style, Subclass, Class and Department levels.

Limitations of Oracle Retail Extract, Transform, and Load Programs

The three programs that exist for receipt plan and plan and size profile have the following limitations:

- The diff type is supported to a maximum of 1 character length.
- The diff id is supported to a maximum of 10 character length.

11

Java Batch Process

This chapter provides an overview of the batch processes of Oracle Retail Allocation. It also provides information about the functions of the batch processes, the Java packages associated with the batches, and how to execute the Java-based batches.

Batch Processing Overview

Allocation contains a set of batch processes that are run in Java. Broadly, the batch process falls under three categories:

- Schedule Allocation batch
- Daily Cleanup batch
- Purge batches
- Rule Level On Hand (RLOH) batches

ScheduledAllocationBatchClient.java creates the child allocations for parent allocations that are scheduled for the day.

SessionCleanUpBatchClient.java deletes data from the temporary tables used by the Allocations and Calculation engine.

Purge batches delete Allocation and Worksheet data from the Allocation tables, which were created before a certain time period.

For RLOH, there are six batch update processes that share the same java batch file; InventorySnapshotBatchClient.java.

Note the following general characteristics of Oracle Retail Allocation's java batch process:

- It is not accessible through a graphical user interface (GUI).
- It is scheduled by the retailer.
- It is designed to process large volumes of data, depending on the circumstances and process.

Java Batch Names and Java Packages

The following table describes Oracle Retail Allocation's batch processes and its associated Java packages:

Table 11–1 Allocation's Batch Process and associated Java Package

Batch Name	Batch Process	Package
Schedule Allocation Batch	ScheduledAllocationBatchClient.java	oracle.retail.apps.alc.batch.client
Daily Cleanup Batch	SessionCleanUpBatchClient.java	oracle.retail.apps.alc.batch.client
Purge Batches	PurgeBatchRunnable.java	oracle.retail.apps.alc.batch.client
RLOH Batch Update	InventorySnapshotBatchClient.java	oracle.retail.apps.alc.batch.client

Running a Java-based Batch Process

To run a Java-based batch process, Oracle Retail provides sample shell scripts (.sh files) and batch files (.bat files). These sample shell scripts must be modified according to the retailer's installation. They perform the following internally:

- Set up the Java runtime environment before the Java process is run.
- Trigger the Java batch process.

Scheduler and Command Line

If the retailer uses a scheduler, arguments are placed into the scheduler.

If the retailer does not use a scheduler, arguments must be passed in at the command line.

For UNIX systems, the Java process is scheduled through an executable shell script (.sh file).

> **Note:** The `AllocScheduleBatch.ksh` and `AlcDailyCleanUp.ksh` batches can be run by an external scheduling system such as APPWORX or a simple UNIX CRON job.

Running the Schedule Allocation Batch

Installation and build scripts create the required user for running the batch in the wallet. There is no way you can cross check to determine whether the user is created inside the wallet other than running the batch scripts. However, you can see if the wallet is present in the environment by checking the wallet location. The wallet location is present in batch.properties file. The wallet is created with a user_id, password and an alias name.

Once the wallet is created the csm.wallet.path key in batch.properties file should be updated.

Only those users who have their role mapped to the SYSTEM_ADMINISTRATOR_JOB enterprise role in LDAP, have the privilege to execute the Schedule Allocation batch script. During installation, Allocation creates the SYSTEM_ADMINISTRATOR user, by default, in the Retail Wallet, which is mapped to the SYSTEM_ADMINISTRATOR_JOB enterprise role in LDAP. An alias for any new user mapped to SYSTEM_ADMINISTRATOR_JOB role in LDAP has to be created in the Wallet in order to execute the Schedule Allocation batch script.

> **Note:** Use the `save_credential.sh` script to create a new user in the Wallet. For more information on instructions to run the `save_credential.sh script` to add a new user, see the *Oracle Retail Allocation Installation Guide*.

The batch.properties file exposes a few configuration parameters related to concurrency management or parallel execution which need to be tuned by the retailer based on the volume of transactions. The concurrent processing in batch is implemented leveraging the standard Java Executor service APIs. The sample file with default configurations will be made available and need to be modified by the retailer to suit to their requirements.

The section below describes the properties that can be configured.

- initialThreadLimit: Initial number of threads in the pool that are available to create child allocations. The default value is 5.
- maxThreadLimit: Maximum number of threads that can be allowed in the pool. The default value is 10.
- queueLimit: Size of queue of pending tasks to create child. The default value is 1.
- providerUrl: Url of the server module (for example, t3://<weblogic host>:<port>). This parameter has to be configured by the retailer to point to the WebLogic Server on which Asynchronous application instance is deployed.
- csm.wallet.partition.name: Partition name in the wallet (for example, alloc13)
- csm.wallet.path: Location of Wallet

1. Login to the application server machine using <username>/<password>.
2. Navigate to the batch folder. If the batch folder is not found, the batch installation did not occur properly. In the batch folder, verify that the `AllocScheduleBatch.ksh` file is present.
3. Run the `AllocScheduleBatch.ksh` batch using the following command:

   ```
   ksh AllocScheduleBatch.ksh <systemadministratoralias>
   ```

   The batch runs by taking the batch user from wallet.

Running the Daily Cleanup Batch

Take the following steps to run the Daily Cleanup batch:

1. Login to the application server machine using <username>/<password>.
2. Navigate to the batch folder. In the batch folder, verify that the `AlcDailyCleanUp.ksh` file is present.
3. Run the AlcDailyCleanUp.ksh batch using the following command:

   ```
   ksh AlcDailyCleanUp.ksh <systemadministratoralias>
   ```

   The batch runs by taking the batch user from wallet.

Running the Purge Batches

Use the following steps to run the Purge batches:

1. Login to the application server machine using <username>/<password>.

2. Navigate to the batch folder. In the batch folder, verify that the AlcPurgeAlloc.ksh and AlcPurgeWksht.ksh files are present.
3. Run the both batch processes using the following command:

   ```
   ksh AlcPurgeAlloc.ksh <systemadministratoralias> PURGE_ALLOC
   ksh AlcPurgeWksht.ksh <systemadministratoralias> PURGE_WORKSHEET
   ```

   The batch runs by taking the batch user from wallet.

Running the Rule Level On Hand Batch

Take the following steps to run the RLOH batch:

1. Login to the application server machine using <username>/<password>. Once logged in, the default folder is /home/alcbatch.
2. Before running the batch, make sure that all the corresponding profile properties are set. For that run the profile file first. Go to the **Profiles** folder inside alcbatch.
3. If there are multiple environments, there are separate profile files for every machine (for example, QA, DEV, TEST). Make sure to identify the right profile file here. Most likely it will be the name of the environment, run the profile file - ./alc132Linuxdev (for example, Dev 13.2 Env).
4. After running the profile successfully, go back to alcbatch. There are separate folders for every machine's batch under the **alcbatch** folder. Go to the current machine's folder. (Most likely the folder name would be same as your profile file name, in this case alc132Linuxdev).
5. Run the following scripts inside the batch folder in the following order:
   - ksh AlcSnapshotSOH.ksh <BatchUserAlias>
   - ksh AlcSnapshotOnOrder.ksh <BatchUserAlias>
   - ksh AlcSnapshotAllocIn.ksh <BatchUserAlias>
   - ksh AlcSnapshotCrosslink.ksh <BatchUserAlias>
   - ksh AlcSnapshotAllocOut.ksh <BatchUserAlias>
   - ksh AlcSnapshotCustomerOrder.ksh <BatchUserAlias>

Summary of Executable Files

The following table describes the executable shell scripts and batch files:

Table 11–2 Scripts to initiate the deletion process

Executable Shell Scripts (UNIX)	Executable Batch File For Windows	Description
AllocScheduleBatch.ksh	No batch file is available	Triggers the schedule batch client.
AllocBatch.ksh	No batch file is available	Configures the environment variables sourced by other batch scripts. This script is not to be run/scheduled in a stand alone mode.
AlcSnapshotSOH.ksh	No batch file is available	
AlcSnapshotOnOrder.ksh	No batch file is available	

Table 11–2 (Cont.) Scripts to initiate the deletion process

Executable Shell Scripts (UNIX)	Executable Batch File For Windows	Description
AlcSnapshotAllocOut.ksh	No batch file is available	
AlcSnapshotCustomerOrder.ksh	No batch file is available	
AlcSnapshotCrosslink.ksh	No batch file is available	
AlcSnapshotAllocIn.ksh	No batch file is available	
AlcDailyCleanUp.ksh	No batch file is available	Deletes data from the temporary tables.
AlcPurgeAlloc.ksh	No batch file is available	Deletes old Allocations from database table
AlcPurgeWksht.ksh	No batch file is available	Deletes old Worksheets from database table

AllocScheduleBatch Process Batch Design

The Allocation Auto Scheduler creates child allocations on pre-defined days of the week set by the Allocation user within the user interface. These allocations are created from an existing parent allocation. The auto creation of the child allocations must be called daily via a batch process at a scheduled time, set by the system administrator.

This process needs to be scheduled to run every day (using an external scheduling framework like APPWORKS or UNIX CRON job).

Usage

The following command runs the AllocScheduleBatch job:

```
AllocScheduleBatch.ksh userAlias
```

Detail

This script is present under the $ALLOCHOME/batch folder.

Log File

log4j.xml is present under $ALLOCHOME/properties folder. This file is edited to specify desired log file location and name. To perform this action, change the value against param with name="file" in log4j.xml. Make sure that folder is already present on the file system and the batch user has write permission. Default value is set to ../logs/alloc133.log.

Properties File

The default batch properties file is present under $ALLOCHOME/properties/oracle/retail/alloc/batch.properties.

The properties below are defined. The default value may be edited.

- initialThreadLimit initial number of threads in the pool that are available to create child allocations.   The default value is 5.

- maxThreadLimit maximum number of threads that can be allowed in the pool. The default value is 10.
- queueLimit size of queue of pending tasks to create child. The default value is 1.
- initialContextFactory specifies the JNDI context factory class (this should not be changed).
- providerUrl url of the server module (e.g t3://<weblogic host>:<port>).   This parameter has to be configured by the retailer to point to the WebLogic Server on which the asynchronous application instance is deployed.
- csm.wallet.partition.name is the partition name in the wallet that stores the credentials to authenticate batch user on WebLogic. For example, alloc13
- csm.wallet.path is the path of the wallet file that stores WebLogic credentials.

Configuration

$ALLOCHOME/batch/AllocBatch.ksh should be edited by the retailer to specify appropriate value of following environment variables

- ALLOCHOME: directory where batch client in installed
- JAVA_HOME: directory where JDK is installed

Assumptions and Scheduling Notes

This job should be scheduled to run every day at the same time.

AlcDailyCleanUp Process Batch Design

Allocation has a number of temporary tables that store intermediate data while creating allocations and while performing calculations. The Daily Cleanup batch process deletes data from these temporary tables. Run this batch immediately after you run the Schedule Allocation batch.

This process should be scheduled to run every day (using an external scheduling framework like APPWORKS or UNIX CRON job).

Make sure you run this process while all users are offline from the system.

Usage

The following command runs the AlcDailyCleanUp job:

```
AlcDailyCleanUp.ksh <BatchUserAlias>
```

Detail

This script is present under the $ALLOCHOME/batch folder.

The temporary tables which are impacted by the AlcDailyCleanUp process are as follows:

Allocation session tables:

- ALC_SESSION_SIZE_PROFILE_RATIO
- ALC_SESSION_SIZE_PROFILE
- ALC_SESSION_QUANTITY_LIMITS

- ALC_SESSION_ITEM_LOC_EXCL
- ALC_SESSION_ITEM_LOC
- ALC_SESSION_GID_PROFILE_LIST
- ALC_SESSION_GID_PROFILE

Worksheet session tables:

- ALC_WORK_SESSION_ITEM_LOC
- ALC_WORK_SESSION_ITEM_ALL
- ALC_WORK_SESSION_ITEM

Temporary tables:

- ALC_LOAD_TEMP

  alc_calc_destination_temp

  alc_calc_need_temp

  alc_calc_rloh_temp

  alc_calc_qty_limits_temp

  alc_calc_rloh_item_temp

  alc_merch_hier_rloh_temp

  alc_calc_source_temp

  alc_calc_need_dates_temp

Allocation approval tables:

- ALC_SYNC_HEADER_TEMP
- ALC_SYNC_DETAIL_TEMP

AlcPurgeAlloc AlcPurgeWksht Batch Processes Design

Allocation has a number of temporary tables that store intermediate data while creating allocations and while performing calculations. The Purge batch process deletes data from these temporary tables. Run this batch immediately after you run the Schedule Allocation batch. This process should be scheduled to run every day using an external scheduling framework. Make sure you run this process while all users are offline from the system.

Along with the above mentioned capability, this batch also allows provides for deletion of older allocations and worksheets created as a part of the Allocation application.

Usage

The following command runs the job:

```
AlcPurgeAlloc.ksh <systemadministratoralias> PURGE_ALLOC

AlcPurgeWksht.ksh <systemadministratoralias> PURGE_WORKSHEET
```

Details:

Allocation deletions are driven by the system option ALLOCATION_RETENTION_DAYS. Allocations exceeding the retention parameter become purge candidates as follows:

- Scheduled Allocations Parents are deleted when their scheduled end date is greater than the allocation retention days parameter.
- Allocations that are linked to RMS allocations in the ALC_XREF table are deleted when the RMS allocations they are linked to no longer exist in RMS.
- Allocations that are not linked to RMS allocations in the ALC_XREF table are deleted when they have not been modified (ALC_ALLOC.LAST_UPDATE_DATE) for ALC_SYSTEM_OPTIONS.TP_ALLOC_RETENTION_DAYS days.
- Allocations in Deleted status - user deleted through the UI.

Worksheets not associated to an allocation (WK worksheets) are deleted based on this setting. Worksheets associated to an allocation (WD worksheets) are deleted when the allocation they are related to is deleted (they follow Allocation deletion). Worksheet deletion is driven by a system option, WORKSHEET_RETENTION_DAYS. Worksheets purge criteria is as follows:

Worksheets not tied to an allocation (type = WK) are deleted when they are not be modified (ALC_WORK_HEADER.UPDATED_DATE) for TP_WORKSHEET_RETENTION_DAYS days.

Rule Level On Hand Pre-Aggregation Inventory Snapshot Batch Design

This batch process addresses the most significant performance issue within the Allocation product, the rule level on hand (RLOH) logic. This functionality requires current and future inventory lookups for potentially entire departments.

Inventory is currently only held in RMS at the transaction level item level. Departments in RMS can have tens of thousands of items under them. Multiply this by the hundreds of locations that can be on an allocation and RLOH can easily end up needing to retrieve inventory for millions of item/location combinations.

Usage

Six separate executables are called by one java batch process. The executables and the commands to run them are as follows:

- AlcSnapshotOnOrder.ksh

  ```
  ./AlcSnapshotOnOrder.ksh <BatchUserAlias>
  ```

- AlcSnapshotCrosslink.ksh

  ```
  ./AlcSnapshotCrosslink.ksh <BatchUserAlias>
  ```

- AlcSnapshotAllocIn.ksh

  ```
  ./AlcSnapshotAllocIn.ksh <BatchUserAlias>
  ```

- AlcSnapshotSOH.ksh

  ```
  ./AlcSnapshotSOH.ksh <BatchUserAlias>
  ```

- AlcSnapshotAllocOut.ksh

  ```
  ./AlcSnapshotAllocOut.ksh <BatchUserAlias>
  ```

- AlcSnapshotCustomerOrder.ksh

  ```
  ./AlcSnapshotCustomerOrder.ksh <BatchUserAlias>
  ```

A remote interface can be called for each batch

```
public interface IInventorySnapshotCoreRemote {

    public void createItemLocSOHSnapshot() throws AllocRemoteException;

    public void createOnOrderSnapshot() throws AllocRemoteException;

    public void createAllocInSnapshot() throws AllocRemoteException;

    public void createCrosslinkInSnapshot() throws AllocRemoteException;

    public void createAllocOutSnapshot() throws AllocRemoteException;

    public void createCustomerOrderSnapshot() throws AllocRemoteException;
}
```

Each method lines up with the appropriate PL-SQL function.

Detail

Retrieving inventory requires accessing four very large RMS tables:

- ITEM_LOC_SOH - current inventory and components of future inventory
- ORDLOC - on order component of future inventory
- ALLOC_DETAIL - allocation in component of future inventory
- TSFDETAIL - crosslink transfer component of future inventory

To improve RLOH performance, four new subclass level aggregated tables are created for use by Allocation

Table 11–3 RLOH Aggregated Tables

Table	Candidate Key	Source Tables
SUBCLASS_ITEM_LOC_SOH_EOD	Dept Class Subclass Loc	ITEM_LOC_SOH ITEM_MASTER
SUBCLASS_ON_ORDER_EOD	Dept Class Subclass Loc On_order_date	ORDLOC ORDHEAD ITEM_MASTER PACKITEM_BREAKOUT
SUBCLASS_ALLOC_IN_EOD	Dept Class Subclass Loc Alloc_in_date	ALLOC_DETAIL ALLOC_HEADER ITEM_MASTER PACKITEM_BREAKOUT ORDHEAD TSFHEAD

Table 11–3 (Cont.) RLOH Aggregated Tables

Table	Candidate Key	Source Tables
SUBCLASS_CROSSLINK_EOD	Dept	TSFHEAD
	Class	TSFDETAIL
	Subclass	ITEM_MASTER
	Loc	PACKITEM_BREAKOUT
SUBCLASS_ALLOC_OUT_EOD	Dept	ALLOC_DETAIL
	Class	ALLOC_HEADER
	Subclass	ITEM_MASTER
	Loc	ORDHEAD
	Alloc_Out_Date	PACKITEM_BREAKOUT
ALC_SUBCLASS_CUST_ORDER_EOD	Dept	TSFHEAD
	Class	TSFDETAIL
	Subclass	ITEM_MASTER
	Loc	PACKITEM_BREAKOUT

These tables are populated by the ALC_HIER_LVL_INV_SNAPSHOT_SQL package (called by a batch program) nightly.

Package Details

The package that needs to be called is ALC_HIER_LVL_INV_SNAPSHOT_SQL.

```
SQL> desc ALC_HIER_LVL_INV_SNAPSHOT_SQL
FUNCTION ROLLUP_ALLOC_IN RETURNS NUMBER
Argument Name                  Type                   In/Out Default?
------------------------------ ---------------------- ------ --------
O_ERROR_MESSAGE                VARCHAR2               IN/OUT
FUNCTION ROLLUP_CROSSLINK_IN RETURNS NUMBER
Argument Name                  Type                   In/Out Default?
------------------------------ ---------------------- ------ --------
O_ERROR_MESSAGE                VARCHAR2               IN/OUT
FUNCTION ROLLUP_IL_SOH RETURNS NUMBER
Argument Name                  Type                   In/Out Default?
------------------------------ ---------------------- ------ --------
O_ERROR_MESSAGE                VARCHAR2               IN/OUT
FUNCTION ROLLUP_ON_ORDER RETURNS NUMBER
Argument Name                  Type                   In/Out Default?
------------------------------ ---------------------- ------ --------
O_ERROR_MESSAGE                VARCHAR2               IN/OUT
------------------------------ ---------------------- ------ --------
ROLLUP_ALLOC_OUT (FUNCTION)<return value>  NUMBER      OUT
ROLLUP_ALLOC_OUT
O_ERROR_MESSAGE                VARCHAR2               IN/OUT
------------------------------ ---------------------- ------ --------
ROLLUP_CUSTOMER_ORDER (FUNCTION) <return value>  NUMBER   OUT
ROLLUP_CUSTOMER_ORDER
O_ERROR_MESSAGE                VARCHAR2               IN/OUT
------------------------------ ---------------------- ------ --------
```

There are six functions in the package. Each of the four functions in the package should be called by its own batch program.

- AlcSnapshotSOH
- AlcSnapshotAllocIn
- AlcSnapshotOnOrder
- AlcSnapshotCrosslink
- AlcSnapshotAllocOut
- AlcSnapshotCustomerOrder

Implementation

There are six different batch executables, each executable calling the appropriate method from the above ALC_HIER_LVL_INV_SNAPSHOT_SQL package. Clarifications on the batch functionality:

- Each batch should state success or failure, whether an exception is caught or not.
- There is no need for restart recovery, intermittent commits, or threading.
- Login validation standard logic used by the scheduled alloc program should be applied here as well.
- The programs are run sequentially. The correct order is documented in the Merchandising batch schedule and controlled by whichever scheduling tool used at a particular customer.
- There are no special security requirements for the program. Any user who can log into the Allocation product can have the ability to run the batch processes.

12

Internationalization

Internationalization is the process of creating software that can be translated easily. Changes to the code are not specific to any particular market. Allocation has been internationalized to support multiple languages.

This section describes configuration settings and features of the software that ensure that the base application can handle multiple languages.

Translation

Translation is the process of interpreting and adapting text from one language into another. Although the code itself is not translated, components of the application that are translated may include the following, among others:

- Graphical user interface (GUI)
- Error messages

The following components are not usually translated:

- Documentation (Online Help, Release Notes, Installation Guide, User Guide, Operations Guide)
- Batch programs and messages
- Log files
- Configuration Tools
- Reports
- Demonstration data
- Training Materials

The user interface for Allocation has been translated into:

- Chinese (Simplified)
- Chinese (Traditional)
- Croatian
- Dutch
- French
- German
- Greek
- Hungarian

- Italian
- Japanese
- Korean
- Polish
- Portuguese (Brazilian)
- Russian
- Spanish
- Swedish
- Turkish

Setting the User Language

To set the language Allocation displays in, set the user language. Choose Preferences from the drop-down menu under the user name, then choose Language in the taskbar. There are two language settings you can choose:

- Default: This will permanently change the language for this user. All subsequent sessions will be in this language.
- Current Session: This will change the language only for this session. It will revert back to the Default the next time this user logs in.

When finished making your selections, click the **Save** button.

Setting Date, Time, and Number Formats

Allocation allows the user to see dates and times in a format appropriate for their own locale, regardless what the data is stored in. Set the language Allocation displays in, set the user language. Choose Preferences from the drop-down menu under the user name, then choose Regional in the taskbar. There are multiple settings you can choose:

- Territory: Choose a territory from this list. Allocation will automatically pick Date, Time, and Number formats appropriate for that territory.
- Date Format: Choose an option from this menu to select a date format that is different from the default for the selected Territory.
- Time Format: Choose an option from this menu to select a time format that is different from the default for the selected Territory.
- Number Format: Choose an option from this menu to select a number format that is different from the default for the selected Territory.
- Time Zone: Choose an option from this menu to select your time zone.

When finished making your selections, click the **Save** button.

Translations

Most user interface and message translations are stored in xlf files. When you select a different language from the Preferences screen, Allocation will choose the correct xlf file for that language.

Some translations for some drop-down menus are stored on four database tables. The tables are RTC_LOOKUP_VALUES_TL, RTC_LOOKUP_TYPES_TL, RAF_FACET_

ATTRIBUTE_CFG_TL, and RAF_NOTIFICATION_TYPE_TL. These tables are multilingual; all languages of these strings (English as well as all translations) are stored in the same place.

13

Allocation ReSTful Web Service Implementation

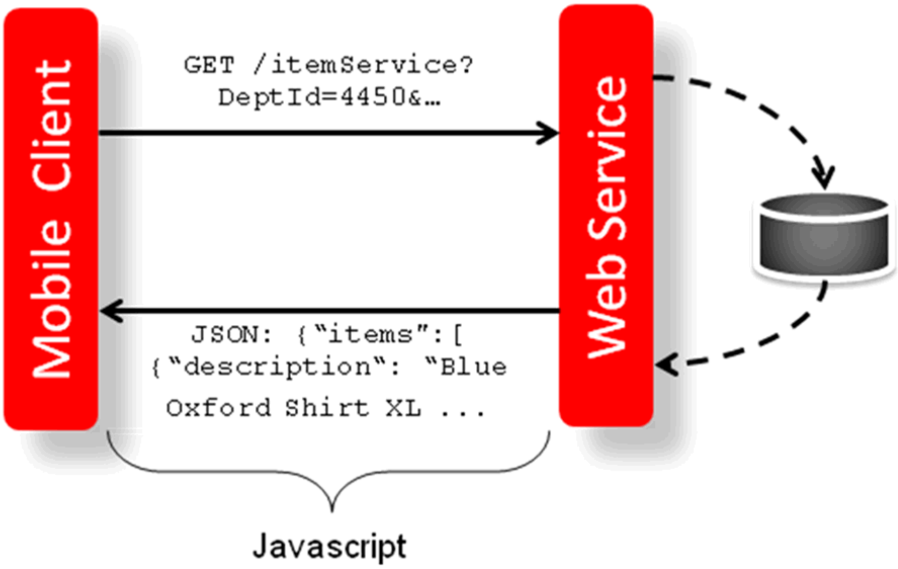
This chapter gives an overview about the Allocation ReSTful Web Service Implementation and API designs used in the Allocation environment. Retailers can access back-end functionality in Retail applications by calling the applications' ReSTful Web Services.

Refer to `http://www.oracle.com/technetwork/articles/javase/index-137171.html` to learn more about ReST as an architectural style applied to building Web services.

Introduction

The Allocation ReSTful Web Services are built based upon a mobile application point of view as defined by Oracle Retail. The services are designed to support the mobile application functionality defined in this point of view.

Figure 13–1 Allocation ReSTful Web Services



Other Uses

The main objective of the ReSTful Web Services is to support the Merchandising Mobile Application; therefore, some of the services functionalities may not be useful for general use. The services can be used as is for client's custom mobile application.

The ReSTful Web Services Java code cannot be customized, but the MBL PL/SQL functions and object types can be customized for client use. In addition you can use the RSE tool to create your own custom services in place of the ReSTful Web Services.

Using ReSTful Web Service During Batch Window

The services should not be used during the restricted batch window.

Common Characteristics of Retail Application ReSTful Web Services

This section covers the following:

- Deployment
- Security
- Standard Request and Response Headers
- Standard Error Response
- URL Path
- Web Service APIs Process Flow

Deployment

A Retail application packages its ReST services as part of the application's Enterprise Archive (EAR) file. Specifically, those services are packaged as a Web Archive (WAR) within the EAR. Installation of the ReST Web Services is therefore done by default.

Security

Services are secured using J2EE-based security model.

- Realm-based User Authentication - This verifies users through an underlying realm. The username and password is passed using Http Basic authentication.
- Role-based Authorization - This assigns users to roles, which in turn are granted or restricted access to resources/services. The authorization of ReSTful Web Services is static and cannot be reassigned to other rules post installation. The following roles are associated with ReSTful Web Services and should be added to the Enterprise LDAP:

ROLE-NAME	PRINCIPAL-NAME
BUYER	BUYER_JOB
ALLOCATOR	ALLOCATOR_JOB
ALLOCATION_ MANAGER	ALLOCATION_ MANAGER_JOB

  All enterprise roles defined above are mapped in web.xml and weblogic.xml of the ReST Service Web application.

- The communication between the server and client should be encrypted using one way SSL to protect sensitive information (e.g. user names and passwords). It is highly recommended that these ReST services are configured to use SSL connections in production environments.

Standard Request and Response Headers

Retail application ReSTful Web Services have the following standard HTTP headers:

```
Accept: application/xml or application/JSON
Accept-Version: 14.1 (service version number)
Accept-Language: en-US,en;q=0.8
Content-Type: application/xml or application/JSON
```

Depending on the type of the operation or HTTP method, the corresponding response header is updated in the Http response with the following codes:

```
· GET/READ : 200
· PUT/CREATE : 201 created
· POST/UPDATE : 204
· DELETE : 204
```

Standard Error Response

An example response payload in case of service error is depicted below:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<messagesRDOes>
<messagesRDO>
<message>REST Service Version Mismatch</message>
<messageType>ERROR</messageType>
<status>BAD_REQUEST</status>
</messagesRDO>
</messagesRDOes>
```

- message: The error message - translated.
- messageType: Value of 'ERROR' is returned.
- status: For a bad request or error, the status is BAD_REQUEST.
- The http error code for an error response is 400

URL Path

To access the Allocation ReSTful Web Services javadoc:

`http://host:port/Allocation14`

To access the Allocation ReSTful Web Services' WADL file:

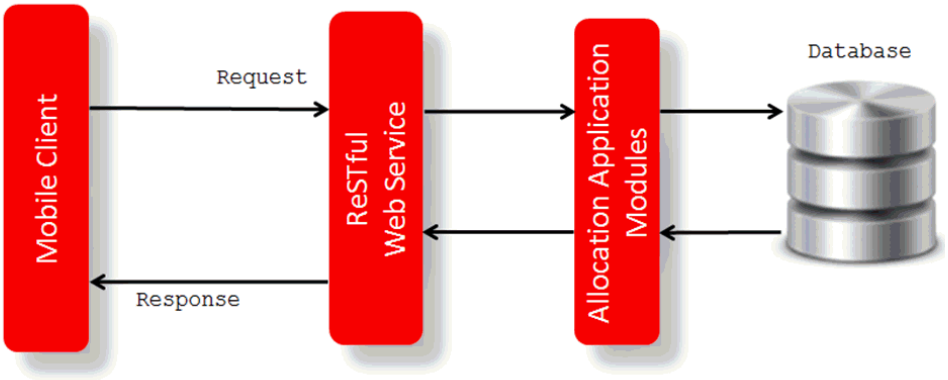
`http://host:port/Allocation14/services/private/application.wadl`

To access the Allocation ReSTful Web Services:

`http://host:port/Allocation14/services/private/<service>`

Web Service APIs Process Flow

Figure 13–2 Web Service API Process Flow



List of ReSTful Web Services

This section covers the following:

- Allocations
- MerchHierarchies

Allocations

This section covers the following:

- Approve
- Load Recent Allocations by User
- Load by Query
- Load by Allocation ID
- Load Item Location Information
- Lookup Allocation Status
- Lookup Process Status
- Reserve
- Submit
- Withdraw

Approve

Business Overview

REST endpoint for approving an allocation.

Service Type

Post

ReST URL
/Allocations/approve

Parameter(s)
AllocationRDO - {"id": Value}

Returns
No content response.

Allowed Job Roles
ALLOCATOR, ALLOCATION_MANAGER

Load Recent Allocations by User

Business Overview
REST endpoint for loading recent allocations of logged in user.

Service Type
Get

ReST URL
/Allocations/recent/{records}

Parameter(s)
records - number of records to return

Returns
A collection of AllocationRDO

Allowed Job Roles
ALLOCATOR, ALLOCATION_MANAGER, BUYER

Load by Query

Business Overview
REST endpoint for loading allocations that satisfy the supplied criteria.

Service Type
Get

ReST URL
/Allocations?dept={dept}&class={class}&subclass={subclass}&id={id}&status={status}
&allocStatus={allocStatus}&createdDate={createdDate}&createdBy={createdBy}

Parameter(s)
- dept - department of the items within an allocation
- class - class of the items within an allocation
- subclass - subclass of the items within an allocation

- ID - comma separated allocation IDs
- status - the process status of an allocation
- allocStatus - the status of an allocation
- createdDate - created date of an allocation in yyyy-MM-dd format
- createdBy - userId information of the user who created allocation

Returns
A collection of AllocationRDO

Allowed Job Roles
ALLOCATOR, ALLOCATION_MANAGER, BUYER

Load by Allocation ID

Business Overview
REST endpoint for loading allocation header based on allocation id.

Service Type
Get

ReST URL
/Allocations/{id}

Parameter(s)
id - allocation id

Returns
A collection of AllocationRDO

Allowed Job Roles
ALLOCATOR, ALLOCATION_MANAGER, BUYER

Load Item Location Information

Business Overview
REST endpoint for loading item location information based on supplied information.

Service Type
Get

ReST URL
/Allocations/{id}/itemloc?facetSessionId={facetSessionId}&allocType={allocType}&itemId={itemId}&whId={whId}&docType={docType}&docId={docId}&diffDescription={diffDescription}&diffTypeId={diffTypeId}

Parameter(s)

- ID - allocation ID

- facetSessionId - facet session ID associated with the allocation. This ID is generated from a loaded allocation.
- allocType - type of allocation. Acceptable values: SA, FA, or FPG.
- itemId - item ID
- whId - warehouse ID associated with the item
- docType - source type where the item is source from. Acceptable Values (PO, ASN, OH, WHIF, BOL,TSF)
- docId - source order number
- diffDescription - diff description retrieved from allocation details service
- diffTypeId - diff type ID retrieved from allocation details service

Returns
A collection of ItemLocRDO

Allowed Job Roles
ALLOCATOR, ALLOCATION_MANAGER, BUYER

Lookup Allocation Status

Business Overview
REST endpoint for looking up allocation statuses.

Service Type
Get

ReST URL
/Allocations/allocStatus

Parameter(s)
None

Returns
A collection of LookUpRDO

Allowed Job Roles
ALLOCATOR, ALLOCATION_MANAGER, BUYER

Lookup Process Status

Business Overview
REST endpoint for looking up process statuses.

Service Type
Get

ReST URL
/Allocations/status

Parameter(s)

None

Returns

A collection of LookUpRDO

Allowed Job Roles

ALLOCATOR, ALLOCATION_MANAGER, BUYER

Reserve

Business Overview

REST endpoint for reserving an allocation.

Service Type

Post

ReST URL

/Allocations/reserve

Parameter(s)

AllocationRDO - {"id": Value}

Returns

No content response.

Allowed Job Roles

ALLOCATOR, ALLOCATION_MANAGER

Submit

Business Overview

REST endpoint for submitting an allocation.

Service Type

Post

ReST URL

/Allocations/submit

Parameter(s)

AllocationRDO - {"id": Value}

Returns

No content response

Allowed Job Roles

ALLOCATOR, ALLOCATION_MANAGER

Withdraw

Business Overview

REST endpoint for withdrawing an allocation.

Service Type

Post

ReST URL

/Allocations/withdraw

Parameter(s)

AllocationRDO - {"id": Value}

Returns

No content response.

Allowed Job Roles

ALLOCATOR, ALLOCATION_MANAGER

MerchHierarchies

This section covers the following:

- Lookup Classes by Department
- Lookup Class
- Lookup All Deparments
- Lookup Department
- Lookup Subclasses by Department and Class
- Lookup Subclass

Lookup Classes by Department

Business Overview

REST endpoint for retrieving all classes under a given department.

Service Type

Get

ReST URL

/MerchHierarchies/dept/{dept}/class

Parameter(s)

dept - department id

Returns

A collection of MerchHierarchyRDO

Lookup Class

Business Overview

REST endpoint for retrieving a specific class.

Service Type

Get

ReST URL

/MerchHierarchies/dept/{dept}/class/{itemclass}

Parameter(s)

- dept - department id
- itemclass - item class id or item class name

Returns

A collection of MerchHierarchyRDO

Lookup All Deparments

Business Overview

REST endpoint for retrieving all departments.

Service Type

Get

ReST URL

/MerchHierarchies/dept

Parameter(s)

None

Returns

A collection of MerchHierarchyRDO

Lookup Department

Business Overview

REST endpoint for retrieving a specific department.

Service Type

Get

ReST URL

/MerchHierarchies/dept/{dept}

Parameter(s)

dept - department id or department name

Returns
A collection of MerchHierarchyRDO

Lookup Subclasses by Department and Class

Business Overview
REST endpoint for retrieving all subclasses under a given department and class.

Service Type
Get

ReST URL
/MerchHierarchies/dept/{dept}/class/{itemclass}/subclass

Parameter(s)
- dept - department id
- itemclass - item class id

Returns
A collection of MerchHierarchyRDO

Lookup Subclass

Business Overview
REST endpoint for retrieving specific subclass.

Service Type
Get

ReST URL
/MerchHierarchies/dept/{dept}/class/{itemclass}/subclass/{subclass}

Parameter(s)
- dept - department id
- itemclass - item class id
- subclass - subclass id or subclass name

Returns
A collection of MerchHierarchyRDO

Platform ReSTful Web Services

This section lists Web Services provided by the Retail Fusion Platform.

Notification ReSTful Web Services

Table 13–1 Notification

Description	URL	HTTP Method
Creating Notifications	/Notifications/create	POST
Updating Notifications	/Notifications/update	PUT
Deleting Notifications	/Notifications/delete/{id}	DELETE
Fetch Notifications	/Notifications/fetch?appCode={appCode}	GET
Get number of Unread Notifications	/Notifications/fetch/unreadCount?appCode={appCode}	GET
Search Notifications	/AccessRoles	GET

14

Implementing Functional Security

This chapter discusses the Allocation functional security and the components used to implement it. Allocation Functional Security is based on OPSS. For more details on OPSS, refer to the *Oracle Fusion Middleware Application Security Guide.*

This chapter includes the following sections:

- Introduction to Retail Roles
- Retail Role Hierarchy
- Default Security Reference Implementation
- Extending the Default Security Reference Implementation

Introduction to Retail Roles

Users are not assigned to permissions directly; rather access is assigned to roles. Roles group particular permissions required to accomplish a task; instead of assigning individual permissions, roles match users with the permissions required to complete their particular task.

There are two main types of roles, enterprise and application.

The Identity Store contains enterprise roles that are available across applications. These are created as groups in LDAP, making them available across applications.

Application roles are stored in the application-specific policy store. These roles and role mappings are described in the jazn-data.xml file under the policy stripe 'ALC_PORTAL'.

Applicable Oracle Retail applications security provides four types of roles: abstract, job, duty, and privilege.

Applicable Oracle Retail applications will record job, abstract roles as enterprise roles and duty, privilege roles as application roles.

Security Policy Stripe

Application roles are stored in the application-specific policy store. These roles and role mappings are described in the jazn-data.xml file under the policy stripe 'ALC_CORE' is the policy stripe.

Abstract Roles

Abstract roles are associated with a user, irrespective of their job or job function. These are also roles that are not associated with a job or duty. These roles are normally assigned by the system (based on user attributes), but can be provisioned to a user on request.

Naming Convention: All the Retail Abstract role names end with' _ABSTRACT'

Example: APPLICATION_ADMIN_ABSTRACT

Job Roles

Job roles are associated with the job of an employee. An employee with this job can have many job functions or job duties.

> **Note:** These roles are called Job roles as the role names closely map to the jobs commonly found in most organizations.

Naming Convention: All the Retail Job role names end with' _JOB'

Example: ALLOCATOR_JOB.

Duty Roles

Job duties are tasks one must do on a job. A person is hired into a job role. These are the responsibilities one has for a job.

Duty roles are roles that are associated with a specific duty or a logical grouping of tasks. Generally, the list of duties for a job is a good indicator of what duty roles should be defined.

Duty roles should:

- Read as a job description at a job posting site
- Duties that we create should be self-contained and pluggable into any existing or new job or abstract role

Naming Convention: All the Retail duty role names end with' _DUTY'

Example: ALC_ALLOC_POLICY_MAINTENANCE_MANAGEMENT_DUTY

Privilege Roles

Privilege is the logical collection of permissions. A privilege can be associated with any number of UI components. Privileges are expressed as application roles.

Naming Convention: All the Retail Privilege role names end with' _PRIV'

Ex: ALC_ALLOC_SEARCH_PRIV

Privilege roles carry security grants.

Example:

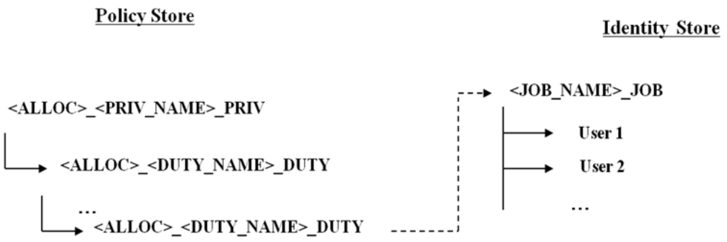
```
<grant>
   <grantee>
      <principals>
            <principal>
<class>oracle.security.jps.service.policystore.
ApplicationRole</class>
               <name>ALC_ALLOC_SEARCH_PRIV</name>
            </principal>
       </principals>
     </grantee>
```

```
  <permissions>
   <permission>
 <class>oracle.adf.controller.security.TaskFlowPermission</class>
 <name>/oracle/retail/apps/alc/allocsummary/publicUi
/flow/AllocationSummaryFlow.xml#AllocationSummaryFlow</name>
    <actions>view</actions>
  </permission>
 </permissions>
</grant>
```

Retail Role Hierarchy

Retail role hierarchies are structured to reflect the retail business process model.

Figure 14–1 Retail Role Hierarchy



Job roles inherit duty roles. For example, the Allocator Job role inherits the ALC_ALLOC_SYSTEM_OPTIONS_INQUIRY_DUTY roles.

```
<app-role>
  <name>ALC_ALLOC_SYSTEM_OPTIONS_INQUIRY_DUTY</name>
  <class>oracle.security.jps.service.policystore.ApplicationRole</class>
  <members>
   <member>
     <class>oracle.security.jps.internal.core.principals.
                JpsXmlEnterpriseRoleImpl</class>
     <name>ALLOCATOR_JOB</name>
  </member>
 </members>
</app-role>
```

Duty roles inherit Privilege roles. Duty roles can inherit one or more other Duty roles.

Example: ALC_ALLOC_SIZE_PROFILE_MANAGEMENT_DUTY inherits ALC_ALLOC_SIZE_PROFILE_INQUIRY_DUTY role.

```
<app-role>
  <name>ALC_ALLOC_SIZE_PROFILE_INQUIRY_DUTY</name>
  <class>oracle.security.jps.service.policystore.ApplicationRole</class>
  <members>
    <member>
     <class>oracle.security.jps.internal.core.principals.
                              JpsXmlEnterpriseRoleImpl</class>
     <name>BUYER_JOB</name>
    </member>
    <member>
     <class>oracle.security.jps.service.policystore.ApplicationRole</class>
     <name>ALC_ALLOC_SIZE_PROFILE_MANAGEMENT_DUTY</name>
    </member>
  </members>
</app-role>
```

Example: ALC_ALLOC_SIZE_PROFILE_INQUIRY_DUTY role inherits the ALC_ALLOC_SIZE_PROFILE_VIEW_PRIV role

```
<app-role>
 <name>ALC_ALLOC_SIZE_PROFILE_VIEW_PRIV</name>
 <class>oracle.security.jps.service.policystore.ApplicationRole</class>
 <members>
  <member>
   <class>oracle.security.jps.service.policystore.ApplicationRole</class>
   <name>ALC_ALLOC_SIZE_PROFILE_INQUIRY_DUTY</name>
  </member>
 </members>
</app-role>
```

Default Security Reference Implementation

Oracle Retail Allocation ships with a default security reference implementation. The source of truth for default reference implementation is jazn-data.xml.

Privileges

Table 14–1 Allocation Privileges

Name	Description
Search Allocations Priv	A privilege for searching for allocations.
Maintain Allocation Priv	A privilege for creating and editing an allocation.
Delete Allocation Priv	A privilege for deleting an allocation.
View Allocation Priv	A privilege for viewing an allocation.
Submit Allocation Priv	A privilege for submitting a allocation for approval.
Review Allocation Priv	A privilege for approving or reviewing an allocation.
Search Allocation Location Groups Priv	A privilege for searching for allocation location groups.
Maintain Allocation Location Group Priv	A privilege for creating and editing an allocation location group
Delete Allocation Location Group Priv	A privilege for deleting an allocation location group.
View Allocation Location Group Priv	A privilege for viewing an allocation location group.
Search Allocation Policy Templates Priv	A privilege for searching for allocation policy templates.
Maintain Allocation Policy Template Priv	A privilege for creating and editing a policy template.
Delete Allocation Policy Template Priv	A privilege for deleting a policy template.
View Allocation Policy Template Priv	A privilege for viewing a policy template.
Search Size Profiles Priv	A privilege for searching for size profiles.
Maintain Size Profile Priv	A privilege for creating and editing a size profile.

Table 14–1 (Cont.) Allocation Privileges

Name	Description
Delete Size Profile Priv	A privilege for deleting a size profile.
View Size Profile Priv	A privilege for viewing a size profile.
Maintain System Options System Properties Priv	A privilege for editing the system properties for system options.
Maintain System Options User Group Properties Priv	A privilege for editing the user group properties for system options.
View System Options Priv	A privilege for viewing system options.

Duties

Table 14–2 Allocation Duties

Duty	Description	List of Privileges
Allocation Management Duty	A duty for managing allocations. This duty is an extension of the Allocation Inquiry Duty.	■ All privileges found in the Allocation Inquiry Duty ■ Maintain Allocation Privilege ■ Delete Allocation Privilege
Allocation Inquiry Duty	A duty for viewing allocations.	■ View Allocation Privilege ■ Search Allocations Privilege
Allocation Submit Duty	A duty for submitting allocation for approval.	Submit Allocation Privilege
Allocation Review Duty	A duty for approving or rejecting an allocation.	Review Allocation Privilege
Allocation Location Groups Management Duty	A duty for managing allocation location groups. This duty is an extension of the Allocation Location Groups Inquiry Duty and Allocation Location Group Search Duty.	■ All privileges found in the Allocation Location Groups Inquiry Duty and the Allocation Location Groups Search Duty ■ Maintain Allocation Location Groups Privilege ■ Delete Allocation Location Groups Privilege
Allocation Location Groups Inquiry Duty	A duty for viewing allocation location groups.	View Allocation Location Groups Privilege
Allocation Location Groups Search Duty	A duty for searching allocation location groups.	Search Allocation Location Groups Privilege
Allocation Policy Template Management Duty	A duty for managing allocation policy template. This duty is an extension of the Allocation Policy Template Inquiry Duty and Allocation Policy Template Search Duty.	■ All privileges found in the Allocation Policy Template Inquiry Duty and the Allocation Policy Template Search Privilege ■ Maintain Allocation Policy Template Privilege ■ Delete Allocation Policy Template Privilege
Allocation Policy Template Inquiry Duty	A duty for viewing allocation Policy Template.	View Allocation Policy Template Privilege
Allocation Policy Template Search Duty	A duty for search allocation Policy Template.	Search Allocation Policy Template Privilege

Table 14–2 (Cont.) Allocation Duties

Duty	Description	List of Privileges
Size Profile Management Duty	A duty for managing size profile. This duty is an extension of the Size Profile Inquiry Duty.	■ All privileges found in the Size Profile Inquiry Duty ■ Maintain Size Profile Privilege ■ Delete Size Profile Privilege
Size Profile Inquiry Duty	A duty for viewing allocation Size Profile.	■ View Size Profile Privilege ■ Search Size Profiles Privilege
System Options System Properties Management Duty	A duty for managing the system properties in system options. This duty is an extension of the System Options Inquiry Duty.	■ All privileges found in the System Options Inquiry Duty ■ Maintain System Options System Properties Privilege
System Options User Group Properties Management Duty	A duty for managing user group properties system options. This duty is an extension of the System Options Inquiry Duty.	■ All privileges found in the System Options Inquiry Duty ■ Maintain System Options User Group Properties Privilege
System Options Inquiry Duty	A duty for inquiring on profile. This duty is an extension of the Size Profile Inquiry Duty.	■ All privileges found in the System Options Inquiry Duty ■ Maintain System Options Privilege
Administrator Duty	A duty for managing WebCenter Portal Administrative duties	N/A
Edit Current Page Duty	This duty provides personalization capability on pages. by default, this feature is disabled for all users, But a system administrator can grant this role to any of the enterprise roles during run time based on need.	N/A

Role Mapping

Table 14–3 Allocation Role Mappings

Role	Duty	Privileges
Administrator	Allocation Management Duty Allocation Submit Duty Allocation Review Duty Allocation Location Groups Management Duty Allocation Policy Template Management Duty Size Profile Management Duty System Options System Properties Management Duty System Options User Group Properties Management Duty	Search Allocations Privilege Maintain Allocation Privilege Delete Allocation Privilege Submit Allocation Privilege Review Allocation Privilege View Allocation Privilege Search Allocation Privilege View Allocation Location Groups Privilege Search Allocation Location Groups Privilege View Allocation Policy Template Privilege Search Allocation Policy Templates Privilege View Size Profile Privilege Search Size Profile Privilege View System Options Privilege Maintain System Options User Group Properties Privilege Maintain System Options System Properties Privilege
Allocation Manager	Allocation Management Duty Allocation Submit Duty Allocation Review Duty Allocation Location Groups Management Duty Allocation Policy Template Management Duty Size Profile Management Duty System Options User Group Properties Management Duty	Search Allocations Privilege Maintain Allocation Privilege Delete Allocation Privilege Submit Allocation Privilege Review Allocation Privilege View Allocation Privilege Search Allocation Privilege View Allocation Location Groups Privilege Search Allocation Location Groups Privilege View Allocation Policy Template Privilege Search Allocation Policy Templates Privilege View Size Profile Privilege Search Size Profile Privilege View System Options Privilege Maintain System Options User Group Properties Privilege
Allocator	Allocation Management Duty Allocation Submit Duty Allocation Review Duty Allocation Location Groups Inquiry Duty Allocation Location Groups Search Duty Allocation Policy Template Inquiry Duty Allocation Policy Template Search Duty Size Profile Management Duty System Options Inquiry Duty	Search Allocations Privilege Maintain Allocation Privilege Delete Allocation Privilege Submit Allocation Privilege Review Allocation Privilege View Allocation Privilege Search Allocation Privilege View Allocation Location Groups Privilege Search Allocation Location Groups Privilege View Allocation Policy Template Privilege Search Allocation Policy Templates Privilege View Size Profile Privilege Search Size Profile Privilege View System Options Privilege
Buyer	Allocation Inquiry Duty Allocation Policy Template Inquiry Duty Allocation Location Groups Inquiry Duty	View Allocation Privilege Search Allocation Privilege View Allocation Location Groups Privilege View Allocation Policy Template Privilege
Application Administrator	Administrator Duty	N/A

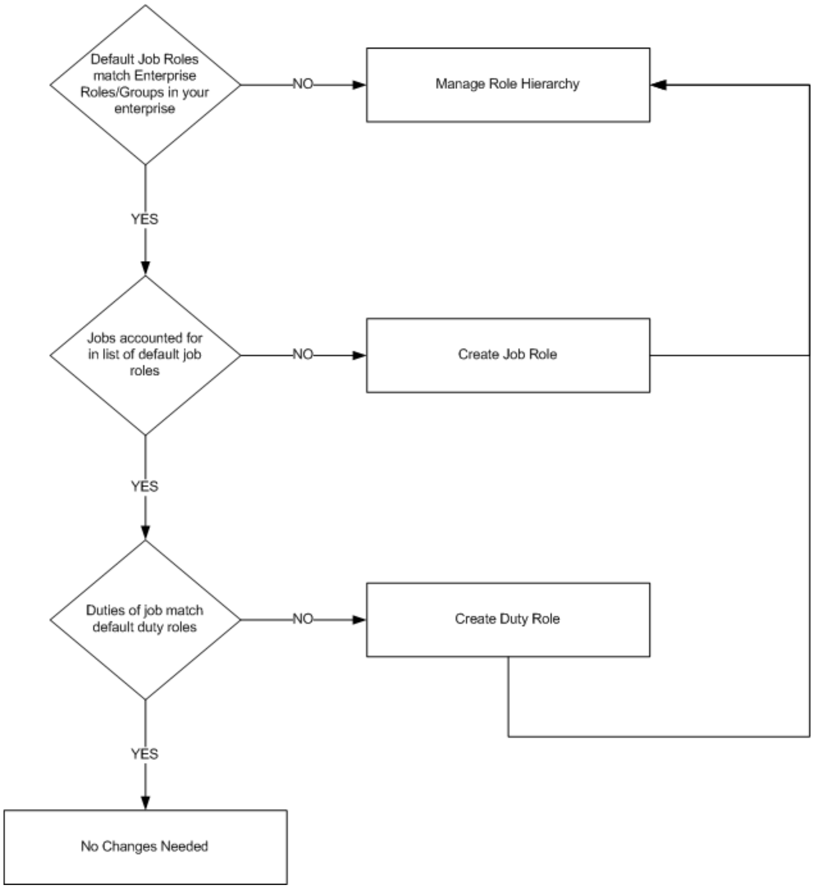
Extending the Default Security Reference Implementation

Note: Make sure that the policy store is loaded with the default security configuration. For more information, see the Post Installation steps in the *Oracle Retail Allocation Installation Guide*.

The common decisions made to match your enterprise to the default security reference implementation include the following:

- Do the default job roles match the equivalent job roles in your enterprise?
- Do the jobs in your enterprise exist in the security reference implementation?
- Do the duties performed by the jobs in your enterprise match the duties in the security reference implementation?

Figure 14–2 Role Hierarchy Decision Flow


Default Job Roles match Enterprise Roles/Groups in your enterprise
NO
Manage Role Hierarchy
YES
Jobs accounted for in list of default job roles
NO
Create Job Role
YES
Duties of job match default duty roles
NO
Create Duty Role
YES
No Changes Needed

Important: It is important when constructing a role hierarchy that circular dependencies are not introduced. Best practice is to leave the default security configuration in place and first incorporate your customized application roles in a test environment.

Access Oracle Enterprise Manager Fusion Middleware Control

Oracle Enterprise Manager Fusion Middleware Control is used to create and manage roles and role hierarchies. The following procedures require you to access Oracle Enterprise Manager Fusion Middleware Control:

- Adding or Removing Members from an Application Role
- Creating a New Application Role
- Creating an Application Role from an Existing Role

> **Note:** Launch Fusion Middleware Control by entering its URL into a Web browser. The URL includes the name of the host and the administration port number assigned during the installation. This URL takes the following form: http://hostname:port_number/em. The default port is 7001. For more information about using Fusion Middleware Control, see *Oracle Fusion Middleware Administrator's Guide*.

Displaying the Security Menu

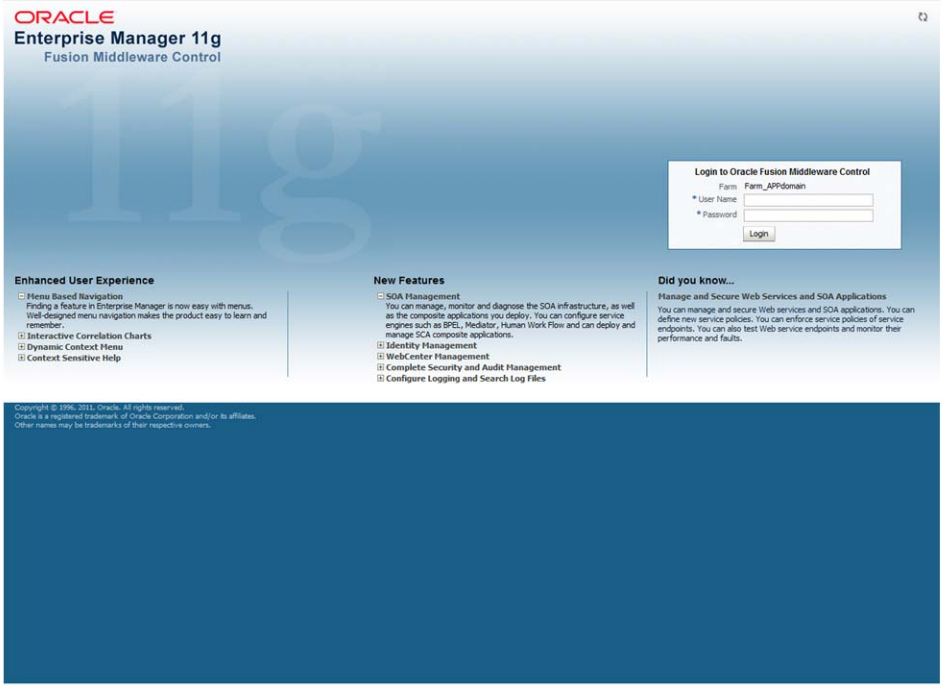
Use the following procedure to display the security menu in Fusion Middleware Control.

1. Log into Oracle Enterprise Manager Fusion Middleware Control by entering the URL in a Web browser.

   For example, http://hostname:7001/em.

   The Fusion Middleware Control login page displays.

Figure 14–3 Logging in to Fusion Middleware Control



Login to Oracle Fusion Middleware Control

Farm Farm_APPdomain

* User Name

* Password

Enhanced User Experience

⊟ Menu Based Navigation
Finding a feature in Enterprise Manager is now easy with menus. Well-designed menu navigation makes the product easy to learn and remember.

⊞ Interactive Correlation Charts

⊞ Dynamic Context Menu

⊞ Context Sensitive Help

New Features

⊟ SOA Management
You can manage, monitor and diagnose the SOA infrastructure, as well as the composite applications you deploy. You can configure service engines such as BPEL, Mediator, Human Work Flow and can deploy and manage SCA composite applications.

⊞ Identity Management

⊞ WebCenter Management

⊞ Complete Security and Audit Management

⊞ Configure Logging and Search Log Files

Did you know...

Manage and Secure Web Services and SOA Applications

You can manage and secure Web services and SOA applications. You can define new service policies. You can enforce service policies of service endpoints. You can also test Web service endpoints and monitor their performance and faults.

Copyright © 1996, 2011, Oracle. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its affiliates.
Other names may be trademarks of their respective owners.

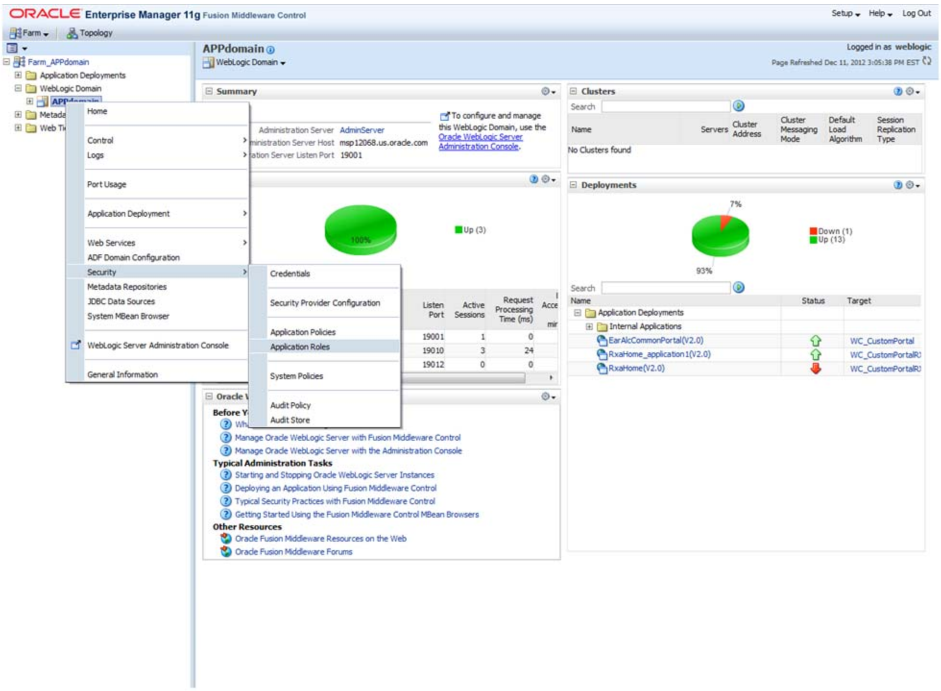
2. Enter the Retail Fusion application's administrative user name and password and click **Login**.

   The password is the one supplied during the installation of the Retail Fusion application. If these values have been changed, then use the current administrative user name and password combination.

3. From the target navigation pane, open the WebLogic Domain to display the application domain (for example: APPdomain). Display the Security menu by using one of the following methods:

   - Right-click the application domain and hover over Security in the popup menu to display a submenu.

Figure 14–4 Displaying the Security Menu via Right-Clicking


ORACLE Enterprise Manager 11g Fusion Middleware Control
Setup
Help
Log Out
Farm
Topology
Farm_APPdomain
Application Deployments
WebLogic Domain
APPdomain
Metada
Web Ti
Home
Control
Logs
Port Usage
Application Deployment
Web Services
ADF Domain Configuration
Security
Metadata Repositories
JDBC Data Sources
System MBean Browser
WebLogic Server Administration Console
General Information
Credentials
Security Provider Configuration
Application Policies
Application Roles
System Policies
Audit Policy
Audit Store
APPdomain
WebLogic Domain
Summary
Clusters
Deployments
100%
Up (3)
7%
93%
Down (1)
Up (13)
Oracle W
Before Y
Wh
Manage Oracle WebLogic Server with Fusion Middleware Control
Manage Oracle WebLogic Server with the Administration Console
Typical Administration Tasks
Starting and Stopping Oracle WebLogic Server Instances
Deploying an Application Using Fusion Middleware Control
Typical Security Practices with Fusion Middleware Control
Getting Started Using the Fusion Middleware Control MBean Browsers
Other Resources
Oracle Fusion Middleware Resources on the Web
Oracle Fusion Middleware Forums

Logged in as weblogic

Page Refreshed Dec 11, 2012 3:05:38 PM EST

Administration Server AdminServer
ministration Server Host msp12068.us.oracle.com
ation Server Listen Port 19001

To configure and manage this WebLogic Domain, use the Oracle WebLogic Server Administration Console.

Listen Port	Active Sessions	Request Processing Time (ms)	Acce mir
19001	1	0	
19010	3	24	
19012	0	0	

Search

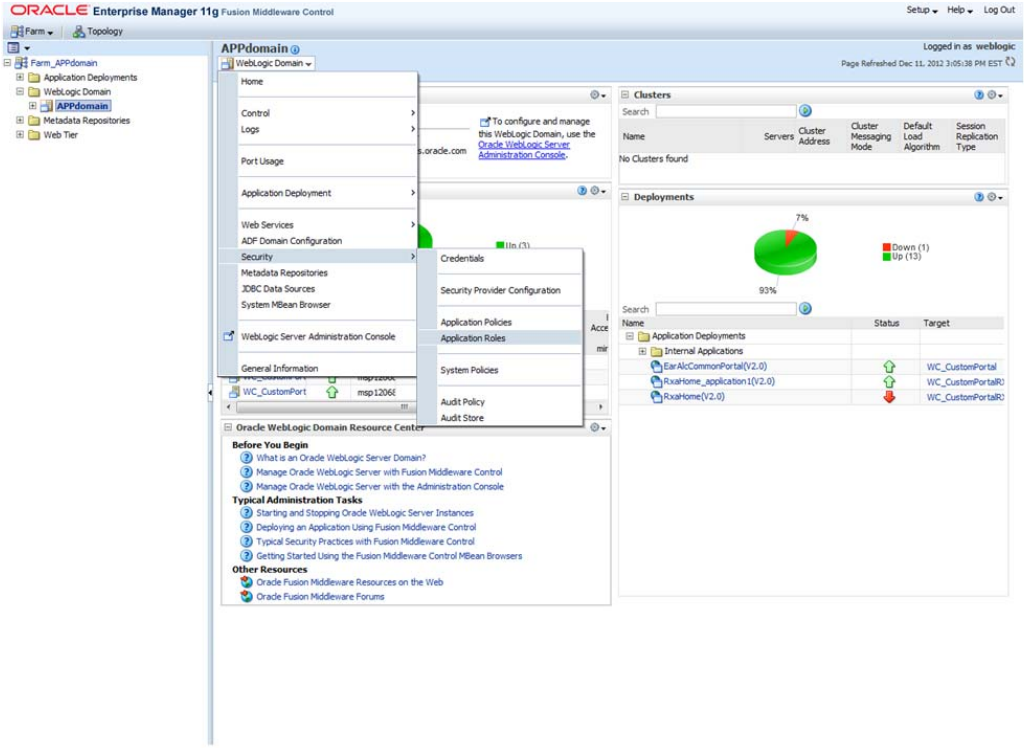
Name	Servers	Cluster Address	Cluster Messaging Mode	Default Load Algorithm	Session Replication Type
No Clusters found					

Search

Name	Status	Target
Application Deployments		
Internal Applications		
EarAlcCommonPortal(V2.0)	⇧	WC_CustomPortal
RxaHome_application1(V2.0)	⇧	WC_CustomPortalR
RxaHome(V2.0)	⬇	WC_CustomPortalR

   - From the content pane, select the application domain in the tree to open the domain's home page. Open the WebLogic Domain menu located below the domain's name and hover over Security to open the Security submenu.

Figure 14–5 Displaying the Security Menu via the WebLogic Domain Menu


ORACLE Enterprise Manager 11g Fusion Middleware Control
Setup
Help
Log Out
Farm
Topology
Farm_APPdomain
Application Deployments
WebLogic Domain
APPdomain
Metadata Repositories
Web Tier
APPdomain
WebLogic Domain
Logged in as weblogic
Page Refreshed Dec 11, 2012 3:05:38 PM EST
Home
Control
Logs
Port Usage
Application Deployment
Web Services
ADF Domain Configuration
Security
Metadata Repositories
JDBC Data Sources
System MBean Browser
WebLogic Server Administration Console
General Information
Credentials
Security Provider Configuration
Application Policies
Application Roles
System Policies
Audit Policy
Audit Store
To configure and manage this WebLogic Domain, use the Oracle WebLogic Server Administration Console.
Clusters
Search
Name
Servers
Cluster Address
Cluster Messaging Mode
Default Load Algorithm
Session Replication Type
No Clusters found
Deployments
7%
93%
Down (1)
Up (13)
Name
Status
Target
Application Deployments
Internal Applications
EarAlcCommonPortal(V2.0)
RxaHome_application1(V2.0)
RxaHome(V2.0)
WC_CustomPortal
WC_CustomPortalR1
WC_CustomPortalR1
WC_CustomPort
msp1206
Oracle WebLogic Domain Resource Center
Before You Begin
What is an Oracle WebLogic Server Domain?
Manage Oracle WebLogic Server with Fusion Middleware Control
Manage Oracle WebLogic Server with the Administration Console
Typical Administration Tasks
Starting and Stopping Oracle WebLogic Server Instances
Deploying an Application Using Fusion Middleware Control
Typical Security Practices with Fusion Middleware Control
Getting Started Using the Fusion Middleware Control MBean Browsers
Other Resources
Oracle Fusion Middleware Resources on the Web
Oracle Fusion Middleware Forums

Managing Role Hierarchy

Members can be added or deleted from an application role using Fusion Middleware Control. Be very careful when changing the permission grants and membership for the default application roles. Changes could result in an unusable system.

Valid members of an application role are groups, or other application roles. The process of becoming a member of an application role is called mapping. That is, being mapped to an application role is to become a member of an application role. Best practice is to map groups instead of individual users to application roles for easier maintenance.

Adding or Removing Members from an Application Role

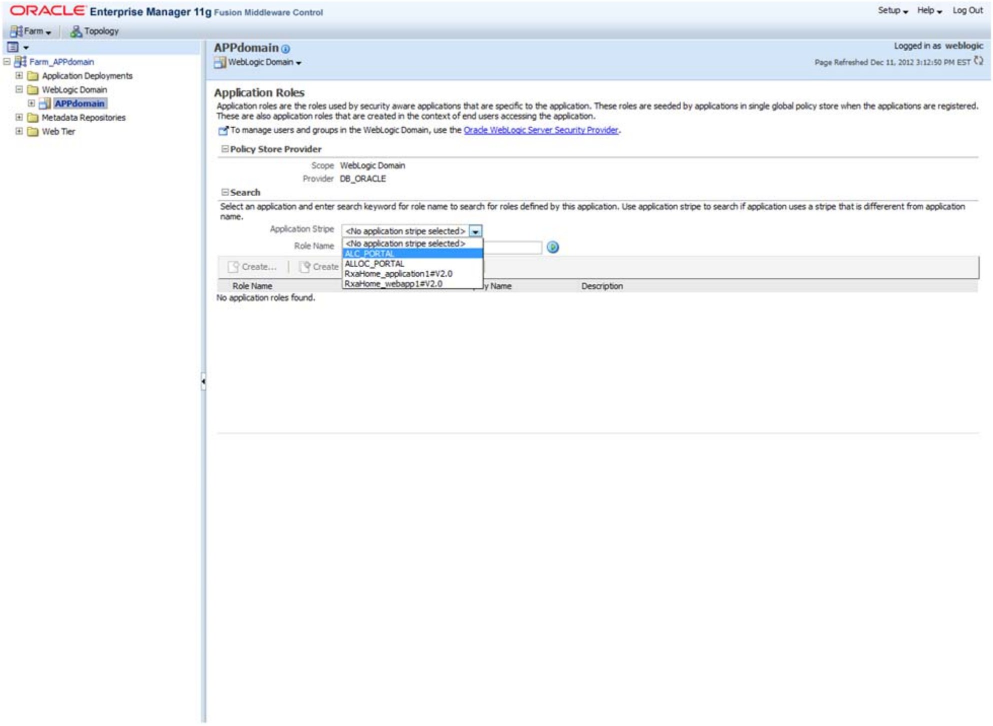
Use the following procedure to add or remove members from an application role.

1. Log into Fusion Middleware Control, navigate to Security, then select Application Roles to display the Application Roles page.

   For information about navigating to the Security menu, see "Access Oracle Enterprise Manager Fusion Middleware Control".

2. Choose Select Application Stripe to Search, then select the policy stripe name (for example: ALC_PORTAL) from the list. Click the search icon next to Role Name.

Figure 14–6 Application Roles Window



Logged in as weblogic

Page Refreshed Dec 11, 2012 3:12:50 PM EST

Application Roles

Application roles are the roles used by security aware applications that are specific to the application. These roles are seeded by applications in single global policy store when the applications are registered. These are also application roles that are created in the context of end users accessing the application.

To manage users and groups in the WebLogic Domain, use the Oracle WebLogic Server Security Provider.

⊟ Policy Store Provider

Scope WebLogic Domain

Provider DB_ORACLE

⊟ Search

Select an application and enter search keyword for role name to search for roles defined by this application. Use application stripe to search if application uses a stripe that is differerent from application name.

Application Stripe <No application stripe selected>

- <No application stripe selected>
- ALC_PORTAL
- ALLOC_PORTAL
- RxaHome_application1#V2.0
- RxaHome_webapp1#V2.0

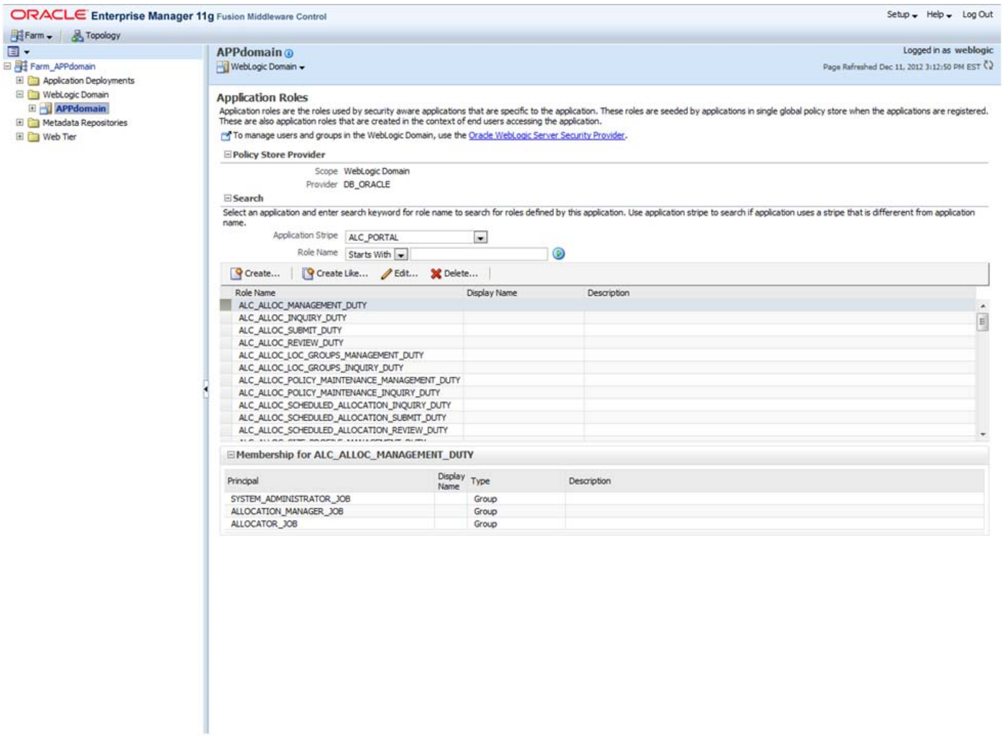
Role Name

Role Name	Display Name	Description

No application roles found.

The Retail Fusion Application's application roles are displayed. As an example, in the following figure the default application roles are shown.

Figure 14–7 Viewing the Default Application Roles



Logged in as weblogic

Page Refreshed Dec 11, 2012 3:12:50 PM EST

Application Roles

Application roles are the roles used by security aware applications that are specific to the application. These roles are seeded by applications in single global policy store when the applications are registered. These are also application roles that are created in the context of end users accessing the application.

To manage users and groups in the WebLogic Domain, use the Oracle WebLogic Server Security Provider.

⊟ Policy Store Provider

Scope WebLogic Domain

Provider DB_ORACLE

⊟ Search

Select an application and enter search keyword for role name to search for roles defined by this application. Use application stripe to search if application uses a stripe that is differerent from application name.

Application Stripe ALC_PORTAL

Role Name Starts With

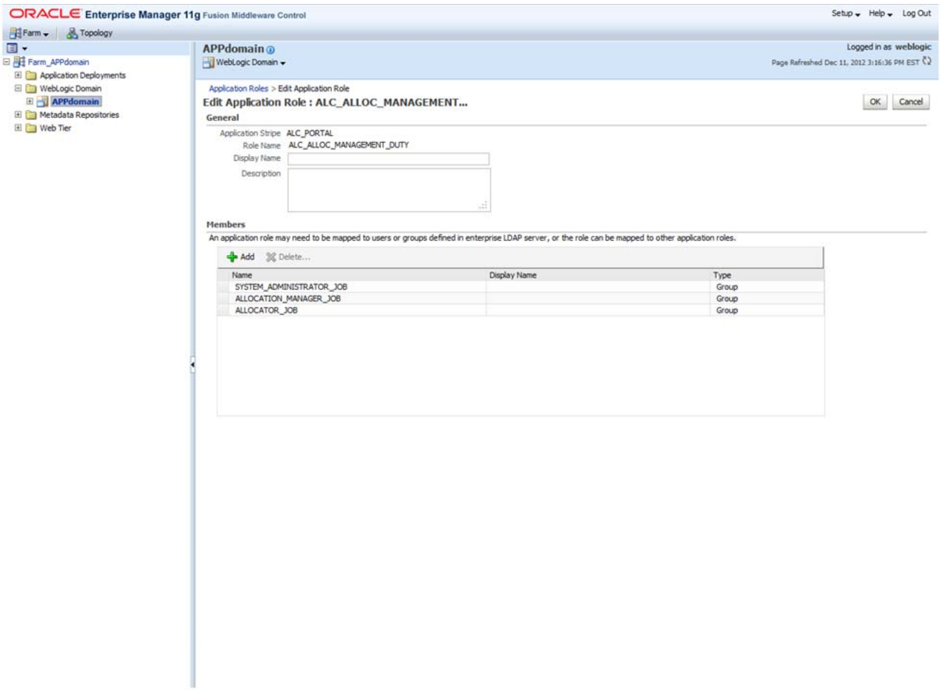
Role Name	Display Name	Description
ALC_ALLOC_MANAGEMENT_DUTY		
ALC_ALLOC_INQUIRY_DUTY		
ALC_ALLOC_SUBMIT_DUTY		
ALC_ALLOC_REVIEW_DUTY		
ALC_ALLOC_LOC_GROUPS_MANAGEMENT_DUTY		
ALC_ALLOC_LOC_GROUPS_INQUIRY_DUTY		
ALC_ALLOC_POLICY_MAINTENANCE_MANAGEMENT_DUTY		
ALC_ALLOC_POLICY_MAINTENANCE_INQUIRY_DUTY		
ALC_ALLOC_SCHEDULED_ALLOCATION_INQUIRY_DUTY		
ALC_ALLOC_SCHEDULED_ALLOCATION_SUBMIT_DUTY		
ALC_ALLOC_SCHEDULED_ALLOCATION_REVIEW_DUTY		

⊟ Membership for ALC_ALLOC_MANAGEMENT_DUTY

Principal	Display Name	Type	Description
SYSTEM_ADMINISTRATOR_JOB		Group	
ALLOCATION_MANAGER_JOB		Group	
ALLOCATOR_JOB		Group	

3. Select the cell next to the application role name and click Edit to display the Edit Application Role page. In the following figure the 'ALC_ALLOC_MANAGEMENT_DUTY' role has been selected.

Figure 14–8 Editing the Application Role



Edit Application Role : ALC_ALLOC_MANAGEMENT...

General

Application Stripe ALC_PORTAL

Role Name ALC_ALLOC_MANAGEMENT_DUTY

Display Name

Description

Members

An application role may need to be mapped to users or groups defined in enterprise LDAP server, or the role can be mapped to other application roles.

Name	Display Name	Type
SYSTEM_ADMINISTRATOR_JOB		Group
ALLOCATION_MANAGER_JOB		Group
ALLOCATOR_JOB		Group

You can add or delete members from the Edit Application Role page. Valid members are application roles and groups.

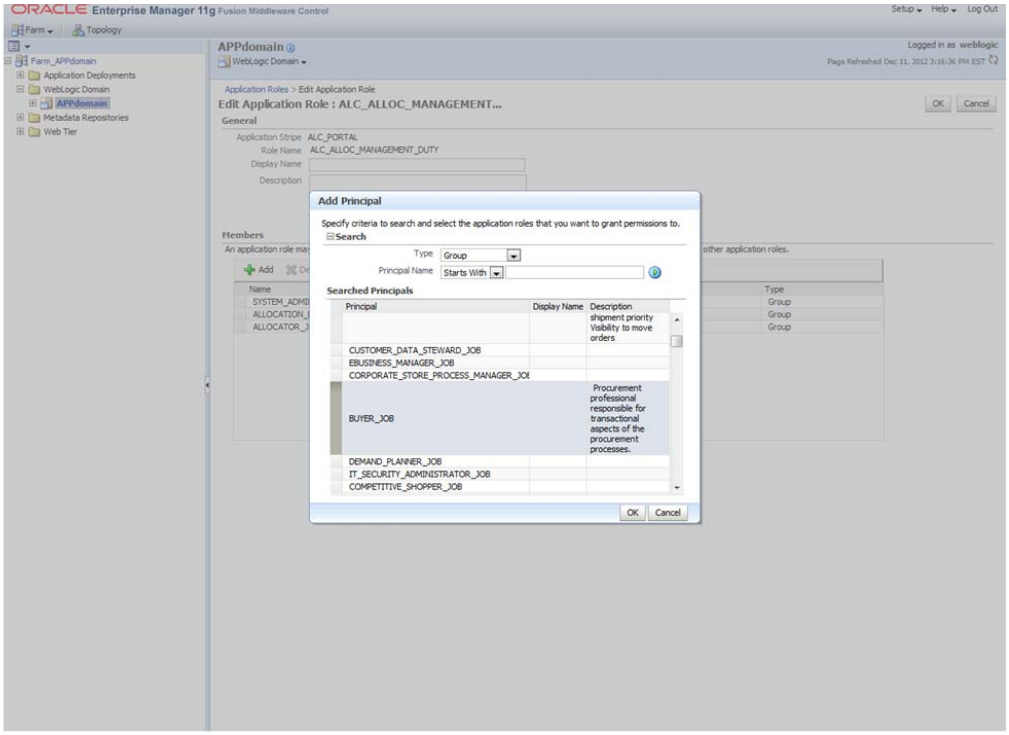
4. Select from the following options:

   - To delete a member, select the member and click **Delete**.
   - To add a member, click the **Add** button that corresponds to the member type being added to open the window. From the window, select from Add Application Role, Add Group, and Add User.

     If adding a member, complete Search and select from the available list and click **OK**.

     For example, the following figure shows the Add Group window after the BUYER_JOB group has been selected.

Figure 14–9 Adding a Group



Edit Application Role : ALC_ALLOC_MANAGEMENT...

General

Application Stripe ALC_PORTAL
Role Name ALC_ALLOC_MANAGEMENT_DUTY
Display Name
Description

Members

An application role ma... other application roles.

Name
SYSTEM_ADMI
ALLOCATION_
ALLOCATOR_J

Add Principal

Specify criteria to search and select the application roles that you want to grant permissions to.

Search

Type: Group
Principal Name: Starts With

Searched Principals

Principal	Display Name	Description
		shipment priority Visibility to move orders
CUSTOMER_DATA_STEWARD_JOB		
EBUSINESS_MANAGER_JOB		
CORPORATE_STORE_PROCESS_MANAGER_JOE		
BUYER_JOB		Procurement professional responsible for transactional aspects of the procurement processes.
DEMAND_PLANNER_JOB		
IT_SECURITY_ADMINISTRATOR_JOB		
COMPETITIVE_SHOPPER_JOB		

The added member displays in the Members column corresponding to the application role modified in the Application Roles page.

Creating Job Roles

There are two methods for creating new Job roles:

- Create New – Refer to the *Oracle® Fusion Middleware Administrator's Guide for Oracle Internet Directory 11g Release 1 (11.1.1)* for creating new Enterprise Roles/Groups
- Replace with Existing – Refer to the Manage Role Hierarchy section to replace the default Job role with existing Enterprise role/group using Fusion Middleware Control.

Creating Duty Roles

There are two methods for creating new duty roles:

- Create New – A new application (duty) role is created. Members can be added at the same time or you can save the new role after naming it and add members later.
- Copy Existing – A new application (duty) role is created by copying an existing application role. The copy contains the same members as the original, and is made a Grantee of the same application policy. You can modify the copy as needed to finish creating the new role.

Creating a New Application Role

Use the following procedure to create a new application role.

1. Log into Fusion Middleware Control, navigate to Security, then select Application Roles to display the Application Roles page.

   For more information, see "Access Oracle Enterprise Manager Fusion Middleware Control".

2. Choose Select Application Stripe to Search, and then click the search icon next to Role Name.

   The Retail Fusion Application's application roles display.

3. Click **Create** to display the Create Application Role page. You can enter all information at once or you can enter a Role Name, save it, and complete the remaining fields later. Complete the fields as follows:

   In the General section:

   - Role Name – Enter the name of the application role.
   - (Optional) Display Name – Enter the display name for the application role.
   - (Optional) Description – Enter a description for the application role.

   In the Members section, select the groups, or application roles to be mapped to the application role, select Add Application Role or Add Group accordingly. To search in the window that displays:

   a. Enter a name in Name field and click the blue button to search.

   b. Select from the results returned in the Available box.

   c. Click OK to return to the Create Application Role page.

   d. Repeat the steps until all members are added to the application role.

4. Click OK to return to the Application Roles page.

   The application role just created displays in the table at the bottom of the page.

Creating an Application Role from an Existing Role

Use the following procedure to copy an existing application role.

1. Log into Fusion Middleware Control, navigate to Security, then select Application Roles to display the Application Roles page.

   For more information, see "Access Oracle Enterprise Manager Fusion Middleware Control".

2. Choose Select Application Stripe to Search, and then click the search icon next to Role Name.

   The Retail Fusion Application's application roles display.

3. Select an application role from the list to enable the action buttons.

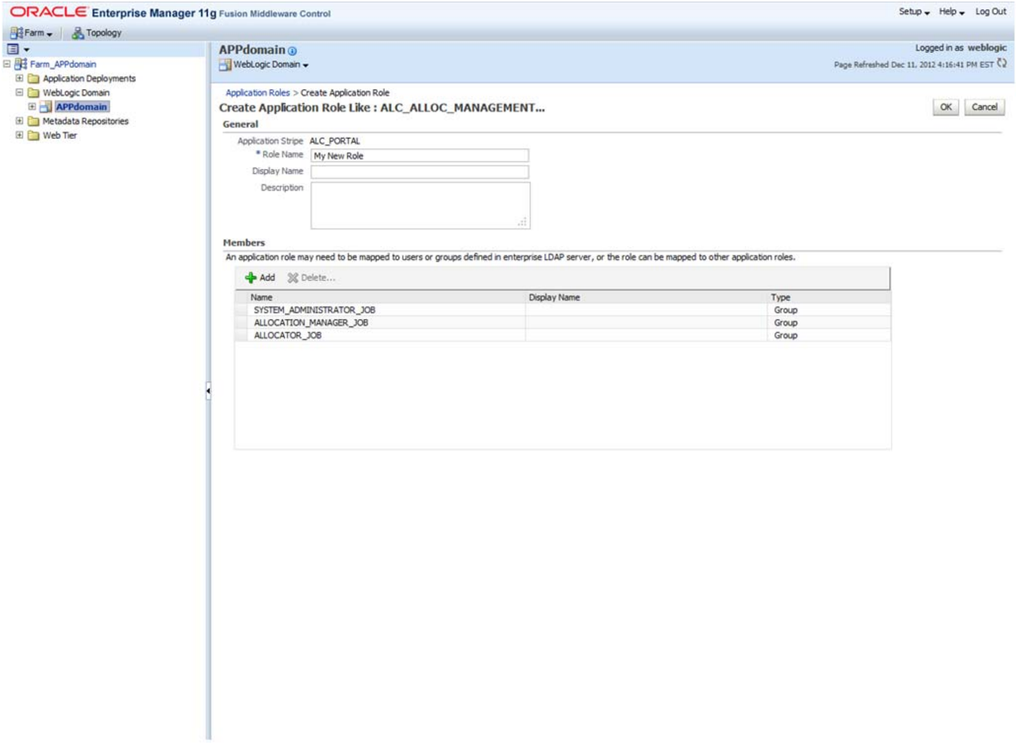
4. Click **Create Like** to display the Create Application Role Like page.

   The Members section is completed with the same application roles, groups that are mapped to the original role.

5. Complete the Role Name, Display Name, and Description fields.

   The following figure shows an application role based upon ALC_ALLOC_MANAGEMENT_DUTY after being named MyNewRole, as an example.

Figure 14–10 Copying an Application Role



Create Application Role Like : ALC_ALLOC_MANAGEMENT...

General

Application Stripe: ALC_PORTAL

* Role Name: My New Role

Display Name:

Description:

Members

An application role may need to be mapped to users or groups defined in enterprise LDAP server, or the role can be mapped to other application roles.

Name	Display Name	Type
SYSTEM_ADMINISTRATOR_JOB		Group
ALLOCATION_MANAGER_JOB		Group
ALLOCATOR_JOB		Group

6. Use Add and Delete to modify the members as appropriate and click **OK**.

   The just-created application role displays in the table at the bottom of the page.

Security in Retail Applications

Retail applications leverage ADF's security framework that is based on the Oracle Platform Security Services.

This section discusses the various assumptions around security for Retail Applications.

Single Sign On (SSO) Setup for Retail Fusion Platform Applications

Retail Fusion Platform provides the following applications as enterprise archive (EAR) files to Retail applications. By default, these applications are installing as part of Retail applications.

1. RetailAppsAdminConsole(RAAC)
2. RetailAppsMobileSecurity
   a. RetailAppsMobileBasicAuth
   b. RetailAppsMobileAccessService
3. RetailAppsRESTServices

In SSO environment, follow the SSO setup procedure for these applications similar to Retail applications.

Displaying External Application Contents in Non-SSO Environments

Retail Applications allow retailers to display content from external applications. These contents are typically business intelligence reports from a third party application that are configured to display within the Retail Application's dashboard.

Some of these contents might be secured requiring users to login before the contents can be accessed and displayed.

In non-SSO environments, when you log out of the Retail application, you may not be logged out of any secured content you have configured access to. Therefore, it is highly recommended that retailers only configure access to external content in a SSO-enabled environments where the application logout manages the logout from any other secured content that was previously accessed.