Oracle® Retail Allocation

Operations Guide

G39277-01
July 2025

Oracle Retail Allocation Operations Guide,

G39277-01

Contents

5 ReST Services

Send Us Your Comments

Oracle® Retail Allocation Cloud Service Operations Guide, Release 22.1.401.0

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

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

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

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

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

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

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

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

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

Preface

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

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

Audience

This guide is for:

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

Documentation Accessibility

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

Access to Oracle Support

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

Customer Support

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

https://support.oracle.com

When contacting Customer Support, please provide the following:

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

Review Patch Documentation

When you install the application for the first time, you install either a base release (for example, 16.0) or a later patch release (for example, 16.0.1). If you are installing the base release and additional patch releases, read the documentation for all releases that have occurred since the base release before you begin installation. Documentation for patch releases can contain critical information related to the base release, as well as information about code changes since the base release.

Improved Process for Oracle Retail Documentation Corrections

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

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

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

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

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

Oracle Retail Documentation on the Oracle Technology Network

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.

Convention	Meaning
monospace	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1 Introduction

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

> ⓘ **Note**
>
> Users should not access Oracle Retail Allocation during the Merchandising batch window as it may cause some unpredictable results.

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 Merchandising tables, with no need for batch modules. Purchase order, item, location, and allocation information are passed from Merchandising to a warehouse management system.
- Access control to the system is 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.

Merchandising 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 Merchandising. 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

Integration

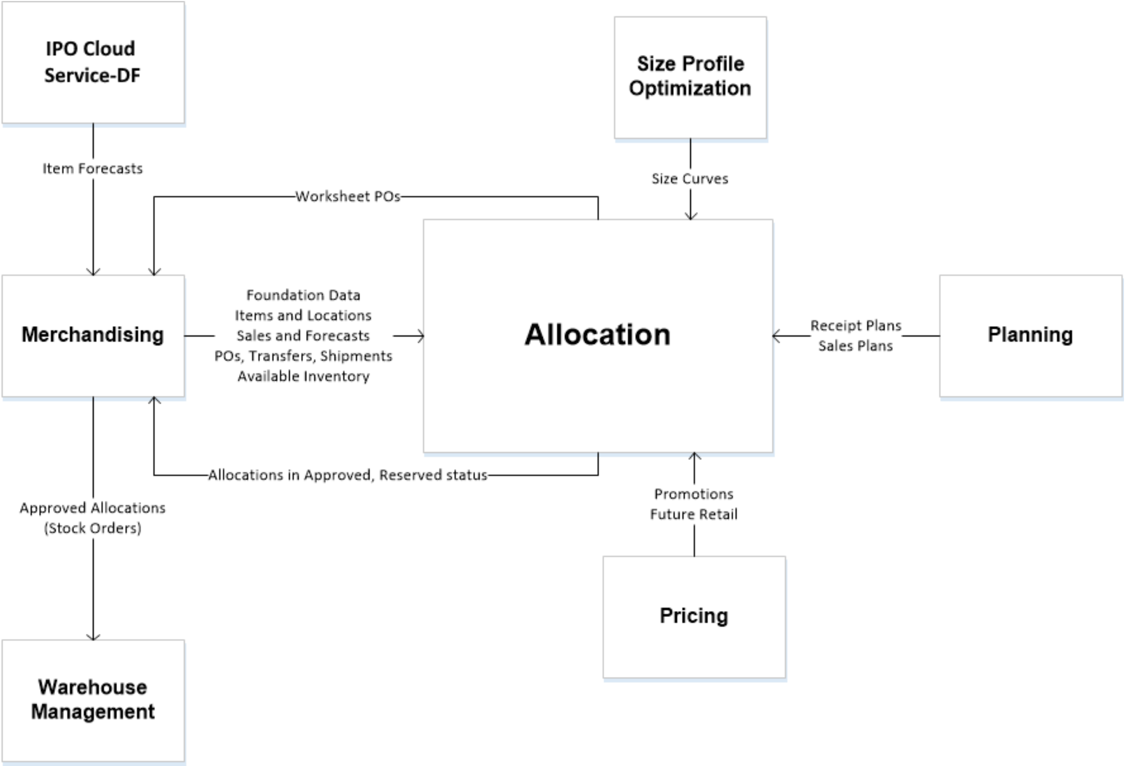
This chapter discusses the integration among Oracle Retail Allocation and other systems and it provides the following:

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

Allocation Integration Overview

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.

Figure 2-1 Oracle Retail Allocation-Related Dataflow



> ⓘ **Note**
>
> Oracle Retail Allocation pulls the data from the Merchandising tables 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 Inventory Planning Optimization-Demand Forecasting

Oracle Retail Allocation pulls the following data from Merchandising:

- Forecasting data: Oracle Retail Allocation accesses forecasting data that originates in the Oracle Retail Inventory Planning Optimization-Demand Forecasting (IPOCS-Demand Forecasting) system. IPOCS-Demand Forecasting 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. IPOCS-Demand Forecasting 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.

From Planning

Oracle Retail Allocation accesses plan data that originates in planning applications (including Oracle Retail's planning applications). The Oracle Retail Planning 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 and execute it. Both the Oracle Retail and the legacy planning applications populate a planning table, ALC_PLAN, which resides within Oracle Retail Allocation.

> ⓘ **Note**
>
> Merchandising is the system of record for Oracle Retail Allocation. Hence Allocation inherits the merchandise hierarchy from Merchandising. However, Merchandising and Assortment Planning (AP) can have different merchandise hierarchies. If you wish to export information from AP to Allocation, you must ensure that the AP merchandise hierarchy is compatible with that of Merchandising/Allocation.

From Size Profiles

Oracle Retail Allocation uses size profile data from planning or science based solutions, like Oracle Retail Size Profile Optimization (SPO). The size profile data can be imported at the following levels: department level, class level, sub-class level, and/or item level. The data for all the levels are imported in a single file. For more information, see the "Planning and Size Profile Integration" chapter.

From Warehouse Management System

Allocation does not integrate with a WMS directly. Instead, it leverages its integration with Merchandising to have access to the following data:

- Appointments

  Appointment data is one source that identifies item(s) to be allocated.
- Warehouse inventory
- ASNs, BOLs, and Transfers

From Oracle Retail Pricing

Pricing provides the following to Allocation:

- **Future Retail Price Data -** Oracle Retail Allocation has the ability to get a real time price from Pricing as it is integrated directly with Pricing. 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 Merchandising

> ⓘ **Note**
>
> Item, purchase order, supplier, sales and other data are accessed directly from the Merchandising tables, with no need to interface data via batch modules.

- Items
- Purchase Orders
- 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 Allocation to Merchandising

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

- Approved or reserved allocation data
- Worksheet status POs that contain product, supplier and quantity information (the only remaining actions to be taken in Merchandising 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.

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

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

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
- Location
- 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 Merchandising 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 Merchandising in four ways:

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

Merchandising Tables used by Allocation

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

Table 2-1 Merchandising Tables Used by Allocation

Merchandising Tables	
Functional Area	Associated Tables
Item data	SUB_ITEMS_HEAD
	SUB_ITEMS_DETAIL
	ITEM_MASTER

Table 2-1   (Cont.) Merchandising Tables Used by Allocation

Merchandising Tables	
	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
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

Table 2-1 (Cont.) Merchandising Tables Used by Allocation

Merchandising Tables	
	BUYER
	STORE_GRADE_STORE
Location traits data	LOC_TRAITS
	LOC_TRAITS_MATRIX
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
Sales data	DEPT_SALES_HIST
	CLASS_SALES_HIST
	SUBCLASS_SALES_HIST
	ITEM_LOC_HIST
	ITEM_PARENT_LOC_HIST
Appointment data	APPT_HEAD
	APPT_DETAIL

Oracle Retail Allocation-Owned Triggers Residing on Merchandising Tables

Table 2-2 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 Merchandising 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 Merchandising table ALLOC_HEADER is updated to 'C' (closed).
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 number to closed if all ALLOC_HEADER records have been closed.

3

Planning and Size Profile Integration

As noted above, Allocation has the ability to take in plan and size profile information from planning and optimization solutions. This is done by leveraging the integration programs described in this chapter. These jobs all use a flat file sent from the external solution, which is loaded using SQL Loader into Allocation and then processed into the Allocation tables.

Functional Overview

The extracts from a planning solution may contain up to four levels of plans or profiles, department level, class level, subclass level, or item level. All of these levels are contained in a single normalized file. Each record in the size profile file has a dedicated space and distinct position for department, class, subclass, item, store, diff1, diff2, diff3, diff4 and size profile quantity values. 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/size 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).
- 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.
- Department, Class, and Subclass Level
  - Department-level financial plans or size 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 plans or size 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 plans or size 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 records contain only the non-aggregate diff values mapped from the specific diff value for items in Merchandising 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
  - Item-level profiles include aggregate level IDs in the dedicated item field. The item level export records contains both the aggregate and non-aggregate diff values mapped from the specific diff id for items in Merchandising 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.

Processing

Processing involves reading the input files placed in the batch incoming folder, which is /u01/retail/rms/batch/incoming/alloc

1. Generating a dat file based on the input files.
2. The dat file is processed using sqlloader.
3. The dat is loaded into a staging where the validations are performed.
4. Inserting and updating the plan or profile records into the final destination ALC_PLAN, ALC_RECEIPT_PLAN, or ALC_SIZE_PROFILE table respectively.
5. Update quantity when matched department, class, subclass, style, store, size1, size2, size3, size4.
6. Otherwise, insert records.

Running the Module

These ad hoc jobs can be run from POM (Process Orchestration and Monitoring) batch scheduler.

Log File Location

/u01/retail/alloc/alloc-batch/batch/data/logs

Plan Data Import

Processing Job:

ALC_PLAN_JOB: Configured in POM as an ad hoc job and should be run on demand after placing the input files in the batch incoming folder as mentioned above.

Plan Data File Layout

Plan data is provided to the allocation interfaces as a flat file. This section describes the file format for uploading plan data.

File Name: p1prodlevel.NN

- The file name should start with p1 followed by four characters for product level and the domain number. The four product levels acceptable are:
  - itpt - for item
  - scls - for subclass
  - clss - for class
  - dept - for department
- NN is for the domain ID - this should always be 01.

Example: p1scls.01 (subclass, domain 01)

Table 3-1 Plan Data File Layout

Field Name	Start Position	Width	Format	Content
Product ID	1	25 char	Alpha	141410001000 Dept length=4 Class length=4 Subclass length=4 Or Item length =25
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 Product ID, Location ID and Diff IDs fields are left justified and blank filled.
- Product Id can be dept/class/subclass or item.
- 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 (for example, 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.

Example:

```
1414            1000000000       _CCOLOR 01_SSZ SMALL_PPAT 04_ESCENT 01       19910130000000050000
```

Receipt Plan Data Import

Processing Job:

ALC_RECEIPT_PLAN_JOB: Configured In POM as an ad hoc job and should be run on demand after placing the input files in the batch incoming folder as mentioned above.

Receipt Plan Data File Layout

Receipt plan data is provided to the allocation interfaces as a flat file. This section describes the file format for uploading receipt plan data.

File Name: p1prodlevel.NN

- The file name should start with p1 followed by four characters for product level and the domain number. The four product levels acceptable are:

  - itpt - for item
  - scls - for subclass
  - clss - for class
  - dept - for department
- NN is for the domain ID - this should always be 01.

Example: p1scls.01 (subclass, domain 01)

Table 3-2 Receipt Plan Data File Layout

Field Name	Start Position	Width	Format	Content
Product ID	1	25 char	Alpha	141410001000 Dept length=4 Class length=4 Subclass length=4 Or Item length =25
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 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 (for example, 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.

Example:

```
1414          1000000000     _CCOLOR 01_SSZ SMALL_PPAT 04_ESCENT 01     19910130000000050000
```

Size Profile Data Import

Processing Job:

ALC_SIZE_PROFILE_JOB: Configured In POM as an ad hoc job and should be run on demand after placing the input files in the batch incoming folder as mentioned above.

Size Profile File Layout

Size profile data is provided to the allocation interfaces as a flat file. This section describes the file format for uploading size profile data.

File Name: dXprodlevel.NN

- 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 levels acceptable are:
  - itpt - for item
  - scls - for subclass
  - clss - for class
  - dept - for department
- The domain ID should be numeric.

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

Table 3-3 Size Profile File Layout

Field Name	Start Position	Width	Format	Content
Product ID	1	25 char	Alpha	100045078 Dept length=4 Class length=4 Subclass length=4 Or Item length =25
Location ID	26	20 char	Alpha	1000000002
Diff IDs	46	48 char	Alpha	CCOLOR01_S30x32
Quantity	94	12 char	Numeric	000000137500

- 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 (for example, 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.

Example:

```
100078404        1000000005     _C1_A4                       030000000000
```

Seasonal Size Profiles

Oracle Retail Size Profile Optimization has the ability to create seasonal size profiles and multiple store size profiles, which are assigned a unique ID called a called a generation ID (GID). If available, these are displayed to the Allocation user as options. This ID, along with a user defined name will be displayed in the Allocation user interface.

The batch for SPO data file format is as follows:

```
<Beginning of file>
<GID>
<GID_DESC>
<End of File>
```

Size Profile GID text file (which must be named spo_gid_label.txt) is passed along with the batch of Size Profile Hierarchy dat file described above. The text file is used as the GID for that batch of dat file. Running Batch for SPO imports data to three tables after extraction.

Table 3-4 Size Profile 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

These three programs for receipt plan, sales plan, and size profile have the following limitations:

- The diff type is supports a maximum of 1-character length.
- The diff id is supports a maximum of 10-character length.

4

Batch Processes

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 packages associated with the batches, and how to execute the Java-based batches.

> ⓘ **Note**
>
> Details around incoming/outgoing file storage are available in the "Merchandising File Transfer Service" section of the *Merchandising Operations Guide - Vol 2*.

Batch Processing Overview

Allocation contains a set of batch processes that are run in Java. Broadly, the batch processing falls into five categories:

- Schedule Allocation batch - ScheduledAllocationBatchClient.java creates the child allocations for parent allocations that are scheduled for the day.
- Daily Cleanup batch - SessionCleanUpBatchClient.java deletes data from the temporary tables used by the Allocations and Calculation engine.
- Purge batches - Purge batches delete Allocation and Worksheet data from the Allocation tables, which were created before a certain time period.
- Rule Level On Hand (RLOH) batches - For RLOH, there are six batch update processes that share the same java batch file; InventorySnapshotBatchClient.java.
- Dashboard Refresh batch - The Dashboard Refresh batch refreshes both Stock to Sales and Top to Bottom Dashboard reports Data.

Batch Processes

The following table describes Oracle Retail Allocation's batch processes:

Table 4-1 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 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 4-2 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	

Table 4-2 (Cont.) Scripts to initiate the deletion process

Executable Shell Scripts (UNIX)	Executable Batch File For Windows	Description
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 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 (for example, 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 - Partition name in the wallet that stores the credentials to authenticate batch user on WebLogic. For example, alloc13
- csm.wallet.path - 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 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 Merchandising allocations in the ALC_XREF table are deleted when the Merchandising allocations they are linked to no longer exist in Merchandising.
- Allocations that are not linked to Merchandising 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 Merchandising at the transaction level item level. Departments in Merchandising 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 Merchandising 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 4-3 RLOH Aggregated Tables

Table	Candidate Key	Source Tables
SUBCLASS_ITEM_LOC_SOH_EOD	Dept Class Subclass Loc	ITEM_LOC_SOH ITEM_MASTER
SUBCLASS_ON_ORDER_EOD	Dept Class Subclass Loc On_order_date	ORDLOC ORDHEAD ITEM_MASTER PACKITEM_BREAKOUT
SUBCLASS_ALLOC_IN_EOD	Dept Class Subclass Loc Alloc_in_date	ALLOC_DETAIL ALLOC_HEADER ITEM_MASTER PACKITEM_BREAKOUT ORDHEAD TSFHEAD

Table 4-3 (Cont.) RLOH Aggregated Tables

Table	Candidate Key	Source Tables
SUBCLASS_CROSSLINK_EOD	Dept Class Subclass Loc	TSFHEAD TSFDETAIL ITEM_MASTER PACKITEM_BREAKOUT
SUBCLASS_ALLOC_OUT_EOD	Dept Class Subclass Loc Alloc_Out_Date	ALLOC_DETAIL ALLOC_HEADER ITEM_MASTER ORDHEAD PACKITEM_BREAKOUT
ALC_SUBCLASS_CUST_ORDER_EOD	Dept Class Subclass Loc	TSFHEAD TSFDETAIL ITEM_MASTER 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.

5

ReST Services

This chapter gives an overview about the Allocation ReSTful Web service implementation and the APIs used in Allocation. For more information on the ReST architectural style applied for building Web services, access the following URL:

http://www.oracle.com/technetwork/articles/javase/index-137171.html

To provide visibility to the background processing taking place, services write to the JOB_AUDIT_LOGS table in the database. Reports can be built based on this to provide visibility to the activities occurring in the background. Additionally, to assist users and developers in troubleshooting any error that may arise, the payload that was processed is also stored in the JOB_AUDIT_PAYLOAD table in the database.

Once the nightly batch run has started, web service execution will be halted and users will receive a warning message that the nightly batch run has commenced.

Introduction

Allocation ReST currently supports a web service with the ability to create, update, and delete corporate rule data within the solution. The ReSTful Web services' Java code cannot be customized.

Common Characteristics

Services are secured using J2EE-based security model.

- Realm-based User Authentication: This verifies users through an underlying Realm. The username and password are passed using HTTP basic authentication.
- Role-based Authorization: This assigns users to roles; authenticated users can access the services with Allocation application roles or custom roles that are assigned to:
  - For Allocation: ALLOCATION_APPLICATION_ADMINISTRATOR_JOB
- The communication between the server and client is encrypted using one-way SSL. In non-SSL environments, the encoding defaults to BASE-64 and so it is highly recommended that these ReST services are configured to be used in production environments secured with SSL connections.
- If you are using Merchandising data filtering, that will apply to the services as well. The user ID used for the calling the service should be added to the Merchandising SEC_USER table (APP_USER_ID), and then associated to the appropriate group in SEC_USER_GROUP table.

Standard Request and Response Headers

Allocation ReSTful web services have the following standard HTTP headers:

```
Accept:  application/xml or
        application/JSON
Accept-Language: en-US
```

> **Note**
>
> Accept-Language is not mandatory and defaults to en-US. User can change it in case they need content in a specific language.

Depending on the type of the operation or HTTP method, the corresponding response header is updated in the HTTP response with the following codes:

- GET/READ: 200
- PUT/CREATE: 201 created
- POST/UPDATE: 204
- DELETE: 204

Standard Error Response

Example response payload in case of service error is depicted below:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<messagesRDOes>
  <messagesRDO>
    <message>REST Service Version Mismatch</message>
    <messageType>ERROR</messageType>
    <status>BAD_REQUEST</status>
  </messagesRDO>
</messagesRDOes>
```

- Message: The error message - translated.
- Message Type: Value of ERROR is returned.
- Status: For a bad request or error, the status is BAD_REQUEST.
- The http error code for an error response is 400.

Allocation URL Paths

Based on your implementation, you will need to prefix the end point with just the deployment hostname. The format that should be used for the hostname is:

```
rex.retail.<Region Name>.ocs.oraclecloud.com/<Customer Subnamespace>/
```

where <Region Name> is the region for your data center and <Customer Subnamespace> is specific to your company's name and environment (production, stage, and so on).

The following links provide access to the Allocation services:

- The ReSTful Web services swagger URL is available at:

  https://<hostname>/MerchIntegrations/swagger-ui/
- The ReSTful Web services openapi.yaml is available at:

  https://<hostname>/MerchIntegrations/services/openapi.yaml
- The ReSTful Web services WADL file is available at:

  https://<hostname>/MerchIntegrations/services/application.wadl
- The ReSTful Web services are available at:

  https://<hostname>/MerchIntegrations/services/private/<service>

OAuth for REST Service Authentication

OAuth 2.0 is the industry standard protocol for authorization. Allocation cloud services REST Services support OAuth 2.0. To invoke these services, you need to obtain and access token and use it as a bearer token.

> **Note**
>
> Basic Authentication access is no longer supported for these services.

Steps to Enable

To obtain a token and call the services, use the following steps:

1. This step should be performed one time for a given environment. Generate the OAuth Client App with the environment-specific scope.

   Each environment has a different scope that is available after the provisioning of the environment. Use this scope to generate the OAuth Client App using the Retail Home UI. For details, refer to the "Creating OAuth Client Apps" section of the *Retail Home Administration Guide*, which can be found under Common Cloud Services Solutions in the Oracle Retail Help Center. Once the OAuth Client App is generated, note the client ID and secret. This information will be required for an access token to call REST services.
2. Get an access token using the OAuth client ID and secret from IDCS with the following command:

   ```
   export ACCESS_TOKEN="$(curl -u <Client ID>:<Secret> -H 'Content-Type: application/x-www-form-urlencoded;charset=UTF-8' --request POST https://<IDCS_BASE_URL>/oauth2/v1/token -d 'grant_type=client_credentials&scope=<Customer Environment Specific Scope>' | jq -r '.access_token')"
   ```

   The token is generally valid for 1 hour.
3. ReST clients that need to call the Allocation ReST service end points should use the client ID and secret of the OAuth client generated in the previous step to get an access token. OAuth tokens can also be obtained by ReST client tools like Postman for testing purposes

by filling in the necessary details like client ID/secret and scope. Use the below information in such cases:

```
Authorization: OAuth 2.0
Access Token URL: https://<IDCS_BASE_URL>/oauth2/v1/token
Client ID: <Client id of OAuth client app>
Client Secret: <Client secret of OAuth client app>
Scope: <Custom environment specific scope>
```

The scope pattern that is used in the MFCS IDCS app creation template is rgbu:merch:<SERVICETYPE>-<ENVIRONMENT> where SERVICETYPE is MFCS and ENVIRONMENT is the environment type (STG, PRD, UAT, DEV1, DEV2, and so on).

For example:

```
"scope": "rgbu:merch:MFCS-PRD"
"scope": "rgbu:merch:MFCS-STG"
```

Example using the access token as a bearer token:

```
curl --location --request GET 'http://<hostname or IP address>:<port number>/RmsReSTServices/services/
private/Common/vDate' \ --header 'Authorization: Bearer $ACCESS_TOKEN'
```

Corporate Rules

Functional Area

Inventory – Transfers and Allocations

Business Overview

This service provides the ability to create, modify, and delete corporate rules within Allocation CS.

Service Type

POST

REST URL

```
/MerchIntegrations/services/inventory/allocation/corporateRules/manage
```

Roles Allowed

ALLOCATION_APPLICATION_ADMINISTRATOR_JOB, rgbu:merch:system

Request Headers

```
Accept:  <application/json or application/xml>
Accept-Language: <locale>
Content-Type: <application/json or application/xml>
```

For Example:

```
Accept:  application/json
Accept-Language: en-US
Content-Type: application/json
```

URL Query Parameters

N/A

Input Payload Details

Table 5-1 Create - Object. See list of elements for detail

Entity	Element Name	Required	Data Type	Description
	collectionSize	Yes	Integer	Collection of corporate rule details.
items	action	Yes	String	Contains the intended action for the corporate rule. Valid values are CREATE, UPDATE and DELETE.
items	corporateRuleId	No	Integer	Corporate rule identifier.
items	corporateRuleName	Yes	String	Corporate rule name.
items	netNeedIndicator	Yes	String	Specifies if the need quantity corresponding to the corporate rule is the Net Need (Y) or Gross Need (N).
items	details	Yes	List	Details of the corporate rule. At least one detail item is mandatory for header action CREATE, otherwise it is ignored.
items/details	location	Yes	String	Location identifier
items/details	dept	No	String	Department identifier
items/details	class	No	String	Class identifier
items/details	subclass	No	String	Subclass identifier
items/details	item	No	String	Item identifier
items/details	diff1	No	String	Differentiator ID that is used to distinguish the item from its parent. For an item that is a parent, this field will contain the first differentiator ID.

Table 5-1   (Cont.) Create - Object. See list of elements for detail

Entity	Element Name	Required	Data Type	Description
items/details	diff2	No	String	Differentiator ID that is used to distinguish the item from its parent. For an item that is a parent, this field will contain the second differentiator ID.
items/details	diff3	No	String	Differentiator ID that is used to distinguish the item from its parent. For an item that is a parent, this field will contain the third differentiator ID.
items/details	diff4	No	String	Differentiator ID that is used to distinguish the item from its parent. For an item that is a parent, this field will contain the fourth differentiator ID.
items/details	needQuantity	Yes	String	The need quantity for the specified hierarchy location combination.

Note

In the ‘details’ section, you need to provide one of the following parameters (depending on the level at which the rule will apply for the specified location) –

- Department
- Class
- Subclass
- Parent/Diff
- SKU

Sample Input Message

```
{
 "collectionSize": 1,
 "items": [
  {
   "action": “CREATE”,
   "corporateRuleId": 1001,
   "corporateRuleName": "Spring 2025 Men's Casuals",
   "netNeedIndicator": “Y”,
   "details": [
    {
     "location": 1121,
     "dept": 1111,
     "class": null,
     "subclass": null,
     "item": null,
     "diff1": null,
          "diff2": null,
          "diff3": null,
          "diff4": null,
          "needQuantity": 1500
        }
      ]
    }
  ]
}
```

Response Code: 200 (Success)

Sample Response Message

```
{
  "message": "SUCCESS",
  "items": [
    {
      "corporateRuleId": 1001,
      "corporateRuleName": "Spring 2025 Men’s Casuals"
    }
  ]
}
```

Response Code: 400 (Error)

In case of error, the following standard error response will be returned. The element validationErrors will be present when input payload or input parameters do not match the schema definition for this service.

Sample Error Message

```
{
  "status": "ERROR",
  "message": "NET_NEED_IND must be Y or N."
}
```

Index