Oracle® Retail Allocation Cloud Service

Operations Guide

Release 16.0.21

E87299-01

May 2017

ORACLE®

Oracle® Retail Allocation Cloud Service Operations Guide, Release 16.0.21

E87299-01

Copyright © 2017, Oracle and/or its affiliates. All rights reserved.

Primary Author: Nathan Young

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Value-Added Reseller (VAR) Language

Oracle Retail VAR Applications

The following restrictions and provisions only apply to the programs referred to in this section and licensed to you. You acknowledge that the programs may contain third party software (VAR applications) licensed to Oracle. Depending upon your product and its version number, the VAR applications may include:

(i) the **MicroStrategy** Components developed and licensed by MicroStrategy Services Corporation (MicroStrategy) of McLean, Virginia to Oracle and imbedded in the MicroStrategy for Oracle Retail Data Warehouse and MicroStrategy for Oracle Retail Planning & Optimization applications.

(ii) the **Wavelink** component developed and licensed by Wavelink Corporation (Wavelink) of Kirkland, Washington, to Oracle and imbedded in Oracle Retail Mobile Store Inventory Management.

(iii) the software component known as **Access Via™** licensed by Access Via of Seattle, Washington, and imbedded in Oracle Retail Signs and Oracle Retail Labels and Tags.

(iv) the software component known as **Adobe Flex™** licensed by Adobe Systems Incorporated of San Jose, California, and imbedded in Oracle Retail Promotion Planning & Optimization application.

You acknowledge and confirm that Oracle grants you use of only the object code of the VAR Applications. Oracle will not deliver source code to the VAR Applications to you. Notwithstanding any other term or condition of the agreement and this ordering document, you shall not cause or permit alteration of any VAR Applications. For purposes of this section, "alteration" refers to all alterations, translations, upgrades, enhancements, customizations or modifications of all or any portion of the VAR Applications including all

reconfigurations, reassembly or reverse assembly, re-engineering or reverse engineering and recompilations or reverse compilations of the VAR Applications or any derivatives of the VAR Applications. You acknowledge that it shall be a breach of the agreement to utilize the relationship, and/or confidential information of the VAR Applications for purposes of competitive discovery.

The VAR Applications contain trade secrets of Oracle and Oracle's licensors and Customer shall not attempt, cause, or permit the alteration, decompilation, reverse engineering, disassembly or other reduction of the VAR Applications to a human perceivable form. Oracle reserves the right to replace, with functional equivalent software, any of the VAR Applications in future releases of the applicable program.

Contents

3 Functional and Technical Integration

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

5 Java Batch Process

6 Internationalization

7 Implementing Functional Security

Send Us Your Comments

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 Cloud Service 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.

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

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

Oracle Retail product documentation is 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. You can obtain these documents through My Oracle Support.)

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 Cloud Service 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.

> **Note:** The Merchandising Cloud Services are based on corresponding on-premises applications. References to the on-premises application names exist throughout the Merchandise Cloud Services applications and documents.

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.

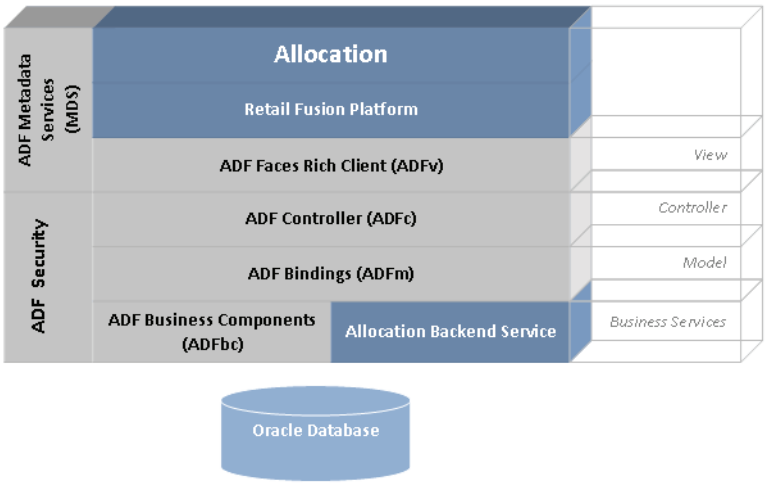
The Oracle Retail Allocation retailer benefits from the following capabilities:

- 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

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 2–1 Holdback Quantity Example

Dept	1	Hardware
Class	12	Tools

Table 2–1 (Cont.) Holdback Quantity Example

Subclass	123	Hand Tools

Table 2–2 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 2–3 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 2–4 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 2–5 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 2–5 (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 2–6 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 2–7 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 2–7 (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 2–8 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 2–9 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 2–10 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

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 2–11 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.

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.

3

Functional and Technical Integration

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

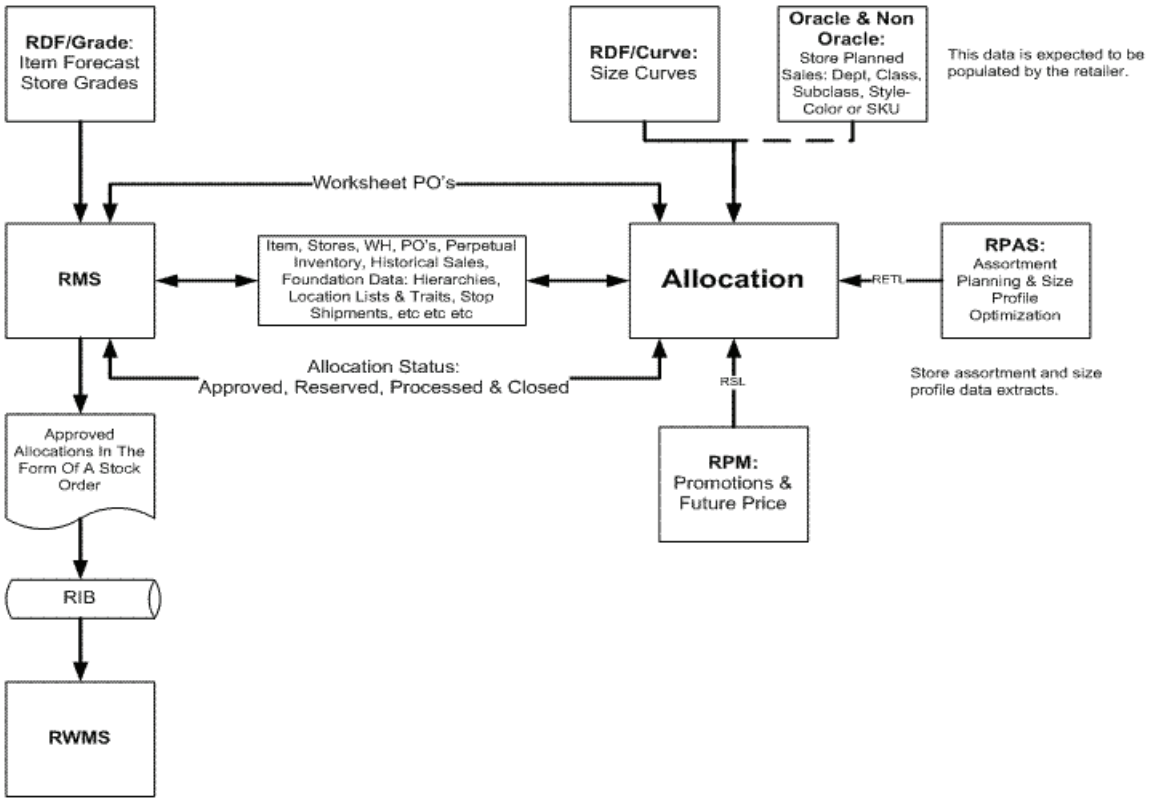
- An integration interface allocation-related dataflow across the enterprise.
- The tables and triggers that are in external systems or related to external systems that Oracle Retail Allocation uses (for example, RMS).
- A functional description of RMS dependencies and assumptions that affect Oracle 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 3–1 Oracle Retail Allocation-Related Dataflow Across the Enterprise



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 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 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 3–1 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 3–1 (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 3–1 (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 3–2 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 3–2 (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.

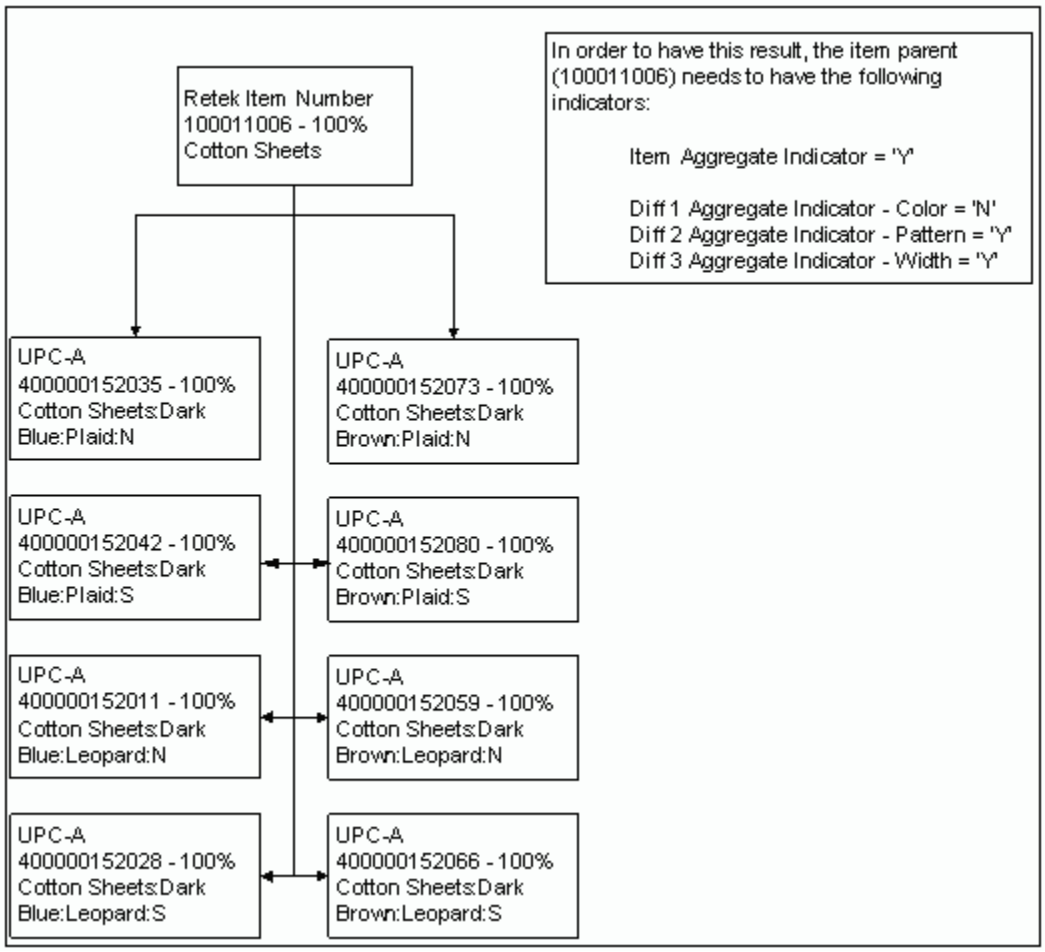
The Aggregate check box is enabled when more than one differentiator is being created for an item where the Item Aggregate Indicator has been selected. The differentiator that the customer wants to be curved by Oracle Retail Allocation must be the only differentiator that is *not* indicated on the Item 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 3–2 RMS Differentiator Setup



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.

4

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).

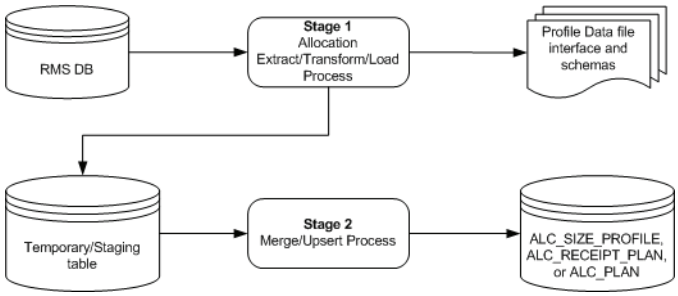
- Department-level financial plan or profiles include a department data value in the dedicated department field. The class, subclass and item fields do not contain any values. They remain empty.
- Class-level financial plan or profiles include a department and class data value in the dedicated department 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 4–1 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 4–1 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 4–1 (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_ grandpare nt					Use item_ grandpare nt as Style if item_ grandpare nt_ aggregate_ ind = 'Y'
		item_ aggregate_ ind					
		item_ parent_ aggregate_ ind					
		item_ grandpare nt_ 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 4–1 (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_ grandpare nt_ aggregate_ ind = 'Y'
		item_ aggregate_ ind					
		item_ parent_ aggregate_ ind					
		item_ grandpare nt_ 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 4–1 (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

This section describes the 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)

- 1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123

- 141410001000 1000000014 _CCOLOR31_SMEDIUM

- 20051225000000137500

Table 4–2 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)

- 12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123
- 141410001000 1000000014 _CCOLOR31_SMEDIUM
- 20051225000000137500

Table 4–3 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*X*_*prodlevel*_*.NN*

Example: File Name: d1scls.01 (diff 1, subclass, domain 01)

Record:

- 12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345
- 100045078 1000000002 CCOLOR01_S30x32 000000137500

Table 4–4 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 4–5 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.

5

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 four categories:

- Schedule Allocation batch
- Daily Cleanup batch
- Purge batches
- Rule Level On Hand (RLOH) batches
- Dashboard Refresh batch

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.

The Dashboard Refresh batch refreshes both Stock to Sales and Top to Bottom Dashboard reports Data.

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 5–1 Allocation's Batch Process and associated Java Packages

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
Dashboard Refresh Batch	AlcDashboardCleanUp.ksh	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 Dashboard Refresh 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 AlcDashboardCleanUp.ksh file is present.
3. Run the AlcDashboardCleanUp.ksh batch using the following command:

```
ksh AlcDashboardCleanUp.ksh <systemadministratoralias>
```

The batch runs by taking the batch user from wallet.

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 5–2 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	
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 5–3 RLOH Aggregated Tables

Table	Candidate Key	Source Tables
SUBCLASS_ITEM_LOC_SOH_EOD	Dept	ITEM_LOC_SOH
	Class	ITEM_MASTER
	Subclass	
	Loc	
SUBCLASS_ON_ORDER_EOD	Dept	ORDLOC
	Class	ORDHEAD
	Subclass	ITEM_MASTER
	Loc	PACKITEM_BREAKOUT
	On_order_date	

Table 5–3 (Cont.) RLOH Aggregated Tables

Table	Candidate Key	Source Tables
SUBCLASS_ALLOC_IN_EOD	Dept	ALLOC_DETAIL
	Class	ALLOC_HEADER
	Subclass	ITEM_MASTER
	Loc	PACKITEM_BREAKOUT
	Alloc_in_date	ORDHEAD
		TSFHEAD
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.

6

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 .java files. When you select a different language from the Preferences screen, Allocation will choose the correct .java files 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.

7

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.*

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

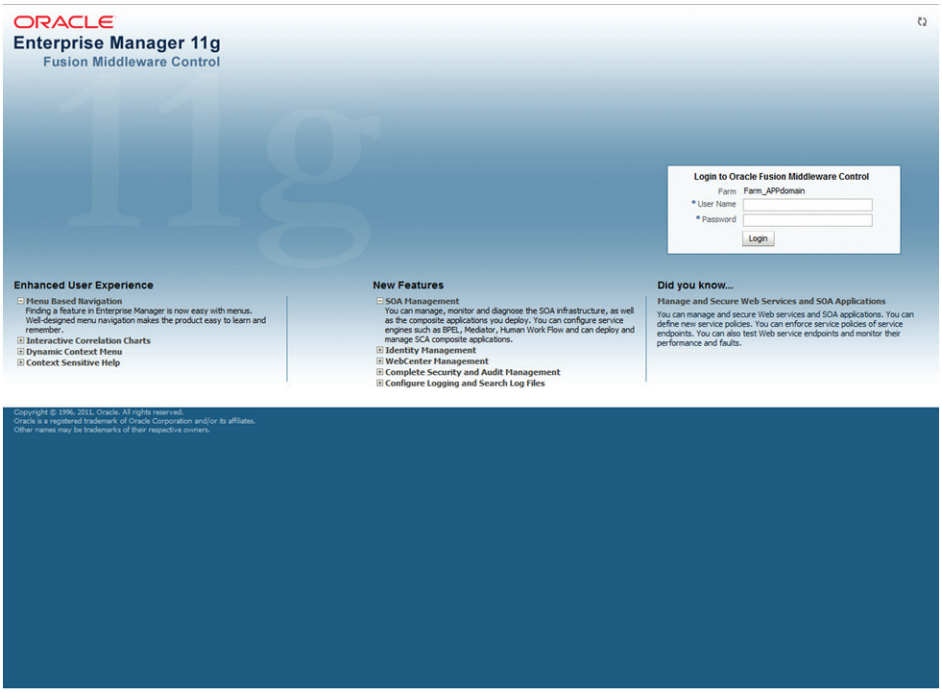
Use the following procedure to display the security menu in Fusion Middleware Control.

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 7–1 Logging in to Fusion Middleware Control



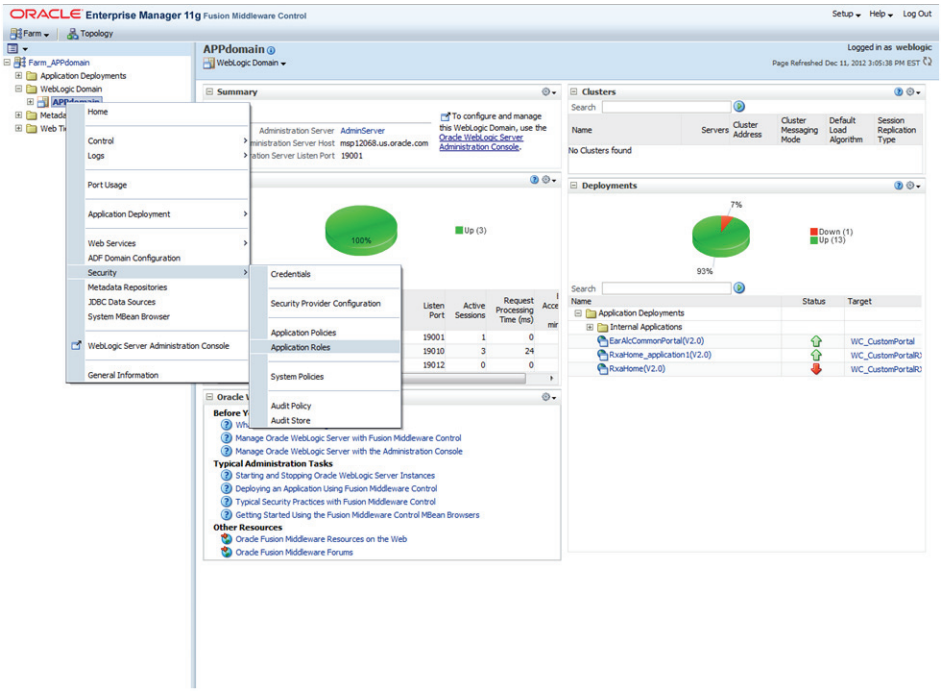
2. Enter the Retail Fusion application's administrative user name and password and click **Login**.

   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:

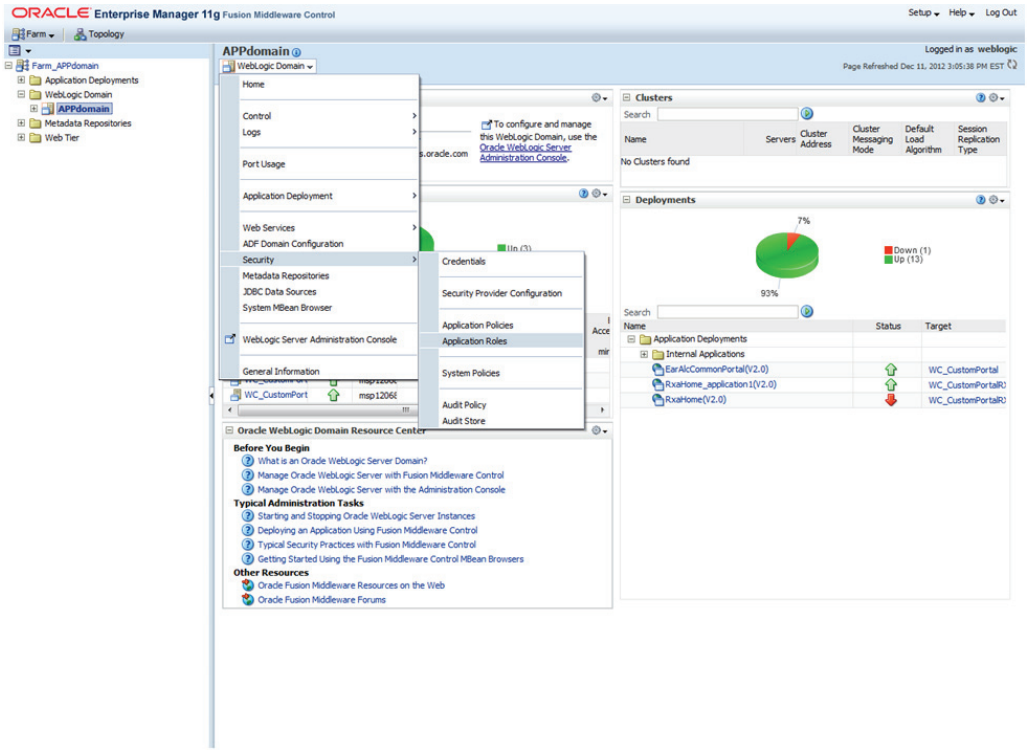
   - Right-click the application domain and hover over Security in the popup menu to display a submenu.

Figure 7–2 Displaying the Security Menu via Right-Clicking



- 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 7–3 Displaying the Security Menu via the WebLogic Domain Menu



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

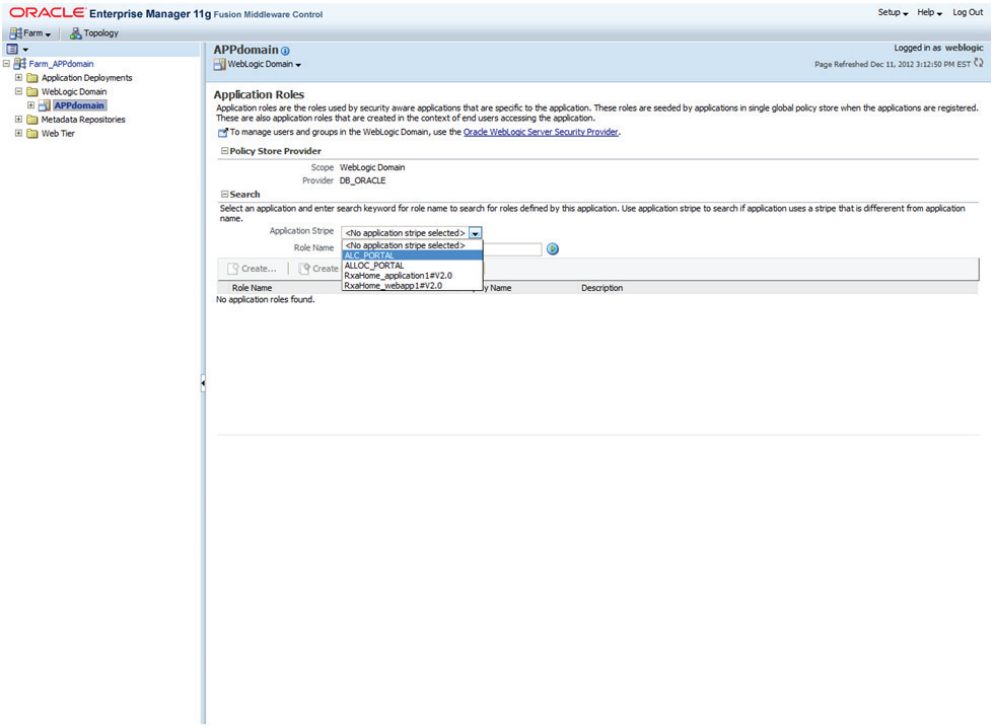
Use the following procedure to add or remove members from an application role.

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".

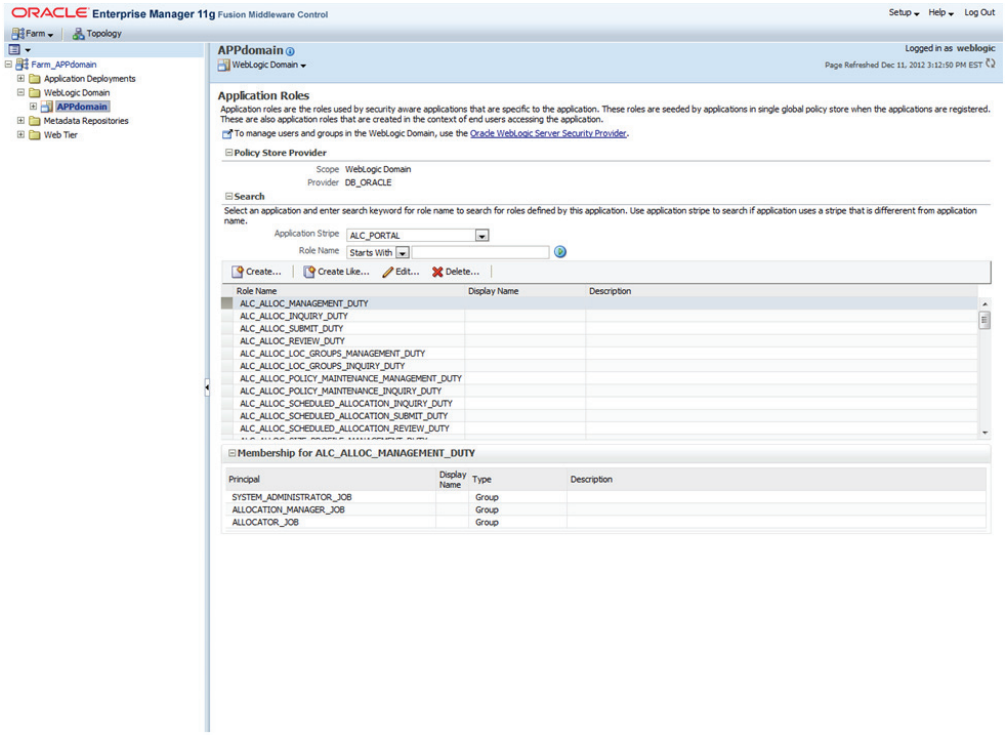
2. Choose Select Application Stripe to Search, then select the policy stripe name (for example: ALC_PORTAL) from the list. Click the search icon next to Role Name.

Figure 7–4 Application Roles Window



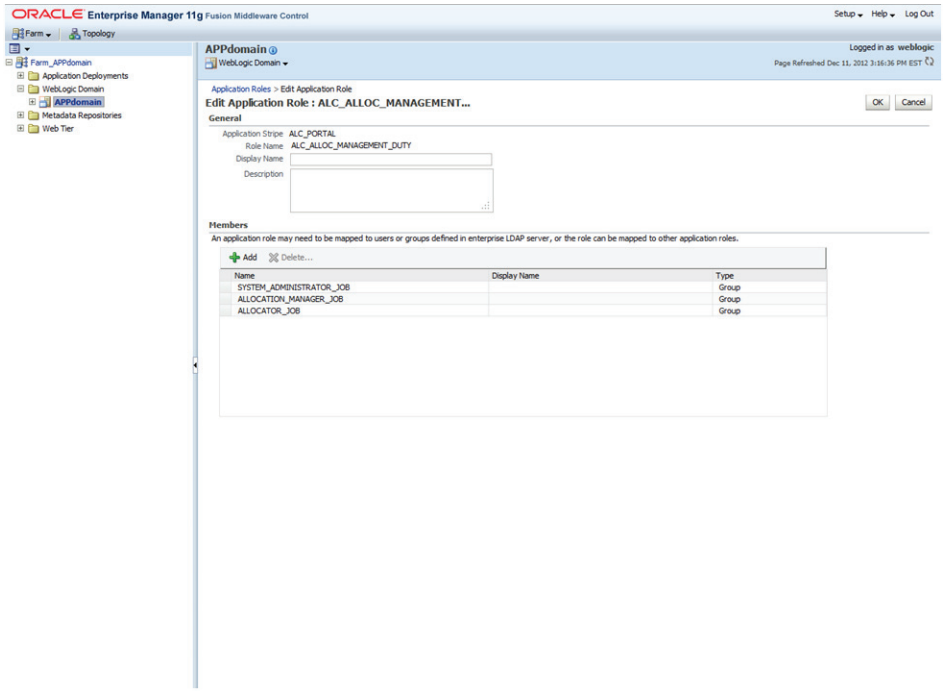
The Retail Fusion Application's application roles are displayed. As an example, in the following figure the default application roles are shown.

Figure 7–5 Viewing the Default Application Roles



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 7–6 Editing the Application Role



You can add or delete members from the Edit Application Role page. Valid members are application roles and groups.

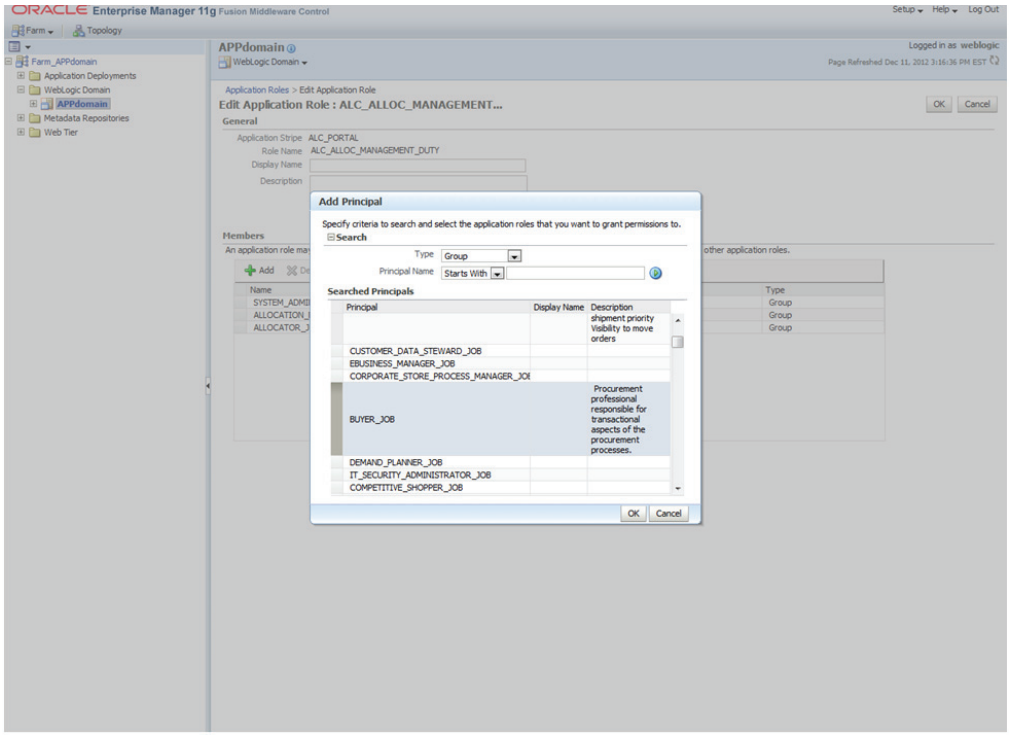
4. Select from the following options:

   - 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 7–7 Adding a Group



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.

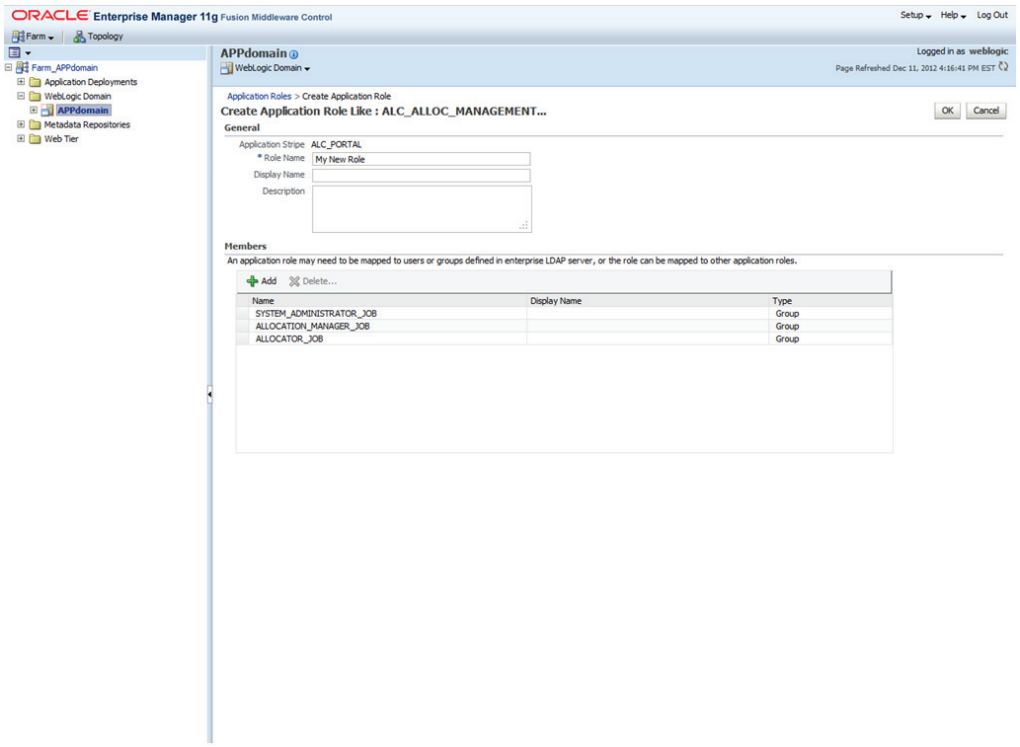
4. Click **Create Like** to display the Create Application Role Like page.

   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 7–8 Copying an Application Role



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.