Oracle® Retail Merchandising
Implementation Guide
Release 12.0.3

April 2007

ORACLE®

Oracle® Retail Merchandising Implementation Guide, Release 12.0.3

Contents

Preface

The Oracle Retail Merchandising Implementation Guide provides a high level view of how the Merchandising Operations Management applications integrate with one another.

This guide provides the following:

- How to implement Oracle Retail Merchandising Operations Management applications
- An overview of each Oracle Retail Merchandising Operations Management application
- Information that each Oracle Retail Merchandising Operations Management application maintains
- How each Oracle Retail Merchandising Operations Management application integrates with other Oracle Retail Merchandising Operations Management applications

Audience

The Implementation Guide is intended for the Oracle Retail Merchandising Operations Management applications Integrators and implementation staff, as well as the retailer's IT personnel.

Related Documents

For more information, see the following documents in the Oracle Retail Merchandising Release 12.0.3 documentation set:

- Oracle Retail Merchandising Data Conversion Guide
- Oracle Retail Merchandising Batch Schedule
- Oracle Retail Merchandising System Operations Guide
- Oracle Retail Merchandising System User Guide
- Oracle Retail Merchandising System Data Model
- Oracle Retail Merchandising System Online Help
- Oracle Retail Trade Management User Guide
- Oracle Retail Price Management Operations Guide
- Oracle Retail Price Management User Guide
- Oracle Retail Price Management Data Model
- Oracle Retail Price Management Online Help
- Oracle Retail Price Management Release Notes
- Oracle Retail Invoice Matching Release Notes

Customer Support

- https://metalink.oracle.com

When contacting Customer Support, please provide:

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

Conventions

Navigate: This is a navigate statement. It tells you how to get to the start of the procedure and ends with a screen shot of the starting point and the statement "the Window Name window opens."

> **Note:** This is a note. It is used to call out information that is important, but not necessarily part of the procedure.

```
This is a code sample
    It is used to display examples of code
```

A hyperlink appears like this.

Overview

The purpose of this implementation guide is to provide a high level view of how the Merchandising Operations Management applications integrate with one another. The guide includes the following:

- Installation Overview - This section gives a high level overview of the process required for a successful installation of the Oracle Retail Merchandising Operations Management applications. In addition, this section provides hardware and software requirements and a list of the order in which the applications must be installed (something that Services has been requesting). It is important to note that this does not replace existing install guides but does provide a one-stop book to view requirements and what to expect when installing the Merchandising Operations Management applications.

- Application chapters - These chapters provide an overview of the Merchandising Operations Management applications. They provide a fairly detailed look at how each application integrates with the other Merchandising Operations Management applications as well as the information that is being passed back and forth between those applications (i.e., what information RMS provides to other apps and what information those other apps provide to RMS). We also discuss the Users that must be created for each application as well as the Security settings that these applications can employ.

1
Installation Overview

The purpose of this section is to give a high level overview of the process required for a successful installation of the Oracle Retail Merchandising Operations Management applications. For complete step by step installation details the installation guides should be referenced for each application.

Pre-Installation

Pre-Installation requirements are tasks that should be researched and completed prior to starting the actual installation process.

Hardware Requirements

> **Note:** Hardware requirements vary between applications. See the installation guide for each of your respective applications for specific hardware requirements.

There are a number of hardware requirements required for the successful installation of the Oracle Retail Merchandising Operations Management applications. The amount, capabilities, and types of hardware required for each implementation will vary based on the client's needs and objectives. In general though most clients will need:

- **Application Server(s)** – These must be a UNIX based OS certified with Oracle Application Server 10g version 10.1.3. OS options include AIX5.2, AIX5.3, Solaris 9 (SPARC), HP-UX 11.23 (PARISC), and Oracle Enterprise Linux release 4.
- **Database Server(s)** – These must be a UNIX based OS certified with Oracle RDBMS 10g Enterprise Edition. OS options include AIX5.2, AIX5.3, Solaris 9 (SPARC), HP-UX 11.23 (PARISC), and Oracle Enterprise Linux release 4.
- **Client PC and Browser –**
  - Windows 2000 or XP
  - 512MB RAM and 1Ghz Processor
  - Sun J2RE Runtime 1.4.2 or higher
  - Microsoft Internet Explorer 5.5, 6.0 and higher

Software Requirements

> **Note:** Software requirements vary between applications. See the installation guide for each of your respective applications for specific software requirements.

There are a number of software requirements for clients to successfully install and run the Oracle Retail Merchandising Operations Management applications. These include but may not be limited to (depending on the client needs):

- Oracle Application Server 10g version 10.1.2.0.2.
- Oracle RDBMS 10g Release 2 Enterprise Edition.

- ANSI Compliant C compiler (Certified with OS and database version).
- Perl Compiler 5.0 or later.
- X-Windows Interface.
- Java 1.4.2.x (for RPM, Allocation, ReIM).
- Oracle Pro*C
- Hibernate
- Bouncy Castle
- libstdc++ (for Allocation)
- Egate (for RIB)

Database Partitioning Strategy

The Database Partitioning Strategy is a mandatory step that must be completed prior to installing any of the Oracle Retail Merchandising Operations Management applications. A spreadsheet is included with the installation materials that define requirements for both mandatory and optional partitioning. This enables the resource in charge of implementing the partitioning strategy to determine the correct strategy.

See Appendix B: Partitioning for additional information on a partitioning strategy.

Installation

Installation Order of Oracle Retail Merchandising Operations Management products

This section provides the order in which the Oracle Retail Merchandising Operations Management applications should be installed. If a client has chosen to use some, but not all, of the applications the order is still valid less the applications not being installed.

1. Oracle Retail Service Layer (RSL)
2. Oracle Retail Integration Bus (RIB)
3. Oracle Retail Extract, Transform, Load (RETL)
4. RIB for Oracle Retail Price Management (RPM)
5. Oracle Retail Merchandising System (RMS), Oracle Retail Trade Management (RTM), Oracle Retail Sales Audit (ReSA)
6. Oracle Retail Active Retail Intelligence (ARI)
7. Oracle Retail Allocation
8. Oracle Retail Invoice Matching (ReIM)
9. Oracle Retail Security Management (RSM) – Since RSM provides the security structure for Oracle Retail Price Management (RPM) it must be installed prior to RPM. See the RPM security section for more details on RSM.
10. Oracle Retail Price Management (RPM)

Installation Process for Oracle Retail Merchandising Operations Management products

Some of the Oracle Retail Merchandising Operations Management applications offer installers to facilitate the installation process while others require a manual installation. This section identifies the process used for each application.

- RMS – Manual Installation
- ARI – Manual Installation
- RDW Manual Installation

- Allocation – Installer
- ReIM – Installer
- RSM – Installer
- RPM – Installer

2
Oracle Retail Merchandising Operations Management Applications

Oracle Retail Merchandising System

Oracle Retail Merchandising System (RMS) is the foundation system that records and controls virtually all data in the retail enterprise and ensures data integrity across all integrated systems. RMS includes key functions such as item maintenance, inventory management, replenishment, and more. This functionality provides easy access to the information that is crucial to the day-to-day merchandising activities within a retail organization, providing the ability to focus on key decisions that help achieve sales and profit targets. RMS streamlines business practices and unifies business systems across retail channels to better serve customers. Since RMS has been developed as a web-based, scalable product, it fully supports the large volumes found in retail, leaving more time for retailers to concentrate on the bottom line.

Oracle Retail Trade Management

Oracle Retail Trade Management (RTM) is an import management system designed to integrate and streamline the international trade transaction process. RTM links multiple departments together for all import functions. RTM provides immediate on-line visibility to the status, location, and disposition of products as they move through the import cycle.

RTM links partners in the supply chain - suppliers, agents, bank, transportation providers, freight consolidators, customs brokers – to share a constant flow of information needed to manage the movement of goods from sources to destinations across international boarders.

As RTM is coupled with RMS, the import purchase order process also ties in with regular purchase order features such as open-to-buy, updating stock ledger and inventory. RTM provides the facility to track and capture expenses incurred in the import process, and to apportion the expenses to the actual landed cost of the inventory. The application also provides Letter of Credit payment processing; the handling payment typically used in import purchase orders.

Oracle Retail Sales Audit

Oracle Retail Sales Audit (ReSA) is an auditing system that provides a simplified sales audit process that accepts "raw" POS data and provides "clean" data to downstream applications while ensuring integrity of audited data. The application supports automatic and interactive auditing of POS sales data. The application is designed to focus on exception conditions while allowing clean data to flow through thus increasing productivity. Flexibility is provided in the creation of user defined rules and totals to configure exceptional conditions. User defined audit rules fine-tune the system to focus validation on potential problem areas, and custom totals are created online for validation of calculations such as data entry or over/short. Interactive audit functionality allows auditors to focus on exceptions and helps navigate the auditor through resolution to ensure a clean data load to integrated applications. The application validates and

balances POS transactions and tender data, detects and corrects errors according to pre-defined rules. It allows sales balancing at store/register or cashier levels. The applications help in identification, review and resolution of errors and irregularities in a timely manner.

The diagram below illustrates the audit process:

Overview of Information that Oracle Retail Sales Audit Maintains
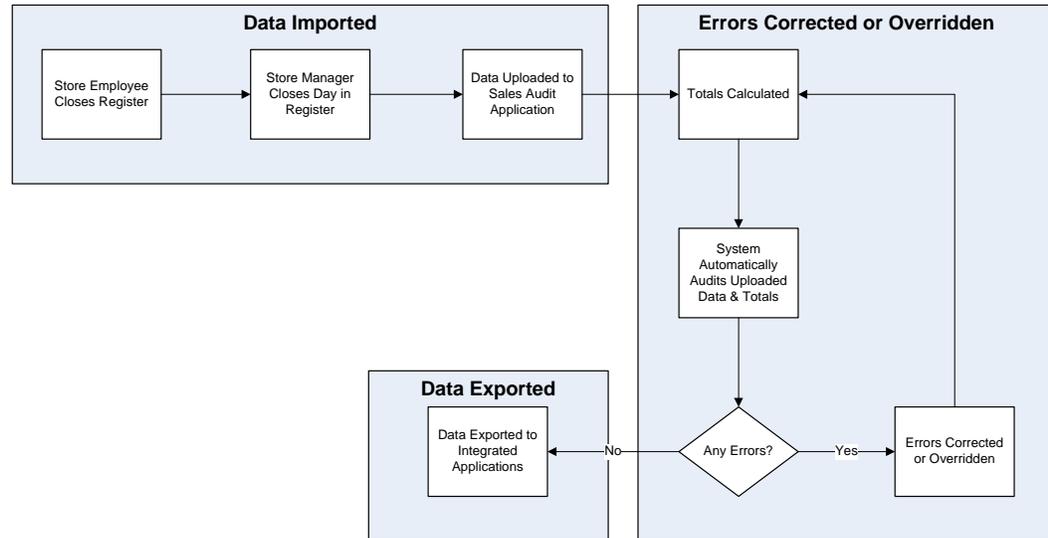
Oracle Retail Allocation

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

In Oracle Retail Allocation, the retailer has the option of executing a 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 reacts to current trends. The system has sophisticated rules that 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 store need.

Oracle Retail Invoice Matching

Invoice matching describes a control procedure designed to ensure the retailer pays the negotiated cost for actual quantities received.

ReIM is designed to support the invoice verification process with accuracy and efficiency, focusing resources on exception management. ReIM accepts electronic invoice data uploads (EDI), and provides for rapid on-line summary entry of invoices. ReIM supports automated and on-line processes allowing one or more invoices to be matched against

one or more receipts. When an invoice cost and quantities are matched within tolerance, it is ready for payment and staged to a table to allow a retailer to extract to their accounts payable solution.

If a cost or quantity difference between the invoice and receipts is outside tolerance, a discrepancy is recognized and must be resolved. A flexible resolution process allows discrepancies to be directed to the most appropriate user group for disposition. Reviewers are empowered to assign one or more reason codes that they are authorized to use, to resolve the discrepancy.

Each reason code is associated to a type of action (for example, create chargeback or receiver cost adjustment). Many reason codes may be associated with a particular action type, allowing for more granular reporting, and so on. Actions drive document creation and EDI downloads to suppliers, inventory adjustments, and accounting activities. Actions also allow the invoice to be extracted by the retailer and posted for payment.

Oracle Retail Price Management

Oracle Retail Price Management (RPM) is a pricing and promotions execution system. RPM's functionality includes the definition, maintenance, and review of price changes, clearances and promotions. The system's capabilities range from simple item price changes at a single location to complex buy/get promotions across zones.

RPM contains three primary pricing execution dialogs for creating and maintaining regular price changes, clearances, and promotions. Although each of the three pricing activities is unique, the system displays these dialogs using a common look and feel. Each of these dialogs uses the conflict checking engine which leverages RPM's future retail table.

The future retail table provides a forward looking view of all pending approved pricing events affecting an item at a given location.

RPM pricing events are defined against the zone structure. The zone structure represents groups of locations organized to support a retailers pricing strategy. RPM allows the user to break out of the zone structure and create location level events as needed.

RPM supports the definition and application of price guides to these pricing events. Price guides allow the retailer to smooth retails and provide "ends-in" rounding logic to derive a final consumer price.

RPM supports Pricing Strategies which allows a user to define how item retails are proposed when pricing worksheets are generated. The strategies are defined at department, class, or subclass in order to represent which items are affected.

RPM also supports the ability to create vendor funded promotions by either associating an existing Deal from RMS with the promotion or by creating a new Deal in RMS based on the information provided for the promotion.

Oracle Retail Active Retail Intelligence

Oracle Retail Active Retail Intelligence (ARI) is an exception management and workflow tool driven by business rules. It sits across the retail systems (Oracle Retail Merchandising Operations Management applications) and provides a central repository for alert notification and associated actions across all Oracle based applications.

ARI can be uniquely configured for each client to fit individual business needs.

ARI provides the capability to build three categories of rules:

- Exception Reporting

- Workflow Management
- Enterprise Process Modification

3

Oracle Retail Merchandising System (RMS) Overview

Overview of Information that Oracle Retail Merchandising System Maintains

Seed Data

RMS contains data that must be populated at the time of installation. This is because either the data is required by the application or because the data is static and can be loaded for any client. The codes tables (CODE_HEAD and CODE_DETAIL) are examples of tables with system-required values that should be loaded at installation. Additional codes can be added as needed to these two tables after installation using either the online forms or additional client scripts. The STATE table is an example of static data that can be used by a U.S. client without modification (although clients in other countries may want to remove the U.S. states and add the abbreviations for their own states/provinces prior to loading the data).

Additionally, some configuration tables must be populated for the application to open and function appropriately even though the configuration values may be subsequently modified. The system options tables (SYSTEM_OPTIONS and UNIT_OPTIONS) are examples of tables required for initial start-up that may be updated prior to implementation to reflect final configuration.

An inventory of the tables and supplied scripts can be found in Appendix C of this document.

Foundation Data

Foundation data is the base for all future information on which RMS builds. This information needs to be present before a client begins using the system. The majority of Foundation data is set up online or more commonly a client performs a data conversion process to import this information from legacy systems.

Foundation Data consists of three types of information; organizational hierarchy, merchandise hierarchy, and supplier and partner management.

The organizational hierarchy allows clients to create the relationships that are necessary in order to support the operational structure of a company. Clients can create a preferred organizational structure to support consolidated reporting at various levels of the company. Clients can also assign responsibility for any level of the hierarchy to a person or people in order to satisfy internal reporting requirements.

The merchandise hierarchy allows clients to create the relationships that are necessary in order to support the product management structure of a company. Clients can assign a buyer and merchandiser at the division, group, and department levels of the merchandise hierarchy. Clients can also link a lower level to the next higher level. For example, a client can indicate to which group a department belongs or to which division a group belongs.

Supplier and partner management provides functionality to address the need to create and store valid merchandise suppliers and partners. RMS provides the ability to

maintain a variety of information about suppliers such as financial arrangements, inventory management parameters, types of EDI transactions, and invoice matching attributes. Suppliers are typically created in a financial system and interfaced into RMS; supplier enrichment can be maintained in RMS.

Item Maintenance

RMS is responsible for the creation and maintenance of all items. RMS uses a flexible data hierarchy for items that consists of levels that allow a retailer to model items any way they wish. The hierarchy consists of up to three levels with the highest level being one and the lowest level three. Within the defined levels for an item family one level is denoted as the transaction level. This is where all inventory and sales transactions take place. This model gives retailers the flexibility to create families of items that share common characteristics.

RMS also has the ability to create several different types of items, such as: regular items, deposit items, packs, concession items, consignment items, and transformable items.

The item maintenance system is also responsible for the maintenance of an item's relationships with other entities such as suppliers, locations, and attributes.

Purchasing

The Purchase Order module allows clients to create and maintain PO's in a variety of ways and is ultimately responsible for providing commitments to vendors for product in specific amounts for specified locations. Purchase Orders are created manually or automatically via replenishment or from external system. They can be created against entered contracts and deals, or directly through direct store delivery or Vendor Managed Inventory (VMI). This system also provides the ability to maintain the items, locations and quantity ordered for Purchase Orders.

Contracts

The contract dialogue gives the user the ability to create contracts, maintain contracts, and submit and approve contracts.

A contract is a legally binding agreement with a supplier to supply items at a negotiated cost. In RMS, the contracting functions fit closely with the replenishment and ordering functions. The main functions of the Contracts window is to book manufacturing time, track supplier availability and commitments, and match them with business requirements. The main business benefit of contracting is to achieve supplier involvement during the planning phase of a retailer's business.

Deals

Deals management allows retailers to create and maintain deals with partners or suppliers. Deal partners can be suppliers, wholesalers, distributors, and manufactures. Within a deal, clients create deal components, specify the items for each deal component, and define thresholds.

Components are deals or parts of that deal a retailer receives from a supplier. Multiple components can exist in a single deal. After a client adds the components, they must define thresholds to define the quantity or amount that must be purchased or sold to receive the deal. RMS components include off-invoice deals, rebates, vendor funded promotions, vendor funded markdowns, and fixed deals.

Finally, clients define the items and locations where the deal can be applied. They choose to include or exclude locations as necessary.

Additionally, clients define the proof of performance (POP) terms for a deal. POP terms are defined by the deal vendor that offers the deal. For deals, POP terms are defined at the deal, deal/component or deal/component/item-loc combination. For fixed deals, the POP terms are defined at the deal level.

Cost Management

For Cost Changes the Cost Management dialogue gives clients the ability to:

- Accept cost changes received via EDI (Flat file)
- Create a cost change
- Edit a cost change
- View a cost change

A cost change is an adjustment to the supplier cost of an item, either up or down. Before clients create a cost change, a list of user defined cost change reasons must be created and then applied to each cost change. This is useful in reporting.

The initial cost of an item is established at item set-up. The cost of the item is adjusted in the item record until the status of the item is Approved. After the item is approved, any cost changes needs to be handled through the cost change dialogue.

When submitted via EDI, the EDI cost change is reviewed and released (to create a cost change document). The cost change document is then viewed and submitted for approval.

When a cost change document is created through the online dialogue, clients enter the cost change, an event description, an effective date, and a reason code. Then submit the cost change for approval.

After clients have approved the cost change, the Item/Supplier cost record is updated and any outstanding purchase order line items with no received units are recalculated if recalculation is indicated on the cost change.

Additionally, the Cost Management dialogue is used to create cost zone groups for zone-type expenses for item estimated landed cost. Zone-type expenses are incurred when imported goods are moved from the discharge port to the purchase order receiving location. Since the expenses can vary depending on the distance between the discharge port and the receiving location, cost zones can be configured to appropriately reflect the expenses. The locations (stores and physical warehouses) should be grouped to reflect the expense variances for moving the goods. Normally a zone strategy is used for these cost zone groups, but it is possible that every location within the company has different expenses to move the goods from the discharge port. If that is the case, a store strategy would be used. Additionally, if every location within the company has the same transportation costs from the discharge port, a corporate strategy is adequate (but only if multi-currency were not being used). Once these cost zone groups are defined, they are added to new items as they are created in anticipation of the expense profiles that are needed for the item.

Inventory Control

Inventory functionality in RMS is the core of the application's functionality. Inventory is tracked two ways in RMS; perpetually and financially. This overview covers perpetual inventory. The stock ledger gives an overview of RMS's financial inventory capabilities.

RMS achieves inventory control through a number of functions which include transfers, return to vendor (RTV), inventory adjustments, purchase order receipts (shipments), and stock counts.

Transfer in RMS provides an organized framework for monitoring the movement of stock within the organization. RMS is responsible for the creation and maintenance of transfers. However, clients can also interface transfer information into RMS from other systems. RMS supports a number of different types of transfers such as intercompany transfers, book transfers, PO linked transfers, externally generated transfers, and customer orders. Transfer supports the movement of one or more items between two internal RMS locations and also provides the capability for a multi-leg transfer where the intermediate location is considered a "finisher" location. Finishers are used as locations where work is performed on merchandise, such as dying fabric and attaching labels. Mass Return Transfers are used to reallocate merchandise to locations or to return merchandise to the supplier.

Return to Vendor transactions, or RTVs, are used to send merchandise back to a vendor. The RTV transaction in RMS allows for one or more items to be returned to a single vendor. For each transaction, the items, quantities and costs are specified and, upon shipment out of a location, inventory is removed from the stock on hand. RTV's are created manually in RMS or interfaced in from an external system. RMS also provides the ability to maintain existing RTVs. Shipped RTV's create a debit memo or credit note request (based on supplier configuration) on the invoice matching staging table in RMS for export to Invoice Matching.

Inventory adjustments are used to increase or decrease inventory to account for events that occur outside the normal course of business (e.g. receipts, sales, stock counts). Inventory Adjustments are created in RMS or interfaced in from an external system (store or warehouse applications). RMS supports the creation of two types of inventory adjustments; stock on hand or unavailable inventory. Inventory adjustments can also be created by locations for multiple items, by item for multiple locations, or via a product transformation for a specific location.

Purchase Order Receipts (Shipments) record the increment to onhand when goods are received from a supplier. Weighted average cost (WAC) is recalculated at time of receipt using the PO landed cost. Transaction audit records are created for financial audit and the receiver is made available to invoice match for matching.

Stock Counts are the process by which inventory is counted in the store and compared against the system inventory level for discrepancies. RMS supports two types of stock counts; unit and unit and value. Unit stock counts adjust the on hand quantities for the item/locations affected and create an inventory adjustment transaction for the stock ledger. Unit and value stock counts adjust the on hand quantities for the item/locations affected and adjust the stock ledger to the results of the stock count.

Replenishment

Automated replenishment is a system that constantly monitors inventory conditions and based on these creates purchase orders or transfers to fulfill consumer demand.

Automated replenishment parameters are set up at the supplier, supplier/department, supplier/location or supplier/department/location level. These parameters include:

- review cycle and order control
- due order processing
- investment buy attributes
- scaling constraints
- rounding attributes
- supplier minimums

- truck splitting constraints

Items can be setup on automated replenishment via the item maintenance dialogue either individually or via item lists.

Automated replenishment also supports ten different methods for determining if purchase orders are created and how much is ordered. These methods are:

- **Constant –** This is a stock-oriented method in which the item is replenished when the inventory level falls below a specified level.

- **Min/Max –** This is a stock-oriented method in which the item is replenished up to the max when the inventory level falls below a specified minimum stock level.

- **Floating Point –** This is a stock-oriented method in which the item is replenished when the inventory level falls below a dynamic system calculated maximum stock level.

- **Time Supply –** This is a stock-oriented method in which replenishment is based on the number of days of supply for the item a retailer wants in inventory. The Time Supply method requires a forecasting system.

- **Time Supply Seasonal –** This is the same as time supply but takes seasonality and terminal stock into account. The Time Supply Seasonal method requires a forecasting system.

- **Time Supply Issues –** Used only by warehouse, this is the same as time supply but uses warehouse issues forecast rather than store sales forecast. The Time Supply Issues method requires a forecasting system.

- **Dynamic –** Controls inventory using dynamic calculations of order point and order quantities based on a number of factors including forecasted sales over order lead time, review lead time, inventory selling days, lost sales factor, and safety stock. The Dynamic method requires a forecasting system.

- **Dynamic Seasonal –** This is the same as dynamic but takes seasonality and terminal stock into account. The Dynamic Seasonal method requires a forecasting system.

- **Dynamic Issues –** Used by warehouses only, this is the same as dynamic but uses warehouse issues forecast rather than store sales forecast. The Dynamic Issues method requires a forecasting system.

- **Store Orders –** This method allows replenishment to look at the store order need quantity when determining the recommended order quantity.

These replenishment methods are applied at the item/location level in the system.

Stock Ledger

The stock ledger in RMS records the financial results of the merchandising processes that occur in the system, such as buying, selling, price changes, transfers, etc. All of these transactions are recorded in the RMS stock ledger and rolled up to the subclass/location level for days, weeks and months (depending on calendar settings). The aggregate levels in the stock ledger are used to measure inventory amounts and merchandise profitability. For the most part the stock ledger is used for reporting purposes; however there is some on-line visibility as well.

The Oracle Retail stock ledger supports multiple currencies. All transaction-level information is stored in the local currency of the store or warehouse where the transaction occurred. As transaction level information is rolled up to the aggregated levels in the stock ledger, records are kept in local currency and converted to primary currency. This allows corporate reporting to be performed in the primary currency of the

company, while still providing visibility by location to the profitability in the local currency.

The stock ledger supports both the retail and cost methods of accounting. The cost method may use standard cost or average cost depending on how the system is configured. The stock ledger supports both the retail (4-5-4) and the normal (Gregorian) calendar. If the retail calendar is used, data is maintained by the 4-5-4 month and the week. If the normal calendar is used, data is maintained only by the Gregorian month. Data can also be maintained daily using the retail (4-5-4) or normal (Gregorian) calendar.

Investment Buy

Investment buy facilitates the process of purchasing inventory in excess of the replenishment recommendation in order to take advantage of a supplier deal or to leverage inventory against a cost increase. The inventory is stored at the warehouse or in outside storage to be used for future issues to the stores. The recommended quantity to 'investment buy', that is, to order, is calculated based on the following:

- Amount of the deal or cost increase
- Upcoming deals for the product
- Cost of money
- Cost of storage
- Forecasted demand for the product, using warehouse issue values calculated by Oracle Retail Demand Forecasting
- Target return on investment (ROI)

The rationale is to purchase as much product as profitable at the lower cost and to retain this profit rather than passing the discount on to customers and stores. The determination of how much product is profitable to purchase is based on the cost savings of the product versus the costs to purchase, store and handle the additional inventory.

Investment buy eligibility and order control are set at one of these four levels:

- Supplier
- Supplier-department
- Supplier-location (warehouse locations only)
- Supplier-department-location

Warehouses must be enabled for both replenishment and investment buy on RMS' WH (warehouse) table. In a multi-channel environment, virtual warehouses are linked to the physical warehouse.

The investment buy opportunity calculation takes place nightly during the batch run, after the replenishment need determination, but before the replenishment order build. The investment buy module IBCALC.PC attempts to purchase additional inventory beyond the replenishment recommendation in order to achieve future cost savings. Two distinct events provide the incentive to purchase investment buy quantities:

- A current supplier deal ends within the look-ahead period.
- A future cost increase becomes active within the look-ahead period.

The calculation determines the future cost for a given item-supplier-country-location for physical warehouse locations only.

If the order control for a particular line item is 'buyer worksheet', it may be modified in the buyer worksheet dialog, and can be added to either new or existing purchase orders.

RMS Integration with other Applications

RMS is a Web based forms application that provides direct database access and has database logic as outlined in the diagram below. Since other Oracle Retail Merchandising Operations Management applications share the same database schema as RMS, information is primarily shared via calls to RMS packages and procedures or via direct database access.
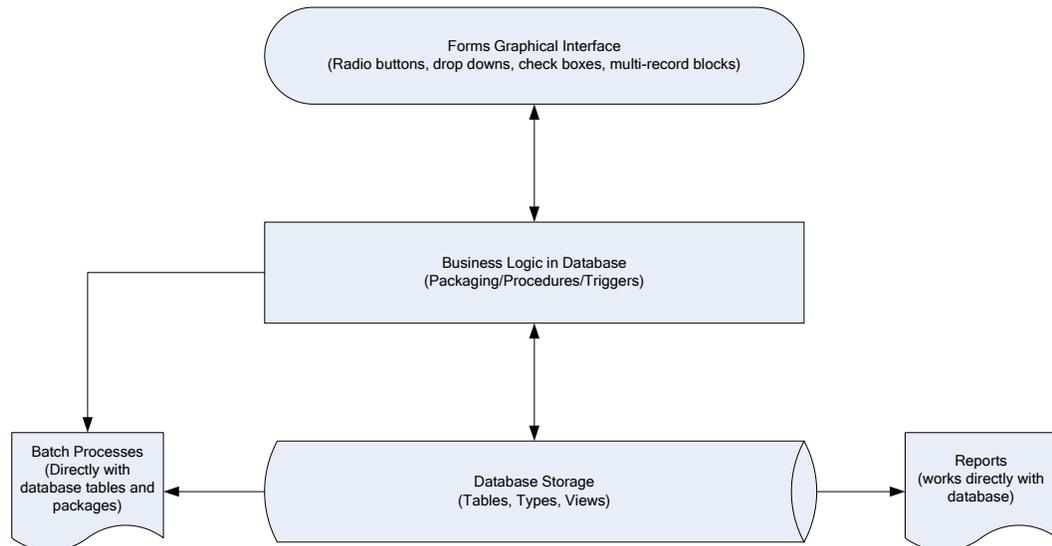
RMS provides essential information to all of the Oracle Retail Merchandising Operations Management applications, and interacts with all of them. RMS exists on the same database schema as all of the other applications which provides flexibility in how information is shared between RMS and the other Oracle Retail Merchandising Operations applications.

Information is shared with other Oracle Retail Merchandising Operations Management applications through direct reads from Oracle Retail Merchandising Operations Management application tables, calls to Oracle Retail Merchandising Operations Management application packages, batch processes, and the Oracle Retail Integration Bus (RIB) if the client is using this option.

The diagram below illustrates where RMS exists in the merchandising footprint:
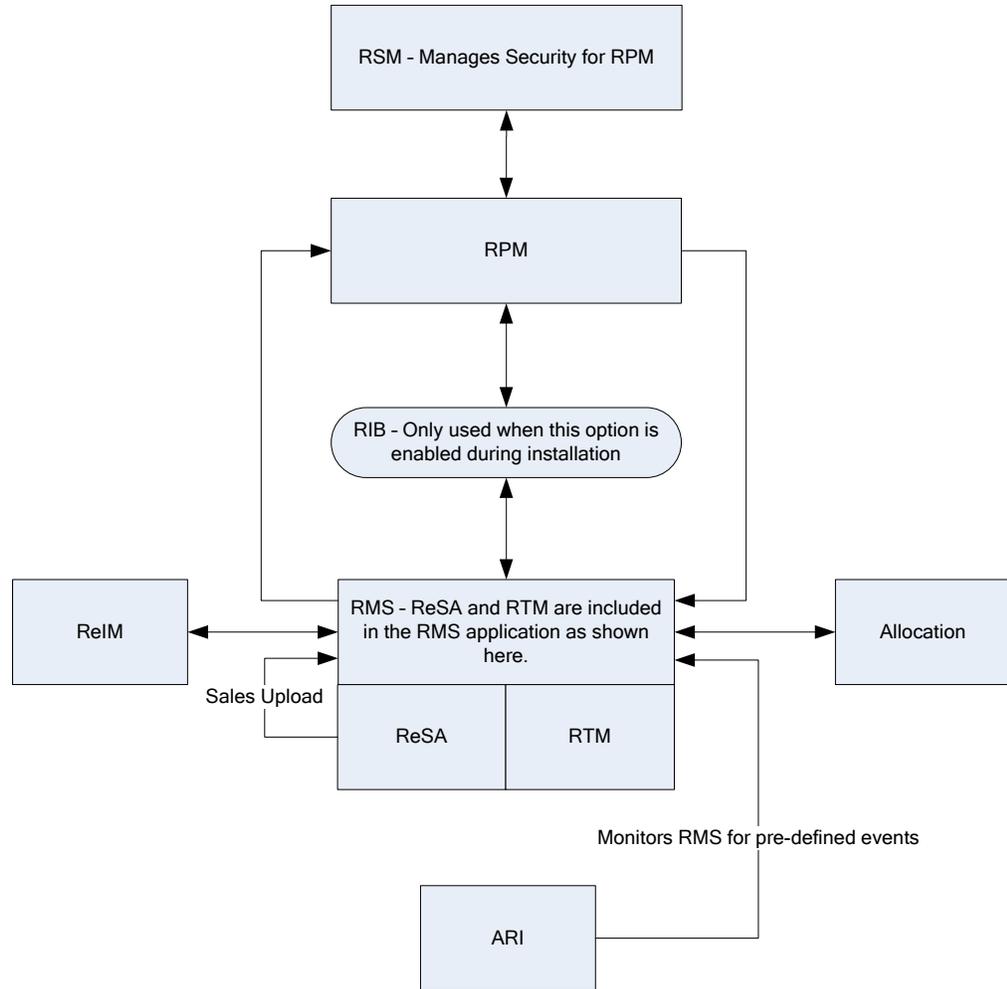
RMS and RTM

Oracle Retail Trade Management (RTM) and RMS share the same database instance. When RTM is enabled in an RMS instance, certain import specific data maintenance is required for country, supplier, partners and items. These are directly updated into the RMS database and subsequently used in RTM.

RMS and ReSA

Oracle Retail Sales Audit (ReSA) and RMS share the same database. ReSA shares some of its master data with RMS. Foundation data such as stores, company/location close dates, location traits, bank setup and tender types are maintained in RMS and used in ReSA.

Sales Upload Process - Current reference data is retrieved from RMS into ReSA by the batch program SAGETREF. The data is extracted into multiple data files. The data in the files is used by the batch program SAIMPLOG as reference data for doing validation checks on the POS transaction data during the data upload from Point of Sales (POS) to ReSA. Having the reference in data file formats increases the performance of the SAIMLOG process. SAGETREF generates the following reference files: Items, Wastage, Sub-transaction level items, Primary variant relationships, Variable weight PLU, Store

business day, Code types, Error codes, Credit card validation, Store POS, Tender type, Merchant code types, Partner vendors, Supplier vendors, Employee IDs, Banner IDs.

All clean and audited sales and returns data is extracted from ReSA into a POSU file by the batch program SAEXPRMS. All sale and return transactions that do not have RMS errors are extracted into the file. The sales audit system options parameter "work unit" controls the export of data into files in case of the presence of RMS errors in the POS transaction data. The batch program POSUPLD loads the data from the POSU file into the RMS tables.

RMS and RPM

There are a number of different options for clients to choose from when implementing both RMS and RPM. RPM exists on the same database schema as RMS which allows information to be shared between applications through direct database reads, package calls, and batch processes. RPM also optionally interacts with RMS by using the RIB. RPM now offers the flexibility to forgo using the RIB for RMS and RPM installations through the use of a system setting. If RIB functionality is disabled RPM uses APIs to facilitate the exchange of information with RMS that would otherwise be conducted through the RIB. It is the client's responsibility to disable existing database triggers that support the RIB functionality should they choose not to implement it.

RPM provides RMS:

RIB Implementation:

- **Regular Price Change Approval/Modification/Deletion –**RPM publishes regular price change messages when a regular price change is created, modified, or deleted (for approved price changes). RMS subscribes to this message to generate (or remove if deleting) ticket request information for the regular price change request.

- **Promotional Price Change Approval/Modification/Deletion –** I RPM publishes promotional price change messages when a promotional price change is created, modified, or deleted (for approved promotions). RMS subscribes to this message to generate (or remove if deleting) ticket request information for the promotional price change request.

- **Clearance Price Change Approval/Modification/Deletion –** RPM publishes clearance price change messages when a clearance price change is created, modified, or deleted (for approved promotions). RMS subscribes to this message to generate (or remove if deleting) ticket request information for the clearance price change request.

- **Price Change Execution –** When regular, promotional, or clearance price changes are set to go into effect or end the PriceEventExecutionBatch owns the process. Once the pricing event has been processed by the batch program it updates item/location pricing in RMS by interfacing with the RMSSUB_PRICECHANGE API in RMS.

- **Initial Pricing –** Initial pricing for items in RMS is dependant upon the primary zone group for the item defined in RPM and characteristics of that zone group. These characteristics include markup percent, markup percent type, and pricing guides. RPM provides this information to RMS through an API (MERCH_RETAIL_API_SQL).

- **Deal Creation –** RPM creates and associates Deals with regular, promotional, and clearance price changes. When this occurs RPM uses an RMS API (PM_DEALS_API_SQL) to create the deal in RMS.

Non-RIB Implementation:

- **Regular Price Change Approval/Modification/Deletion –** Regular price change creation, modification, or deletion triggers a call to an RMS API to generate (or remove if deleting) the ticket request information.

- **Promotional Price Change Approval/Modification/Deletion –** Promotional price change creation, modification, or deletion triggers a call to an RMS API to generate (or remove if deleting) the ticket request information.

- **Clearance Price Change Approval/Modification/Deletion –** Clearance price change creation, modification, or deletion triggers a call to an RMS API to generate (or remove if deleting) the ticket request information.

- **Price Change Execution –** When regular, promotional, or clearance price changes are set to go into effect or end the PriceEventExecutionBatch owns the process. Once the pricing event has been processed by the batch program it updates item/location pricing in RMS by interfacing with the RMSSUB_PRICECHANGE API in RMS.

- **Initial Pricing –** Initial pricing for items in RMS is dependant upon the primary zone group for the item defined in RPM and characteristics of that zone group. These characteristics include markup percent, markup percent type, and pricing guides. RPM provides this information to RMS through an API (MERCH_RETAIL_API_SQL).

- **Deal Creation –** RPM creates and associates Deals with regular, promotional, and clearance price changes. When this occurs RPM uses an RMS API (PM_DEALS_API_SQL) to create the deal in RMS.

RMS provides RPM with:

- **Foundation Data –** This information is essential to RPM functionality. To successfully setup price changes RPM requires RMS merchandise hierarchy, organizational hierarchy, and suppliers. RPM is able to access this information via the RMS database.

- **Item –** Any price change created in RPM ultimately relates to an item/location within RMS. RPM needs to know all eligible items currently in the merchandising system, the locations at which they are eligible (item/location relationships in any status but 'D' – delete requested), and suppliers associated with the items. RPM is able to access this information via the RMS database.

- **Competitive Pricing Information –** RPM has the ability to create price changes based off competitive activity in the marketplace. RPM is able to access this information via the RMS database.

- **Deals –** Deals can be associated to price changes in RPM (including vendor funded promotions). In order to associate a price change to an existing deal RPM needs visibility to the deals currently available in the RMS system. RPM is able to access this information via the RMS database.

- **Event Notification –** There are certain events (outlined below) that can occur in RMS that RPM needs to be notified of to ensure the appropriate processing occurs.

  - RIB Implementation
    - **Store/Warehouse Creation -** When new stores and warehouses are created in RMS, RPM needs to add them to a zone structure. To do this RMS provides RPM with the store and/or virtual warehouse being added, its pricing location, and its currency (to ensure it is the same as the zone it is being added to). When a client is using the RIB for an implementation this

notification process is handled through message publication and subsequent RPM subscription.

– **Item/Location Creation –** When new item/location relationships are established, RPM needs to verify that no future retail records currently exist, create an initial future retail record (for sellable items), and determine if there are any existing price changes that would affect the item resulting in a future retail record for the price change as well. When a client is using the RIB for an implementation this notification process is handled through message publication and subsequent RPM subscription.

– **Item Modification –** This event is used to notify RPM when an item is reclassified. The details of the reclassification are written to an item modification table in RPM for the next batch processing run. When a client is using the RIB for an implementation this notification process is handled through message publication and subsequent RPM subscription.

– **Department Creation –** This event is used to notify RPM when new departments are created in RMS. RPM creates aggregation level information for the new department using predefined system defaults. When a client is using the RIB for an implementation this notification process is handled through message publication and subsequent RPM subscription.

- Non-RIB Implementation

  – **Store/Warehouse Creation -** When new stores and warehouses are created in RMS, RPM needs to add them to a zone structure. To do this RMS provides RPM with the store and/or virtual warehouse being added, its pricing location, and its currency (to ensure it is the same as the zone it is being added to). When a client is using a non-RIB implementation a store/virtual warehouse creation event in RMS triggers an API call to RPM to execute the necessary processing.

  – **Item/Location Creation –** When new item/location relationships are established, RPM needs to verify that no future retail records currently exist, create an initial future retail record (for sellable items), and determine if there are any existing price changes that would affect the item resulting in a future retail record for the price change as well. When a client is using a non-RIB implementation an item/location creation event in RMS triggers an API call to RPM to execute the necessary processing.

  – **Item Modification –** This event is used to notify RPM when an item is reclassified. The details of the reclassification are written to an item modification table in RPM for the next batch processing run. When a client is using a non-RIB implementation an item modification creation event in RMS triggers an API call to RPM to execute the necessary processing.

  – **Department Creation –** This event is used to notify RPM when new departments are created in RMS. RPM creates aggregation level information for the new department using predefined system defaults. When a client is using a non-RIB implementation a department creation event in RMS triggers an API call to RPM to execute the necessary processing.

RMS and Allocation

RMS provides Allocation with:

- **Foundation Data** - This information is essential to all areas of Allocation including valid locations to allocate to and from, location groupings, and valid merchandise hierarchies to allocate within.
- **Item** - Allocations are generated at the item location level so it is necessary that the Allocation application understand what items and item/locations are eligible in the system.
- **Purchase Order** – One of the sources from which a user can allocate from is Purchase Orders. Allocation relies on RMS to provide Purchase Order information.
- **Transfer** – One of the sources from which a user can allocate from is Transfers. Allocation relies on RMS to provide Transfer information.
- **Bill Of Lading (BOL)** – One of the sources from which a user can allocate from is a BOL. Allocation relies on RMS to provide BOL information.
- **Advance Shipment Notices (ASN)** – One of the sources from which a user can allocate from is an ASN. Allocation relies on RMS to provide ASN information.
- **Inventory** – In order to determine the correct need at an item location level before performing an allocation the application needs visibility to the current on hand inventory at each location being allocated to. Allocation relies on RMS to provide inventory information at the item/location level.
- **Shipping Information** – One of the sources which the Allocation application can allocate from is an ASN. Allocation relies on RMS to provide ASN information.
- **Sales Information** - Allocation can use historical sales, forecast sales, and plan sales in order to determine the need at an item/location level for an allocation. Allocation interfaces this information in from external planning system to an Allocation table.

Allocation provides RMS/RTM/ReSA with:

- **Allocations** - Once allocations have been moved to Approved or Reserved status the Allocation is written to RMS tables to give visibility to the allocation results.
- **Purchase Orders created by "What-If" process in Allocation** – If this option is enabled in Allocation the client can create a simulated allocation based on current need and then automatically create a purchase order from that Allocation in RMS to fulfill that need. Allocation uses an RMS API to build the purchase order in RMS.

Invoice Matching and RMS

RMS provides Invoice Matching with:

- **Foundation Data** - This information is essential to all parts of invoice matching including valid locations for Invoices to be executed at, valid suppliers to receive invoices from, and supplier addresses to send credits and debits based on invoice matching results.
- **Item** - This information is essential to the invoice matching process as item information ensures that invoices being received are valid for the business. For example an item received on an invoice is carried by the client, is supplied by the supplier who sent the invoice, and is carried in the locations for which the item was received.

- **Purchase Orders** – Purchase Orders are used by Invoice Matching to facilitate the invoice matching process which is performed at the purchase order location level.

- **Shipments** – Shipment information is used by Invoice Matching to determine if a PO has been received yet which affects the matching algorithm used by the AutoMatch batch program in Invoice Matching.

- **Deals and Rebate** – Invoice Matching creates credit memos, debit memos, and credit requests based on deal and rebate information in RMS for processing by the financial (AP) system. This is performed by the ComplexDealUpload and FixedDealUpload batch processes that read from RMS staging tables.

Invoice Matching provides RMS with:

- **Invoice Matching results for shipments** – Shipment records are updated with the invoice matching results from the invoice match process (this involved updating the match status and quantity matched of the shipments in question). The matching process is handled by the AutoMatch batch process in Invoice Match which attempts to match all invoices in ready-to-match, unresolved, or multi-unresolved status.

- **Receiver Cost Adjustments** – When invoice matching discrepancies are resolved via a Receiver Cost Adjustment, information is placed on a staging table in Invoice Matching for export to RMS to perform the cost adjustment on the affected purchase order and shipment. The ReasonCodeActionRollup batch in Invoice Matching is responsible for inserting these records on a staging table in Invoice Matching for export to RMS. A trigger on the Invoice Matching staging table completes the transaction by updating RMS with the cost adjustment information.

- **Receiver Unit Adjustments** - When invoice matching discrepancies are resolved via a Receiver Unit Adjustment, information is placed on a staging table in Invoice Matching for export to RMS to perform the unit adjustment on the affected order and shipment. The ReasonCodeActionRollup batch in Invoice Matching is responsible for inserting these records on a staging table in Invoice Matching for export to RMS. A trigger on the Invoice Mathing staging table completes the transaction by updating RMS with the unit adjustment information.

- **Closing unmatched shipments** – Invoice matching closes the invoice matching status for shipments in RMS after a set period of time (defined by the client in system options). This updates the invoice matching status of the shipment on the shipment table in RMS. This process is managed by the ReceiptWriteOff batch program.

RMS and ARI

ARI is a monitoring system that interacts with any applications database (including RMS). As such, it does not "use" any information from RMS, rather it monitors the RMS database for events defined by a client and notifies the client when said events occur.

RMS Users and Security

User roles must be created within RMS and provide users with a mechanism for accessing the system. When security is leveraged, it controls a user's access to individual application functions and data sets.

Users in RMS are setup by the database administrator using a user creation script. RMS users are database users that connect directly to the database to access RMS. Each database user has synonyms to the master RMS schema and are able to update and modify data within that schema but are unable to change the structure of the tables and objects. This user structure is specific to RMS, ReSA and RTM. Applications such as

RPM, Allocation and ReIM use other means to manager users. User management for each application is discussed within its respective section.

In order to ensure users are limited to parts of the application and information that is relevant to their business role, RMS has a three tier security structure.

The three levels of security offered by RMS are:

- Database Level Security – A built in feature of Oracle's Database based on database role(s)
- Application Level Security – Form or Screen level security based on database role(s)
- Data Level Security – Built into RMS to give a client the ability to further limit user access to information

Database Level Security

Database Level Security is a feature provided by the Oracle Database and can be used by RMS. Since this is a database feature, this level of security is wholly maintained within the database and is not configurable via RMS. In order take advantage of this feature a client needs to associate all users in the system to an Oracle Role. Oracle Roles are defined in terms of business roles for a client as it is the business role of a user that should drive that user's security requirements. Developer, Inventory Planner, Assistant Buyer, Merchandiser are sample roles.

Database security is set up by first identifying the different business roles that exist within a client's organization. These roles need to have their responsibilities mapped to the appropriate table operations, table access, package calls, and form access. These roles are then defined as Oracle Roles with the corresponding table operations and access. At this point users are associated with the appropriate Oracle Roles which defines database security for the users.

Database security also provides an additional feature by allowing a client to define Oracle Roles as Secure Application Roles. This allows a client to attach a procedure to granting a user Secure Application Roles. The procedure is defined by the client and contains the logic for determining what attributes a user needs to have to be given access to the role in question. This provides an additional layer of security for a client around how a user can gain access to a specific role effectively limiting how people can gain access to privileged information.

Application Level Security

Application Level Security restricts the user's access to either entire areas of the system (for example, Purchase Orders) or restricts the modes in which user's can access areas (for example, viewing Purchase Orders only). Application Level security consists of five primary components:

- Menu Level Security – This allows a client to determine which menu options are available to a user based on that user's Oracle Role. Menu level security is set up using the SEC_FORM_ACTION and SEC_FORM_ACTION_ROLE tables to define what options each Oracle Role has access to for the various menus in RMS.
- Navigator – When RMS is launched the user operates from a folder driven interface on the Oracle Retail Start form. These folders provide access to all of the functional areas available in RMS. Navigator security is established via the Start Tree Administration form. From here the client defines the folder tree structure in the system in addition to defining security access to forms throughout the tree by Oracle Role. A client could also develop a script to perform these updates. The updates affect the NAV_FOLDER, NAV_ELEMENT, NAV_ELEMENT_MODE, and

NAV_ELEMENT_MODE_ROLE tables. Like the form (which updates these tables) these tables control the folder tree structure, the forms accessible from the tree structure, and the user roles cleared for access to the different forms.

- Find Forms – This is the most common entry point into RMS's core functionality. Within this form the user typically has the option to elect what type of action they want to perform (New, Edit, or View) from an action list box. Find form security is set up via the P_SECURITY package within the find forms. This logic is hard coded into each of the find forms, as a result these packages need to be modified once Oracle Roles have been defined for a client if customized security is required.

- Hierarchy Forms – These forms differ from find forms in that they have New, Edit, and View buttons. Hierarchy form security is managed through the P_SECURITY package within the forms. However, it differs from the find forms in that it leverages the SEC_FORM_ACTION and SEC_FORM_ACTION_ROLE tables to drive what options each Oracle Role has access to.

- Form Link Security – The FORM_LINK tables allow a client to restrict access and visibility of certain item maintenance subforms by Oracle Role.

Data Level Security

Data Level Security restricts a user's access to specific data within the merchandising system. The client has the ability to limit user's data access both from a merchandise hierarchy perspective in addition to an organizational hierarchy perspective. For example, a buyer for the Small Appliances department could have data level security put in place so that they only have the ability to access items within the Small Appliances department. This prevents users from accessing information that does not pertain to their job.

Unlike the other layers of security, this level of security can be configured in the RMS application itself. The client can accomplish this by:

1. Defining different groups within the organization and what merchandise hierarchy and group hierarchy information these groups should have.

2. Using the Security Group Maintenance form to establish each of the groups defined in step 1 in RMS.

3. Using the Filter Level Group Hierarchy Maintenance form to establish the relationships between the groups built in step 2 and the merchandise and organizational hierarchy components they are cleared to access based on the analysis of step 1.

4. Using the Security User/Group Link form to establish the relationships between the groups defined in step 2 and the user's who are tied to those groups (which was established in step 1).

Users that are not associated to a security group or are associated to a security group that was not associated to any parts of the organizational or merchandise hierarchy are considered "Super Users" in terms of data. In other words, they have access to all data within the system.

When all of the security tools for RMS are effectively leveraged by a client a user can become a powerful tool to not only grant access to a system, but to ensure that every employee is systematically focused on the aspect of the client's business that they are responsible for.

Internationalization

The technical infrastructure of RMS supports the following languages other than English. The following languages are available in addition to English:

- French
- German
- Spanish
- Portugese (Brazilian)
- Japanese
- Korean
- Chinese (Simplified)
- Chinese (Traditional)
- Russian

The software efficiently handles multiple languages. The RMS application contains tables to accommodate internationalization. The client sets up the user's language preferences. RMS determines the user's language setting and displays the code string associated with it. RMS has a fail/safe mechanism built into the code. If the user's preference language string is not found, then RMS rolls back to English.

3
Oracle Retail Trade Management (RTM) Overview

Overview of Information that Oracle Retail Trade Management Maintains

Master Data

Oracle Retail Trade Management (RTM) shares the same database with RMS. RTM is enabled in an RMS instance by the following system options parameter settings:

- Simplified RTM Indicator is 'N'
- Import Indicator is 'Y'

RTM requires certain master data maintenance such as outside locations, Freight Type, Freight Size and Standard Carrier Alpha Codes (SCAC). Import specific data maintenance is required for country, supplier, partners and items. For calculation of tax and duties applicable on import merchandise, the Harmonized Tariff Schedule (HTS) files need to be uploaded into the system.

Landed Costs

Landed cost is the total cost of an item received from a vendor inclusive of the supplier cost and all costs associated with moving the item from the supplier's warehouse or factory to the purchase order receiving location. RTM facilitates the setting up of various cost components, associating them to the purchase orders, calculating the estimated landed costs at the time of purchase order creation. It also facilitates the tracking and booking of the actual costs after the receipt process.

Estimated Landed Cost (ELC) is composed of cost components from the Supplier, Trading Partners, Item and Origin Country, which are brought together during Purchase Order creation to develop an estimate of costs associated with purchasing a particular item on the current PO.

The components of landed cost are defined using Expenses, Assessments, and Computation Value Bases (CVBs).

Expenses

Expenses are direct and indirect costs incurred in moving a purchased item from the supplier's warehouse/factory to the purchase order receiving location. Expenses should not be confused with upcharges which allow add-on costs from an initial receiving location to a final retail location and are not part of the landed cost.

An example of a direct expense is the packing cost or insurance cost. Charges incurred for clearing and loading goods at the lading port are an example of indirect costs. Expenses are either added to the base inventory value or booked as a separate expense. Expenses apportioned to inventory affect the weighted average cost (WAC) of the item. Expenses can be assigned to a particular country, supplier, or partner. Expenses are tracked at two levels – country or zone level, which are detailed below.

Country Level Expenses

Country level expenses track the costs of bringing merchandise from the origin country, through the lading port, to the import country's discharge port. Example: Track expenses for a silk blouse from China, through the lading port, Hong Kong, to the discharge port, L.A.

Zone Level Expenses

Zone level expenses track the costs of bringing merchandise from the import country's discharge port to the purchase order receiving location. Example: Track expenses for a silk blouse from discharge port, Los Angeles., through to the retailer's New York City warehouse and to the retailer's Chicago warehouse. Costs are different based on the final destination (longer truck route, railroad, etc.).

Zones are defined using the Cost Zone dialogue (See Cost Management in RMS). Once the zones are created they are used to define expenses at the supplier level (by zone) for default to items.

Assessments

Assessments are the cost components that make up the total tax and duty charges for an item. Computation formulas and specific tax types contained in the Harmonized Tariff Schedule (HTS) determine most assessments. The harmonized tariff schedule is defined for an import country.

The HTS comprises a hierarchical structure for describing all goods in trade for duty, quota and statistical purposes. The HTS structure is based on the international Harmonized Commodity Description and Coding System (HS), administered by the World Customs Organization in Brussels, Belgium.

There are two components used to track HTS within the system:

- HTS Chapter Tables
- HTS Tariff Item Tables

The HTS Chapter Tables are organized by the first four digits of the Tariff Schedule classification. The HTS Tariff Item Tables are organized around the ten digit tariff item number. The item level tariff codes do vary among GATT (General Agreement on Trades and Tariffs) countries. Each tariff item has various duty rates assigned to it. Classification of goods in this system must be done in accordance with the country specific rules. For example in the U.S. it should be done as per General and Additional U.S. Rules of Interpretation, starting at the four digit heading level to find the most specific provision and then moving to the subordinate categories. Other assessments, such as taxes, fees, countervailing charges, and anti-dumping charges are also assigned at the tariff item level.

RTM also allows the retailer to set up the quota restrictions imposed by the government on an item. The quota is linked to the HTS classification.

Computation Value Bases (CVB) is a mechanism used to create a compound expense. CVB's provide the ability to set up complex expenses that are dependent on other expense components. They also allow expenses to be a certain percentage of a group of other expenses and assessments rather than just of one value.

Purchasing

If RTM is enabled in an RMS instance, then import orders can be created in RMS. The purchase order dialog provides additional import functionalities along with standard PO information, if the import purchase order indicator is checked. Throughout each step of the Import Purchase Order creation, there are several options available to capture additional information specific to ordering imported merchandise.

The menu options within the PO Header Maintenance window are used to attach Shipping, Letter of Credit, Order Dates, Attributes, Required Documents, and Timeline information. The menu options within the PO Item Maintenance window are used to attach HTS, Required Documents, and Timeline information for line items on the PO. The PO Item/Location window provides the option to assign expenses at the Item/Location level. All of the expense components are brought together in one view in the Order Item Expense Maintenance window.

Letter of Credit

A letter of credit (LC) is a bank instrument used by most retailers to finance purchase of imported goods. LCs are a preferred method of international settlement because the conditions of the purchase, such as required documents and special instructions are detailed in the text of the LC and reviewed by the bank for compliance before the release of payment. They provide importers with a secure method to pay for merchandise and vendors with a secure method to receive payment for merchandise. Letters of credit can be created and applied to purchase orders. Activity against the letter of credit can also be tracked. Once goods have shipped and shipping documents are available, the seller presents documents at the desk of the advising bank, and if there are no discrepancies, collects the specified payment amount. The advising bank debits the issuing bank for the amount of the negotiation as well as any charges, and the issuing bank makes payment and notifies the retailer of the amount of the principal draw down and associated charges.

RTM also supports the exchange of bank Letter of Credit (LC) information with a bank in the internationally recognized SWIFT (Society for Worldwide Interbank Financial Telecommunications) format.

Transportation

The transportation functionality in RTM provides a facility to track information from trading partners as merchandise is transported from the manufacturer through customs clearance in the importing country. Information is recorded at various levels such as: This information is recorded at various levels: Vessel/voyage, Bill of Lading (BOL), Container, PO/Item, Commercial invoice. Transportation information is most often received through EDI. Once entered, the information is tracked and edited, as it changes.

The Transportation module is used to track the following information: Shipments, Deliveries, Licenses and Visas, Claims, Missing Documents, Packing, Dates, Timelines, Commercial Invoices, Inland Freight and Totals.

When a transportation record is complete, the record is finalized. Finalized transportation records are used to automatically create customs entries. The goods are then tracked as they move through customs.

Customs Entry

The Customs Entry module manages entries of merchandise for clearance through customs while providing the information required for government documentation and reporting. Government duties, taxes and fees are calculated in accordance with the country of import. Entry information is prepared for transmission to the customs broker for entry submission. The Customs Entry module provides the ability to track the arrival of a shipment at customs, the customs clearance or associated delays, customs exams, and entry and liquidation payment amounts.

When the charges and assessments are complete, the retailer can choose to allocate the costs to the actual landed cost module. When the customs entry is complete, the custom entry record needs to be confirmed. Non-merchandise invoices are created automatically from confirmed customs entries.

Obligations

As an item progresses through the import process, bills are received from various service providers. As commercial invoices are received from trade partners and suppliers, they can be recorded in the obligations module. They are allocated over the shipment, order(s) and item(s) that they cover, and the system proportionally allocates the charges to the line item level. Approving an obligation allocates the costs to the actual landed cost module. Approving an obligation creates a non-merchandise invoice in approved status.

Actual Landed Costs

RTM helps to track the actual landed cost incurred when buying an import item. The module reports variances between estimated and actual landed costs by cost component and shipment. The Actual Landed Cost (ALC) dialogues provide the flexibility to view information about actual and estimated landed costs for any item on a purchase order or invoice obligation. The information can be organized by obligation, obligation cost component, shipment, or location. Duties, fees and taxes are posted directly to the Actual Landed Cost module bypassing the obligations maintenance process. While finalizing the ALC at a purchase order level, the user can also choose whether or not to update weighted average cost (WAC) for each purchase order. For those circumstances where goods may be partially or completely processed through a distribution center to another location prior to assembly and calculation of ALC, it may be more accurate to have WAC be determined by ELC to stand rather than updating it with an ELC/ALC variance.

Overview of Integration of RTM with other Applications

Integration with RMS

Oracle Retail Trade Management (RTM) and RMS share the same database schema. When RTM is enabled in an RMS instance, certain import specific data maintenance is required for country, supplier, partners and items. These are directly updated into the RMS database and subsequently used in RTM.

Integration with Oracle Retail Invoice Matching

RTM, RMS and ReIM share the same database schema.

Customs Entry Record - Finalization of the customs entry record, inserts an approved non-merchandise invoice record into the Invoice Matching staging tables for extract and upload into ReIM.

Obligations Entry - Finalizing the obligations entry also inserts an approved non-merchandise invoice record into the Invoice Matching staging tables for extract and upload into ReIM

Integration with External Partners

RTM automates the international import transaction data. Four components of RTM - customs entry, harmonized tariff schedule, letter of credit, and transportation - have batch-processing modules that facilitate the flow of data between RTM and external applications and files.

Sharing the Letter of Credit Data with the Trading Partners

Letter of credit batch modules process letter of credit applications and amendments to banks, and upload confirmations, drawdown notifications, and related information from banks. Letter of credit downloads and uploads data in an internationally recognized standard format called S.W.I.F.T. (Society for Worldwide Interbank Financial Telecommunications).

The batch program LCADNLD extracts approved letter of credit applications to banks. The LCMT700 Perl script converts the LC applications from an RTM file format to the S.W.I.F.T. (MT 700) format.

The issuing bank sends the retailer a confirmation once it agrees to stand as guarantee for the LC. The LCMT730 Perl script converts letter of credit confirmations from a S.W.I.F.T. format (MT730) to an RTM flat file format. The batch program LCUPLD uploads the converted data from the table to the RTM database tables.

The issuing bank informs the retailer when credit drawdowns are made against the LC or when bank charges the retailer the bank fees. The LCMT798 Perl script converts drawdowns and bank fees data from a S.W.I.F.T. file format to an RTM format. The batch program LCUP798 uploads the converted data from the table to the RTM database tables.

At times after a LC has been issued and confirmed, the retailer may want to make amendments to the LC. An amendment to a LC is valid once all parties involved agree to the new conditions and the agreement has been registered. Retailer initiated amendments are extracted by the batch program LCMDNLD.

Integration with Customs Broker

The batch program CEDNLD (customs entry download) extracts custom entry information from the RTM database to custom brokers. Custom Entry (CE) transactions that are in a Sent status are written into a flat file. One flat file is written per broker. Information contained in the files include: order items, bill of lading / airway bill information, shipment information, container information, license and visa information, broker charges and missing documents information.

The output file could be FTP to the broker. This process has to be handled during implementation and is outside the scope of the RTM.

Upload of HTS data

U.S. Government publishes the HTS data in file format. The RTM batch program HTSUPLD uploads the data from the file into the RTM/RMS database. The program handles both the initial HTS information load as well as mid-year HTS updates that are supplied by the U.S. government. The initial upload is handled by inserting information from the file into the tables. HTS information already available in the tables is handled by adjusting the effective dates of the existing HTS records and inserting a new set of HTS records into the tables. The files provided by the U.S. government include HTS Chapters, HTS classification, HTS Tax, HTS Fee, HTS OGA, Tariff Treatments, Tariff Treatment Exclusions.

The files provided by the U.S. Government should be placed in the proper input directory, so that the batch program HTSUPLD can pick up the file and process the same.

Integration with Supply Chain Partners

As the merchandise moves from manufacturers' warehouse/factory to the discharge port at the import country, the supply chain partners share information with the retailer in a specified file format. The batch program TRANUPLD uploads the information provided in the files into the transportation tables in RTM/RMS database, thus providing online visibility of the merchandise in the supply chain.

The files provided by the supply chain partners should be placed in the proper input directory, so that the batch program HTSUPLD can pick up the file and process the same. In case the files provided by the supply chain partners differ from the expected file format, custom program must be written to convert the data from the original file format to the required file format.

The diagram below illustrates the interaction between RTM and external agencies.
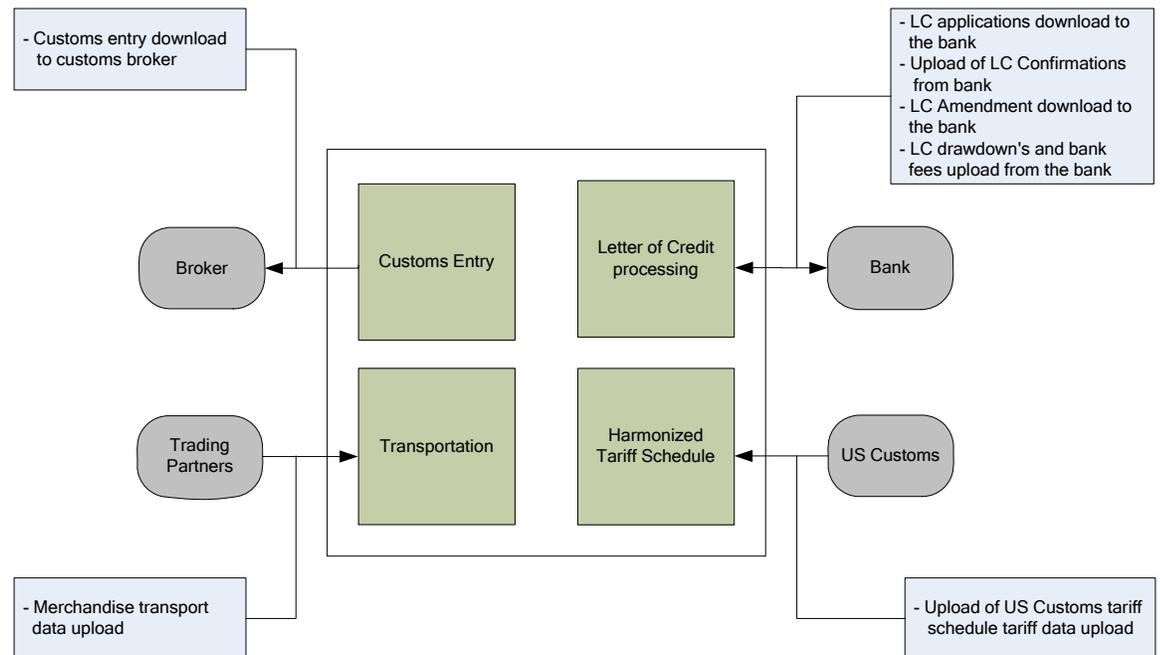
User Setup and Security

As RTM and RMS share the same instance, the application shares the security framework of RMS. The user setup, role assignment and access permission are done in the RMS framework.

For details on the user and security setup in RMS refer to the RMS Users and Security section of this guide.

Simplified RTM Configuration

Simplified RTM is a simplified version of Oracle Retail product suite targeted at mid-tier retailers. The Simplified Oracle Retail Merchandising Operations applications support basic retail processes needed by a mid-tier retailer. Advanced features are turned-off through system parameters, with the goal to reduce implementation complexity and enabling faster implementation and lower total cost of ownership.

If the system option parameter Simplified RTM indicator is enabled, then the following RTM functionality is not available in the application:

- Setting up RTM specific master data such as Freight Type, Freight Size and Standard Carrier Alpha Codes (SCAC)
- Letter of Credit functionality
- Transportation functionality
- Customs Entry functionality
- Obligation Maintenance
- Actual Landed Costs

If both the Simplified RTM indicator and the Import indicator are enabled, then some import related functionality is available in RMS. With this setup, the retailer has the option to setup the HTS data and use it in the purchase order process. The retailer can also choose Letter of Credit as a payment option in the Purchase Order header level, but all other related LC functionality is not available. It is assumed that the retailer is using some other external system for LC processing.

If the import indicator is not enabled then no RTM functionality is available in the application. See the RMS Installation Guide for additional information on setting the value of the system_options table.

Other Features

As RTM is related to import purchases and may use foreign currency, it is important that the current exchange rate between the primary currency and the supplier's currency is maintained in the application. RMS does maintain currency exchange rates and can accept currency updates.

Oracle Retail Sales Audit (ReSA) Overview

Overview of Information that Oracle Retail Sales Audit Maintains

System Options

Oracle Retail Sales Audit (ReSA) contains a set of system options that is different from the RMS system options set. These system options control functionality in ReSA. One of the important system option parameters is the balancing level. The balancing parameter could be Cashier or Register. If the balancing parameter is chosen as 'Cashier' then the totals are balanced for each cashier. If the balancing parameter is chosen as 'Register' then the totals are balanced for each register. Another important option is the unit of work. This determines whether transactions are exported prior to the elimination of all errors for the store/day. Other parameters include defining the escheatment party and the details of the clearinghouse. Certain business rules like duplicate and missing transaction number checks are also controlled through the system options setting.

Foundation Data

ReSA requires a certain amount of foundation data to start working. ReSA shares some of its foundation data with RMS, while the rest of the foundation data needs to be maintained in ReSA.

Foundation data in ReSA includes company/location close dates, location traits, bank setup, tender types, reference maintenance, error codes and store-specific foundation data.

Totals

ReSA allows retailers to define the totals needed for their business. A total in ReSA can be a ReSA calculated value from raw transaction data or a total that comes from the POS via the RTLOG. ReSA calculates a total based on raw transaction data or on existing totals. Totals are used for performing store balancing - over/short analysis within ReSA. Totals are also used to create extract data for external systems such as General Ledger. By assigning an audit rule to the total business validation is built in ReSA. For example, the retailer defines a ReSA calculated total of total cash tendered at a register for a store/day. He defines another total, a POS-declared total, for the total cash declared at the register for a store/day. He defines an audit rule to compare both the totals for over/short analysis. Totals are used to export consolidated data to an external system. So, while defining a total, the retailer defines the systems to which the total is exported.

Audit Rules

Rules are used in ReSA to perform custom data validation against transactions and totals. Audit rules run on POS transaction data and totals during the automated and interactive audit process. When the rule is broken an exception error is thrown that must be edited or overwritten by the auditor. ReSA comes with some standard pre-configured rules and also provides flexibility to the retailers to define rules needed for their business.

Error Codes

Error codes provide information to the auditor about the type of error and the remedial action needed to solve the error, thus increasing the effectiveness of the audit process. When setting up audit rules for automated audit process, an error code is associated to the rule. The error code contains an explanation of why the rule failed, as well as a recommended solution. The error code also contains security settings to determine if the error can be overridden at the store or headquarters. When a rule fails, the error code is available to the auditor in the error list. Additionally the specific location where the error has occurred is available in the transaction details, thus helping the auditor to quickly understand and correct the error.

Automatic Audit Process

Automatic auditing is done in ReSA using batch programs. The goal of the automatic audit process is to accept transaction data from point-of-sale (POS) applications and move the data through a series of processes that culminate in "clean" data. ReSA uses several batch-processing modules to perform the following activities:

- Import POS transaction data from RTLOGS
- Perform initial validation of data during upload of data from RTLOGs to ReSA database
- Produce totals using user-defined totaling calculation rules that are user reviewable during the interactive audit
- Validate transaction and total data with user-defined audit rules and generate errors whenever data does not meet the criteria. The user reviews these errors during the interactive audit
- Create and export files of clean data in formats suitable for transfer to other applications
- Update the ReSA database with adjustments received from external systems on previously exported data

Interactive Audit Process

Auditors use the interactive audit process to view and correct errors. This process is done after the automated audit is completed. This process allows the auditors to view errors at summary or detail level, fix or override errors, update the totals and close the store/day. In addition, it is also possible to review and edit data of missing transactions or transactions that have passed automated audit, add transactions and delete invalid or missing transactions.

Summary Views

ReSA provides summary views for the auditor to identify and fix the problem areas quickly. A store/day summary view form contains the following information about a store/day: the audit status, the data status, the number of transactions, the number of errors and the over/short amount. The auditor has access other options from the store/day summary form such as: the error list, the balancing level summary, over/short, miscellaneous totals, missing transactions and the import/export log.

ReSA provides two other summary views: the Tender Summary and the Item Summary.

Single or Multi –Level Auditing

It is common to have one of the following workflows for the sale audit:

- Single level audit, with all audits being performed at the head office
- Multi-level audit process in which some audit occurs at the store and the final auditing happens at the head office.

For the multi-level audit it is important that the store auditors must perform their audits before the head – quarter's auditors, making workflow management very important.

Single or multi-level auditing in ReSA is achieved through proper assignment of employees to stores and headquarters.

Automated Clearing House (ACH) Processing

Automated Clearing House (ACH) is a U.S. based banking network used to electronically transfer funds. Retailers use ACH to enable them to have access to funds before the funds have been physically deposited in the bank. This is done by estimating the following day's bank deposit and sending this amount to the consolidating bank via the ACH network. In this way, the cash to be received from the stores is hedged.

Escheatment Processing

Escheatment is the process of forwarding monies of outstanding, non-expiring vouchers to the proper government authorities (state or country) after a defined period of time from the date of issuance. Some government authorities require that unredeemed vouchers be "escheated" after a specific period of time has passed. Escheatment is the process of forwarding monies of outstanding, non-expiring vouchers to the proper government authorities (state or country) after a defined period of time from the date of issuance. When a voucher is escheated, an invoice is generated that initiates payment of the escheated voucher amount to the government authority. The government authorities then attempt to locate the consumers owed the monies.

To accommodate Escheatment, a new total should be added to Sales Audit to create escheatment totals. ReSA automatically totals sales transactions based on calculation definitions setup for the total.

Audit Trail

The audit trail functionality provides the store and headquarters employees with the capability of tracking all changes to transactions and totals. ReSA maintains versions of all modified transactions thus enabling easy tracking of changes.

Totals for General Ledger that are impacted by a revised transaction are reversed and both the reversal and the new total are extracted for the General Ledger.

Reporting

ReSA provides some basic online reporting that allows users to view sales audit data that is in the system. The following reports are available in ReSA:

- Flash Totals Report from Store Day Summary
- Flash Sales Report
- Flash Sales Prior year Comparison report
- Credit Card Summary Report
- Voucher Activity Report

Overview of Integration of ReSA with other Oracle Retail Merchandising Operations Applications

Oracle Retail Sales Audit interfaces with the following Oracle products:

- Oracle Retail Merchandise System
- Oracle Retail Point of Service System
- Oracle Retail Data Warehouse
- Oracle Retail Invoice Matching System
- Oracle Financials General Ledger
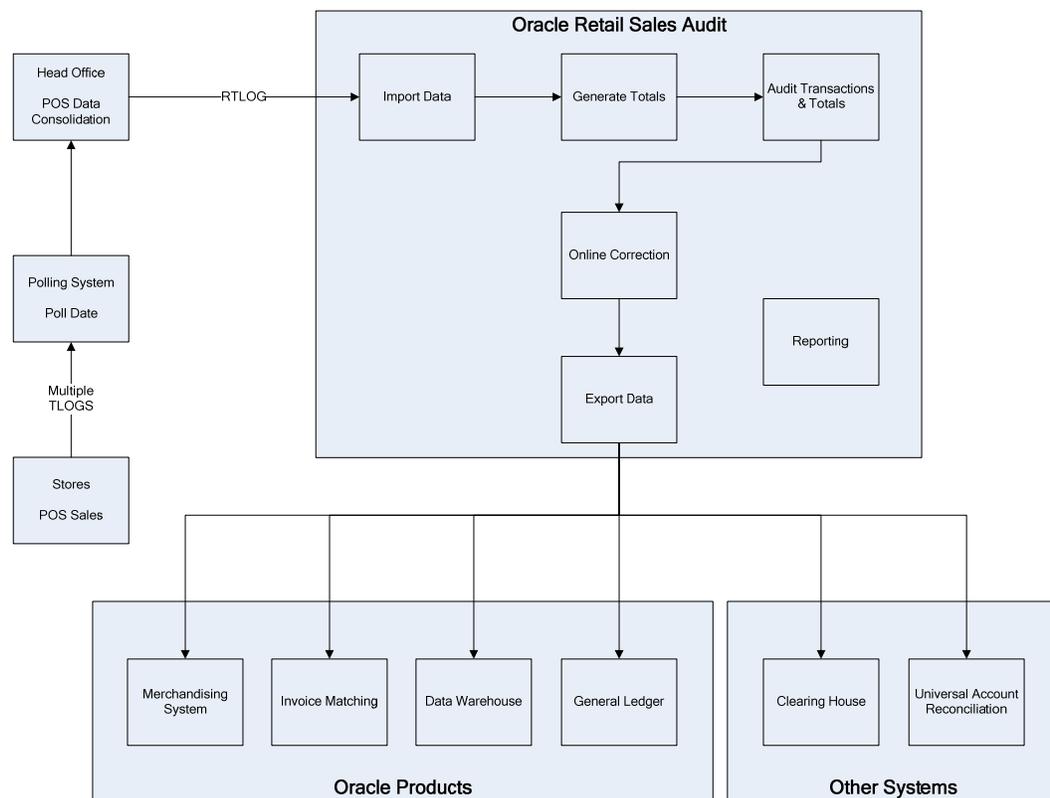
Oracle Retail Sales Audit also interfaces with the following third party applications:

- Automated Clearing House
- Universal Account Reconciliation System

Integration with Oracle Retail Merchandising System

ReSA and RMS share the same database. ReSA shares some of its master data with RMS. Foundation data such as items, stores, company/location close dates, location traits, bank setup and tender types are maintained in RMS and used in ReSA.

Current reference data is retrieved from RMS into ReSA by the batch program SAGETREF. The data is extracted into multiple data files. The data in the files are used by the batch program SAIMPLOG as reference data for doing validation checks on the POS transaction data during the data upload from POS to ReSA. Having the reference in data file formats increases the performance of the SAIMPLOG process. SAGETREF generates the following reference files: Items, Wastage, Sub-transaction level items, Primary variant relationships, Variable weight PLU, Store business day, Code types,

Error codes, Credit card validation, Store POS, Tender type, Merchant code types, Partner vendors, Supplier vendors, Employee ids, Banner ids.

All clean and audited sales and returns data is extracted from ReSA into a POSU file by the batch program SAEXPRMS. All corrected sale and return transactions that do not have RMS errors are extracted into the file. The sales audit system options parameter work unit controls the export of data into file in case of presence of RMS errors in the POS transaction data. The batch program POSUPLD loads the data from the POSU file into the RMS tables.

Integration with Point-of-Service

Sales, returns and other POS transaction data is loaded into RESA from the point-of-service application. The point-of-service should provide a standard RTLOG file. The point-of-service data is loaded into ReSA either in trickle mode or once a day. If the data is uploaded in the trickle mode, then corporate inventory reflects a more accurate intra-day stock position. The data from the RTLOG is loaded into ReSA using the batch program SAIMPLOG for end of day processing or SAIMPTLOGI for trickle processing. If trickle processing is used the final RTLOG for the day must include a count of all RTLOG files for the store/day.

Integration with Oracle Retail Data Warehouse

The batch program SAEXPRDW collates all corrected sale and return transactions that do not have any Oracle Retail Data Warehouse (RDW) errors from the ReSA database tables for transmission to the RDW. The data is sent at the store day level. SAEXPRDW provides four output files:

- RDWT for sales transactions
- RDWF for tender transactions
- RDWC for cashier over/short
- RDWS for other user totals

It is the responsibility of the implementation team to ensure that the output files are transferred to the input directory of RDW.

Integration with Oracle Retail Invoice Matching

In the normal course of business, payments are made to vendors at the store level. Payments for merchandise purchases done at store level are booked against a corresponding merchandise invoice. Payments of non-merchandise purchases or miscellaneous services availed at the store are booked against a corresponding non-merchandise invoice. These transactions are passed from the POS to ReSA as specially designated PAID OUT transactions (sub-transaction type of EV – Expense Vendor or MV – Merchandise Vendor). All these invoices are assumed paid. The batch program SAEXPIM transfers the PAID OUT type of transactions to the Invoice Matching staging tables.

The batch program SAEXPIM is also used for escheatment processing. Unclaimed monies of outstanding, non-expiring vouchers are totaled after a defined period of time from the date of issuance of the voucher and posted to the Invoice Matching staging tables as a Non-merchandise Invoice by SAEXPIM. The unclaimed amount is paid out as income to the issuing Retailer or in some U.S. States, it is paid out to the State (based on configuration). ReSA determines who receives this income and accordingly posts a non-merchandise invoice for the partner. These invoices are assumed not paid.

The batch job EDIDLINV is used to extract the invoices from the Invoice Matching staging tables and load as EDI invoices to ReIM.

Integration with Oracle General Ledger

The batch program SAEXPGL transfers the sales data from ReSA into the financial staging tables in RMS. This batch program executes only if the external financial system is set to 'O' for "Other" – either Oracle Financials or some other external Financial application based on system options configuration.

SAEXPGL directly inserts the data into financial stage tables of RMS.

Integration with Automated Clearing House

ReSA determines the estimated bank deposit for each store/day. The batch program SAEXPACH posts the store/day deposits into a database table and creates the standard ACH format file. The output file is sent to a Clearing House. The output file conforms to the requirements imposed by the National Automated Clearing House Association (NACHA). The nature of the ACH process is such that as much money as possible must be sent as soon as possible to the consolidating bank. Any adjustments to the amount sent can be made front-end. This batch assumes that there is only one total to be exported for ACH per store/day.

Integration with Universal Account Reconciliation Solution

The batch program SAEXPUAR extracts specified TOTALS to a flat file that is interfaced to an account reconciliation application such as the *Driscoll UAR* application. For each store day, all specified totals are posted to their appropriate output files. All the stores and totals with usage type starting with 'UAR' are exported.

User Setup and Security

Access to form and records in ReSA is controlled at two levels: RMS security and ReSA specific security.

Since ReSA and RMS share the same instance, ReSA utilizes the basic security functionality defined within RMS. ReSA users must have an Oracle User ID and must have the appropriate product and location security defined within RMS. In addition to the general RMS security, additional security must be defined for ReSA users.

Security setting at ReSA level includes classifying the employee as a store employee or a headquarters employee and then defining some field level access. These security settings affect the forms and fields to which an auditor is given access, and at what point in the auditing process the access is granted.

The employee maintenance form in ReSA is used to define the employee type. The employee is also assigned the stores for which he can perform an audit process. Note that the employee still has the option of viewing all stores to which they have been assigned RMS security. Additionally the POS Ids can be assigned to the user on the form. Based on a ReSA system option setting, the user id of the cashier is validated during the RTLOG upload and the employee's POS ID is used for cashier validation. Stores are assigned to headquarter employees through the use of location traits.

Field level access provides the setup of security for nine specially-designated fields based on the Oracle Role. Since both store and headquarters use the same forms and have access to the same data, additional controls for data security are needed. This additional security is provided by the field level access control in select forms.

5

Oracle Retail Allocation Overview

Overview of Information that Oracle Retail Allocation Uses and Maintains

Item Sources – Oracle Retail Allocation provides the client a number of sources from which to allocate product from. These sources include:

- Purchase Orders
- Advanced Shipping Notices (ASN)
- Transfers
- Bill of Lading (BOL)
- Warehouse inventory

Clients have more access and control to existing transactions as a result of the different item sources which results in increased supply chain efficiencies.

Advanced Need Determining Calculations – Allocations are calculated in real time by advanced internal algorithms that calculate store need based on the rule parameters established by the client and the current perpetual inventory and sales for the item(s) being allocated.

"What if" Allocations – "What if" allocations are similar to regular allocations with the exception being an infinite amount of product available to allocate. This allows a client to determine the true need for the locations being allocated to, and if they choose, create a purchase based on the allocation to fulfill that need.

Allocation Templates – Oracle Retail Allocation provides clients with the ability to create standard templates to apply to allocations to save time and produce consistent results. Two types of templates can be created, location template or rule template.

Rules – Oracle Retail Allocation requires the selection of a rule for the calculation of an allocation. The rule defines where data used in the calculation of the allocation comes from and other parameters that are used in the calculation.

Split Allocations – Oracle Retail Allocation allows the user to react to changes (such as: short ships, delays, and cancelled product) by splitting an allocation. Using this functionality a user is able to split one or many items off from an existing allocation and onto their own allocation.

Pre-pack Allocations – A pre-pack is a package containing multiple items for distribution. Oracle Retail Allocation has the capability to define optimum pre-packs for clients in addition to allocating them. This function is used to minimize shipping and handling costs.

Oracle Retail Allocation Integration with other Applications

Oracle Retail Allocation uses a Java architecture built on a layered model. Layers of the application communicate with one another through an established hierarchy and are only able to communicate with neighboring layers. The application is divided into a presentation layer, a middle tier consisting of services and business objects, and a database access/driver layer.

The segregation of layers has the following advantages, among others:

- The separation of presentation, business logic, and data makes the software cleaner, more maintainable, and easier to modify.
- The look and feel of the application is updated more easily because the GUI is not tightly coupled to the back end.
- A layered architecture has become an industry standard.
- Portions of the data access layer (DAL) can be radically changed without effecting business logic or user interface code.
- The application takes advantage of Java database connectivity (JDBC), minimizing the number of interface points that must be maintained.
- Market-proven and industry-standard technology is utilized (for example, JSPs, JDBC, and so on).

The diagram below illustrates the Allocation n-tier architecture:

--------------- Denotes separation of tier
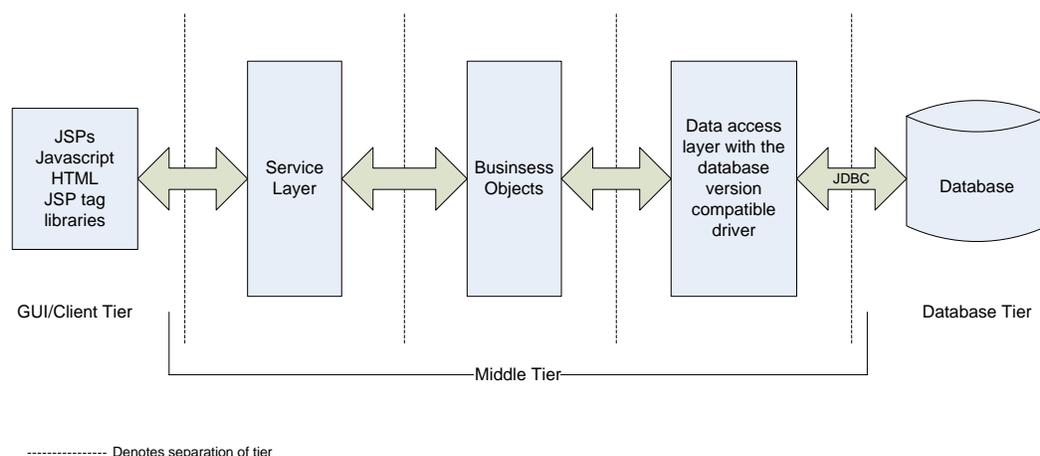
RMS owns virtually all of the information that Oracle Retail Allocation needs to operate and the information that Oracle Retail Allocation provides is of primary interest/use for RMS. As a result Oracle Retail Allocation has limited interaction with other Oracle Retail Merchandising Operations Management applications. For Oracle Retail Merchandising Operations Management applications that Oracle Retail Allocation does interact with it is managed via 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.

The diagram below outlines where Oracle Retail Allocation lives within the merchandising footprint.
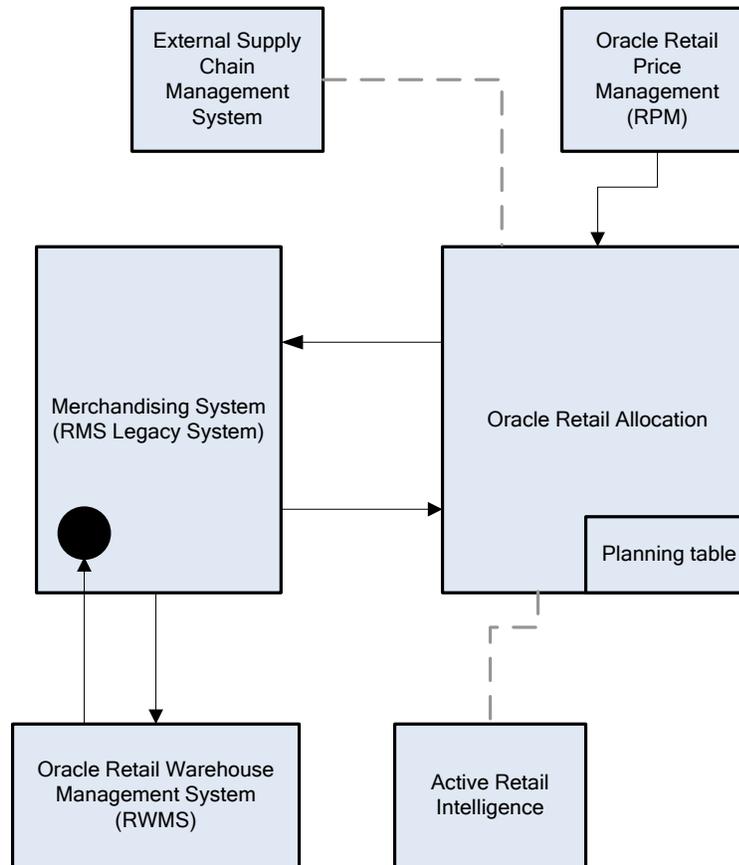
Note: This symbol denotes tables held on the merchandising table. Oracle Retail Allocation pulls the data from these merchandising tables through the use of the JDBC connection.

Oracle Retail Allocation and RMS

RMS provides Oracle Retail Allocation with:

- **Foundation Data** - This information is essential to all areas of Oracle Retail Allocation including valid locations to allocate to and from, location groupings, valid merchandise hierarchies to allocate within, etc.

- **Item** - Allocations are generated at the item location level so it is necessary that the Allocation application understand what items and item/locations are eligible in the system.

- **Purchase Order** – One of the sources from which a user allocates from is Purchase Orders. Oracle Retail Allocation relies on RMS to provide Purchase Order information.

- **Transfer** – One of the sources from which a user allocates from is Transfers. Oracle Retail Allocation relies on RMS to provide Transfer information.

- **BOL** – One of the sources from which a user allocates from is a BOL. Oracle Retail Allocation relies on RMS to provide BOL information.

- **ASN** – One of the sources from which a user allocates from is an ASN. Oracle Retail Allocation relies on RMS to provide ASN information.

- **Inventory** – In order to determine the correct need at an item location level before performing an allocation the application needs visibility to the current on-hand inventory at each location being allocated to. Oracle Retail Allocation relies on RMS to provide Inventory information at the item/location level.
- **Shipping Information** – One of the sources which the Oracle Retail Allocation application allocates from is an Advance Shipment Notice (ASN). Oracle Retail Allocation relies on RMS to provide ASN information.
- **Sales Information** - Oracle Retail Allocation uses historical sales, forecast sales, and plan in order to determine the need at an item/location level for an allocation. Oracle Retail Allocation interfaces this information in from external planning system to an allocation table.

Oracle Retail Allocation provides RMS with:

- **Allocations** - Once an allocation has been moved to Approved or Reserved status the allocation is written to RMS tables to give visibility to the allocation results.
- **Purchase Orders created by "What-If" process in Allocation** – If the user selects the "What-if" option when creating an allocation the allocation is created based on current need and then have RMS build a Purchase Order from the allocation to fulfill the need. Oracle Retail Allocation uses an RMS API to build the purchase order in RMS.

Oracle Retail Allocation and RTM

No information is exchanged.

Oracle Retail Allocation and ReSA

No information is exchanged

Oracle Retail Allocation and RPM

RPM provides Oracle Retail Allocation with:

Future Retails – When the option is enabled Oracle Retail Allocation has the ability to get future retail prices for items being allocated. This allows a client to calculate the total retail of an allocation based off from the pricing when an allocation is physically executed which may vary from the time the allocation is created. Oracle Retail Allocation uses an RPM API call to access this information from RPM tables.

Promotions – Users in Oracle Retail Allocation can select a promotion from RPM to associate an allocation with.

Oracle Retail Allocation provides RPM with:

No Information

Oracle Retail Allocation and ReIM

No information is shared between these applications

Oracle Retail Allocation and ARI

ARI is a monitoring system that interacts with any applications database (including Oracle Retail Allocation). As such it does not "use" any information from Oracle Retail

Allocation; rather it monitors the Oracle Retail Allocation database for events defined by a client and notifies the client when said events occur.

Oracle Retail Allocation User's and Security

Users in Oracle Retail Allocation will need to be setup by a DBA. They are managed via a user table on the Oracle Retail Allocation database called ALC_USERS. When a user logs onto Oracle Retail Allocation their user name and password are authenticated against the ALC_USERS table. If the user/password exists, they are granted access. Valid users in Oracle Retail Allocation have access to all functionality in Oracle Retail Allocation

Once a user has been authenticated to use the Oracle Retail Allocation application the system provides a single tier security structure that provides data level security based on the merchandise hierarchy.

Data – Users are authorized to access different information within the Oracle Retail Allocation system based on which departments they are authorized for.

Users in Oracle Retail Allocation are setup with data level security at the department level in Oracle Retail Allocation. A user has access to one or many departments in the application and this authorization determines the items that a user is able to allocate per the merchandise hierarchy. Users are setup with data level security by having a system administrator update the ALC_USER_DEPTS table with the departments they are authorized to perform allocations for.

Internationalization

The technical infrastructure of Oracle Retail Allocation supports languages other than English. Features of Oracle Retail Allocation that are specific to one language or locale (such as text, date formatting, etc) have been placed in files external to the application. The content of these files is interface related, as distinct from executable code. The text in multiple allocation.properties files is translated so that the interface functions in local settings. These files comprise the interface layer.

6

Oracle Retail Invoice Matching (ReIM) Overview

Overview of Information that Oracle Retail Invoice Matching Uses and Maintains

Multi-dimensional matching - Utilizes complex matching logic designed to maximize match rates and processing productivity for both invoice and credit note matching.

Discrepancy routing - Identifies cost and quantity discrepancies when a match has not occurred after a user-specified period of time and automatically routes discrepancies to user groups capable of efficient and effective disposition.

Resolution dialog - Offers a powerful, streamlined approach to handling invoice discrepancies where reviewers can disposition a discrepancy based on a set of user-defined reason codes.

Self-billing and deals bill-back integration - Provides robust integration with the Oracle Retail Merchandising System that supports supplier billing for RTVs, rebates and other deals, consignments, direct store delivery, evaluated receipts, and other non-merchandise billings from obligations and customs entry.

Receiver adjustments integration - Provides direct updates of receiver cost and quantity adjustments initiated from the matching/resolution process to inventory valuation and positions in the RMS.

Best terms date - Uses payment terms rankings (predetermined by the user) to identify the invoice or purchase order term best supporting the retailer's cash management objectives. Payment terms and terms date information is exported to the retailer's accounts payable solution to support payment of the invoice.

Debit reversals - Allows the user to efficiently convert a supplier-disputed debit memo into an editable credit memo with supplier comments for resolution—allowing for flexible handling through the routing process or central processing.

Matching tolerances - Offers the flexibility to set up tolerances by monetary range, nominal amount, or percent. Separate tolerances can be applied for quantity and cost and for discrepancies in either the retailer's or its supplier's favor. Tolerances are set at supplier, merchandise department, and system levels

Invoice Matching Integration with other Applications

Invoice Matching uses a Java architecture built on a layering model. Layers of the application communicate with one another through an established hierarchy and are only able to communicate with neighboring layers. The application is divided into a presentation layer, a middle tier consisting of services and business objects, and a database access/driver layer.

The segregation of layers has the following advantages, among others:

- The separation of presentation, business logic, and data makes the software cleaner, more maintainable, and easier to modify.
- The look and feel of the application is more easily updated because the GUI is not tightly coupled to the back end.

- A layered architecture has become an industry standard.
- Portions of the data access layer (DAL) can be radically changed without effecting business logic or user interface code.
- The application takes advantage of Java database connectivity (JDBC), minimizing the number of interface points that must be maintained.
- Market-proven and industry-standard technology is utilized (for example, JSPs, JDBC, and so on).

The diagram below illustrates the Invoice Matching architecture:

Web Browser (Internet Explorer)

Presentation Layer	JSP/Struts ← Tags
Middle Tier	Services Responsible for Business Logic/ Business Objects
Data Access Layer	Merchandising system data access — Custom RMS Driver — RMS 12.x Driver — ReIM Data Access — Technical Services
Database	DB Custom Tables — RMS 12.x Table — DB ReIM tables
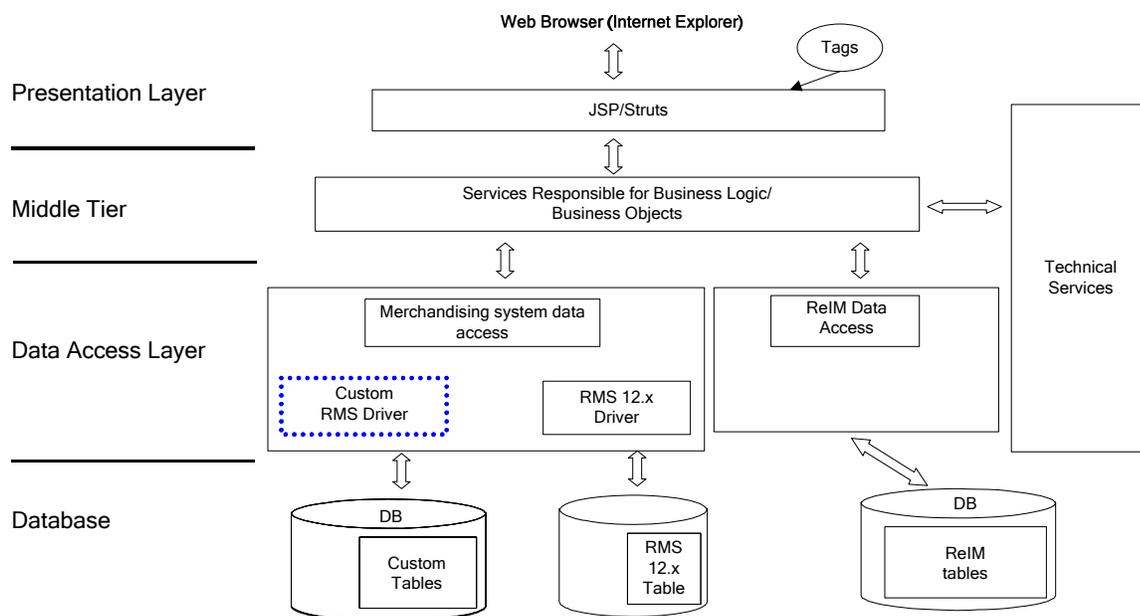
The Invoice Matching application's primary purpose is to match invoices so they can be exported to Accounts Payable (AP) to be paid. Invoice Matching has limited interaction with the Oracle Retail Merchandising Operations Management applications with the exception of RMS. RMS is the owner of the information that Invoice Matching needs to match Invoices it receives.

Information from the Invoice Matching application is shared with Oracle Retail Merchandising Operations Management applications through direct reads from Oracle Retail Merchandising Operations Management application tables, calls to Oracle Retail Merchandising Operations Management application packages, Invoice Matching packages based on Oracle Retail Merchandising Operations Management application tables, and batch processes.

The diagram below illustrates how Invoice Matching interacts within a client's merchandising systems.
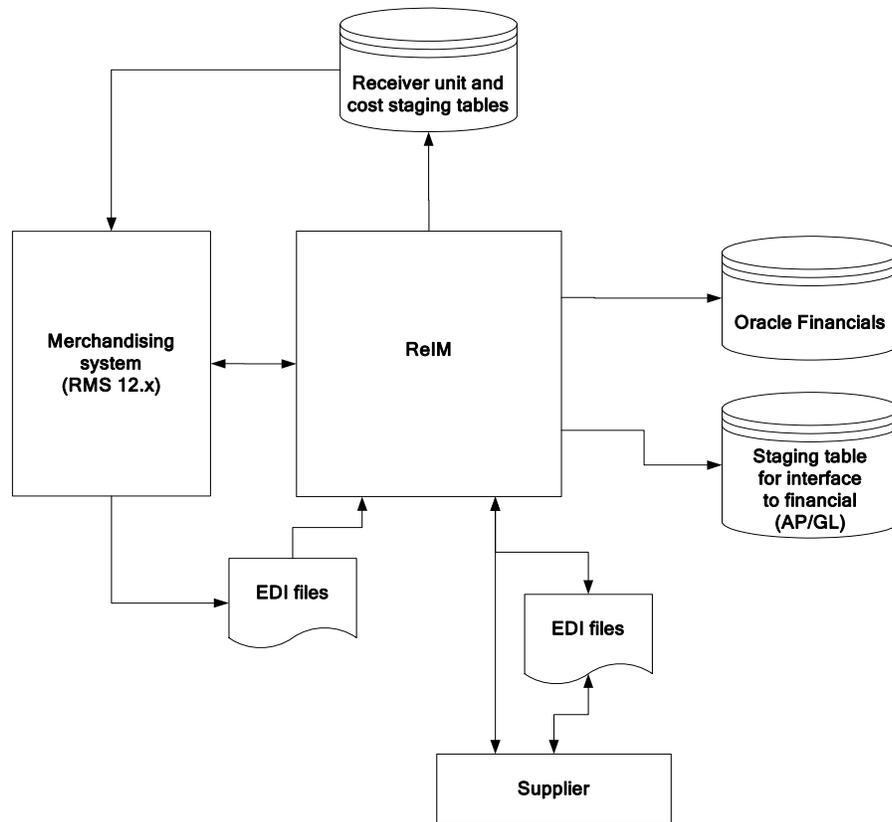
Invoice Matching and RMS

RMS provides Invoice Matching with:

- **Foundation Data** - This information is essential to all parts of invoice matching including valid locations for invoices to be executed at, valid suppliers to receive invoices from, supplier addresses to send credits and debits based on invoice matching results, and more.

- **Item** - This information is essential to the invoice matching process as item information ensures that invoices being received are valid for the business. For example an item received on an invoice is carried by the client, is supplied by the supplier who sent the invoice, and is carried in the locations for which the item was received.

- **Purchase Orders** – Purchase Orders are used by Invoice Matching to facilitate the invoice matching process which is performed at the purchase order location level.

- **Shipments** – Shipment information is used by Invoice Matching to determine if a PO has been received yet which affects the matching algorithm used by the AutoMatch batch program in Invoice Match.

- **Deals and Rebate** – Invoice Matching creates credit memos, debit memos, and credit requests based on deal and rebate information in RMS for processing by the financial (AP) system. This is performed by the ComplexDealUpload and FixedDealUpload batch processes that read from RMS staging tables.

- **Staged Accounts Payable transactions**: Accounts payable documents created in RMS for consignment invoices, Obligations invoices, custome entries invoices, payment transactions sent via ReSA, and Return to Vendor chargebacks (either debit

memos or credit note requests) are staged to Invoice Matching staging tables in RMS and extracted using the batch EDIDLINV to be loaded as EDI documents into Invoice Matching.

Invoice Matching provides RMS/RTM/ReSA with:

- **Invoice Matching results for shipments** – Shipment records are updated with the invoice matching results from the invoice match process (this involves updating the match status and quantity matched of the shipments in question). The matching process is handled by the AutoMatch batch process in Invoice Matching which attempts to match all invoices in ready-to-match, unresolved, or multi-unresolved status.

- **Receiver Cost Adjustments** – When invoice matching discrepancies are resolved via a Receiver Cost Adjustment, information is placed on a staging table in Invoice Matching for export to RMS to perform the cost adjustment on the affected purchase order and shipment. The ReasonCodeActionRollup batch in Invoice Matching is responsible for inserting these records on a staging table in Invoice Matching for export to RMS. A trigger on the Invoice Matching staging table completes the transaction by updating RMS with the cost adjustment information.

- **Receiver Unit Adjustments** - When invoice matching discrepancies are resolved via a Receiver Unit Adjustment information is placed on a staging table in Invoice Matching for export to RMS to perform the unit adjustment on the affected order and shipment. The ReasonCodeActionRollup batch in Invoice Matching is responsible for inserting these records on a staging table in Invoice Matching for export to RMS. A trigger on the Invoice Matching staging table completes the transaction by updating RMS with the unit adjustment information.

- **Closing unmatched shipments** – Invoice matching closes the invoice matching status for shipments in RMS after a set period of time (defined by the client in system options). This updates the invoice matching status of the shipment on the shipment table in RMS. This process is managed by the ReceiptWriteOff batch program.

Invoice Matching and RTM

RTM provides Invoice Matching with:

- **Finalized Customs Entry** – When Customs Entries are confirmed in RTM a non-merchandise invoice is automatically created in Invoice Matching staging tables.

- **Approved Obligations** – When an Obligation is approved in RTM a non-merchandise invoice is automatically created in Invoice Matching staging tables.

Invoice Matching provides RTM with:

No information is provided.

Invoice Matching and ReSA

ReSA provides Invoice Matching with:

- **Store Level Purchasing -** Payments for merchandise purchases done at store level are booked against a corresponding merchandise invoice. Payments of non-merchandise purchases or miscellaneous services availed at the store are booked against a corresponding non-merchandise invoice. These transactions are passed from the POS to ReSA as specially-designated PAID OUT transactions. All these invoices are assumed paid. The batch program SAEXPIM transfers the specially-designated PAID

OUT type of transactions to the Invoice Matching staging tables for extract to the Invoice Matching application.

- **Escheatment Processing -** Unclaimed monies of outstanding, non-expiring vouchers are totaled after a defined period of time from the date of issuance of the voucher and posted to Invoice Matching staging tables as a Non-merchandise Invoice by SAEXPIM. The unclaimed amount is paid out as income to the issuing Retailer or in some U.S. States, it is paid out to the State. ReSA determines who receives this income and accordingly posts a non-merchandise invoice for the partner. These invoices are assumed not paid.

Invoice Matching provides ReSA with:

No information is provided

Invoice Matching and RPM

Information is not shared between these applications.

Invoice Matching and Oracle Retail Allocation

Information is not shared between these applications.

Invoice Matching and ARI

ARI is a monitoring system that interacts with any applications database (including Invoice Matching). As such it does not "use" any information from Invoice Matching; rather it monitors the Invoice Matching database for events defined by a client and notifies the client when said events occur.

Invoice Matching and Financial Systems

Invoice Match exports data to financial staging tables via the ResolutionPosting batch program. If integrated with Oracle Financials, there is a standard interface of data to Accounts Payable and General Ledger. However, if the client is using any other financials system, the client must create their own interface to deliver this information to the applicable financial system.

Invoice Matching and External Suppliers

Invoice Matching gets invoices from external suppliers in one of two ways, EDI or hardcopy. When EDI is used the EdiUpload batch program is responsible for uploading the invoice details from the vendor using a standardized file format. When a hardcopy is used, the client needs to manually enter the invoice in the system before matching can proceed.

Notification to suppliers of chargebacks and requests for credit notes is provided in a flat file extracted by EdiDownload batch process.

Invoice Matching Users and Security

Users in Invoice Matching can be setup either using LDAP or the database via the IM_USER_AUTHORIZATION table. If clients have existing applications that use LDAP Invoice Matching can be configured to point towards the clients LDAP server to eliminate the redundancy of creating multiple user id's for the same user. Clients need to determine which method works best for them during implementation.

Users logging into Invoice Matching are validated against LDAP or the database to authenticate the user prior to beginning an application session. A system administrator is responsible for setting up new users in the system (LDAP or the database).

Once a user has been authenticated Invoice Matching provides a two-tier security structure to limit the functions and information that they are authorized to use.

Application Level Security

This provides a way for a client to limit the functionality that users have access to in the Invoice Matching application. Invoice Matching provides a client with the ability to grant users with one of three levels of access to various functionalities:

- Edit – This provides a user group with total access to the functionality in question.

- View – This provides a user group with the ability to access a functional area, however the user group is not able to interact with information in the functional area (only view it).

- No Access – This prevents a user group from accessing the functional area in question.

Application Level Security for Invoice Matching is maintained on the IM_BUSINESS_ROLES table and is associated to users via the IM_BUSINESS_ROLE_MEMBER table.

Data Level Security

This provides a way for a client to limit the information that users have access to in the Invoice Matching application. Invoice Matching allows a client to limit a user's access for three types of information:

- Dept/Class – Limits a user group to performing Invoice Matching functions for specific department/classes. This information is maintained on the IM_BUSINESS_ROLES_DEPT table.

- Location – Limits a user group to performing Invoice Matching functions for specific locations. This information is maintained on the IM_BUSINESS_ROLES_LOC.

- Reason Codes – Limits a user group to specific reason codes that they are allowed to use. This information is maintained on the IM_BUSINESS_ROLES_REASON_CODES table.

To implement the security structure offered by Invoice Matching the concept of business roles is used. Business roles are unique to each client and need to be defined during the implementation effort. Business roles should accurately represent the different types of users that access the Invoice Matching application and the types of functions and information that those roles need to access.

The Invoice Matching application allows a client to setup these business roles through the application by using the User Group dialogue. The Invoice Matching application also allows a client to associate users to a respective business role (User Group) via the User Group dialogue. This results in the user inheriting the security characteristics of that user group. It is important to note that the relationship between a user and a User Group is 1:1.

Internationalization

The Invoice Matching application is internationalized to support multiple languages. The application can be set up to have the language displayed consistent for all users, or different depending on the user logged into the application.

Language configuration is achieved via the reim.properties file which points the application to the location of the user's properties file based on the specified locale. The properties files, ReIMResources and ReIMMessages, include the translations for all user interface strings.

See the Invoice Matching Operations Guide for additional information regarding internationalization.

7

Oracle Retail Price Management (RPM) Overview

Overview of Information that Oracle Retail Price Management Uses and Maintains

Zone Structures

Zone structures in RPM allow a retailer to define groupings of locations for pricing purposes and eliminate the need to manage pricing at a location level. At the highest level, these groupings are divided into categories called 'zone groups'. While these zone groups may be flexibly defined, they are primarily defined by their pricing scheme. The three types of zone groups in RPM are regular zone groups, clearance zone groups, and promotion zone groups. The default zone group for a new item is determined by the merchandise hierarchy defaults defined in RPM.

Codes

Market Basket Codes

A market basket code is a mechanism for grouping items within a hierarchy level in order to apply similar pricing rules (margin target or competitiveness). Market basket codes cannot vary across locations in a zone. RPM thus assigns and stores market basket codes against an item/zone. An RPM user can set up the following two market basket codes per item/zone:

- One used in conjunction with the competitive pricing strategy (competitive market basket code).
- One used in conjunction with the margin and maintain margin pricing strategy (margin market basket code).

When the merchandise extract batch process runs, the program identifies the pricing strategy being executed and uses the items extracted, the zone information on the strategy, and the type of strategy to determine what market basket codes to use when proposing retails.

Link Codes

Link codes are used to associate items to each other at a location and price them exactly the same. RPM users set up and maintain item/location link code assignments.

Price Changes

Price changes are the pricing events in RPM that affect the regular retail price. When a price change is created, you are specifying the following:

- What item is receiving the price change
- Where the price change is occurring
- How the price of the item is changing
- When the price change will take effect
- Reason for the price change

When price changes are approved in RPM, they are made available to RMS for ticketing purposes.

Promotions

Promotions are events in RPM that discount the price of an item for a defined amount of time. Promotions are set up to apply to the regular retail price, the clearance retail price, or both, and when the promotion ends, the price reverts back to the retail price that existed prior to the promotion. When a promotion is entered in RPM, the retailer specifies the duration of the promotional price, what kind of promotion takes effect, and to which item(s)/location(s) the promotional price applies.

Clearances

Clearances in RPM are defined as a markdown or a series of markdowns designed to increase demand and therefore move inventory out of a store. Subsequent clearances always result in the price of an item decreasing. When a clearance is created, the retailer is specifying the item(s) and locations where the clearance is in effect and the discount or set price for the markdown.

Promotion Constraints

Promotion Constraints warn users when a price change or promotion is occurring within a set number of days of another approved price change or promotion. The number of days is determined by a promotion constraint variable that is stored at the subclass/location level.

When the user runs conflict checking on a price change record, promotion record, or worksheet status record, promotion constraint checks are run. If a promotion constraint is violated, the user has the option to either ignore the constraint, or have the system help suggest a date that does not violate the constraint (price change and worksheet dialogs only).

Pricing Strategies

The pricing strategies front end allows a retailer to define how item retails are proposed when pricing worksheets are generated. The strategies are defined at department, class, or subclass in order to represent which items are affected.

RPM offers five types of Pricing Strategies –

- **Area Differential Pricing –** This allows clients to set prices for items at a particular zone or zone group differently than another zone or zone group. The price differential is based on the rules a retailer defines. Area differentials are used when a

retailer creates a price change to ensure consistent pricing. Differential pricing cannot be applied to other pricing events, such as clearances or promotions.

- **Clearance Pricing –** This allows clients to define the method used to markdown items.
- **Competitive Pricing –** This allows a client to define a pricing strategy for their items based on their primary competitor's prices. All locations in a competitive pricing strategy must use the same currency.
- **Margin Pricing –** This allows a client to define a pricing strategy for items based on margin targets.
- **Maintain Margin Pricing –** This allows a client to define the pricing strategy for items based on future cost changes. The proposed retails can be based on current or market basket margin percentages.

Price Inquiry

Price inquiry is designed to allow retailers to retrieve the price of an item at an exact point in time. This price may be the current price of a particular item or the future price. You can search for prices based on the following search criteria:

- Merchandise hierarchy
- Item
- Zone group
- Zone
- Location
- Location (warehouse or store)
- Date

Worksheet

The RPM worksheet functionality is designed to allow for the maintenance of automatically generated price change and clearance proposals resulting from the RPM merchandise extract batch program. These proposed price changes/clearances are the product of existing strategies, calendars and item/location information.

Calendar

Calendars are set up in RPM for the primary purpose of attaching them to pricing strategies. Calendars last for a user defined period of time and contain review periods that occur one or many times over the duration of the calendar.

Aggregation Level

Aggregation level functionality is used in RPM to define parameters that vary at the department level. Within this functional area, a retailer selects a department and specifies the 'lowest definable level' at which the pricing strategies can be defined. The merchandise hierarchy levels at which a pricing strategy can be defined are department, class, and subclass.

When the merchandise extract runs to generate worksheets the 'Worksheet Level' setting is used to determine the level at which the worksheets should be generated. Merchandise hierarchy levels with varying strategies are aggregated into the same worksheet based on this aggregation level setting. For example, the strategies for a worksheet may be defined at the class level but if the worksheet level for the department that class is in is set to

department then a single worksheet status row exists per zone with all the classes rolled up to the department. The sales settings on the aggregation level screen determine the sales types that are pulled during the extract process and represented in the worksheet as historical sales. The inventory settings determine how warehouse inventory is utilized and which inventory the sell through calculations use.

Location Moves

Location moves in RPM allow a retailer to select a location that exists in a zone and schedule it to move to a different zone within a zone group on that scheduled date. This location move from zone to zone does not impact the location's pricing until the next pricing change for the new zone.

Simplified RPM

In order to satisfy the needs of clients who want a less complex version of RPM and the lower costs of ownership associated with it, a simplified version of RPM is offered.

The simplified version of RPM has the amount of functionality it offers limited by the security settings in RSM (which is established at the time of installation).

Simplified RPM is configured through RSM seed scripts. Running the simplified version of the RSM seed scripts will populate the named_permission and named_permission_dsc tables with only the tasks that are available for Simplified RPM. Only those tasks are then available within RSM when assigning task permissions to roles. The configuration of the security settings is determined by install scripts that are available with the RSM installation:

- RSM_RPM_SE_named_permission.sql – This script defines the named permissions and actions associated with them for RSM. This script should be run when simplified RPM is installed.

- RSM_RPM_named_permission.sql – This script defines the named permissions and actions associated with them for RSM. This script should be run in addition to the RSM_RPM_SE_named_permission.sql script when a client is using enterprise RPM.

- RSM_RPM_SE_named_permission_dsc.sql – This set of scripts defines the named permission descriptions and the language settings for the named permissions. These scripts should be run when simplified RPM is installed. There is one script for each supported language. They are named "RSM_RPM_SE_named_permission_dsc_*.sql where * is the language code with an optional country code.

- RSM_RPM_named_permission_dsc.sql – This set of scripts defines the named permission descriptions and the language settings for the named permissions. These scripts should be run in addition to the RSM_RPM_SE_named_permission_dsc.sql scripts when enterprise RPM is installed. There is one script for each supported language. They are named "RSM_RPM_named_permission_dsc_*.sql where * is the language code with an optional country code.

The functionality offered by both versions of RPM is outlined below:

- System Options
- Foundation
- Link Codes
- Zone Structure
- Price Guides
- Price Changes

- Clearances
- Promotion Events
- Promotions
- Promotion Threshold
- Promotion Constraints
- Vendor Funding Defaults
- Conflict Checking Results
- Price Inquiry

Enhanced Pricing Functionality

This functionality is NOT available in simplified RPM:

- Pricing Strategies
- Worksheets
- Candidate Rules
- Calendars
- Market Basket Codes
- Aggregation Level

RPM Integration with other Applications

RPM uses a Java architecture built on a layering model. Layers of the application communicate with one another through an established hierarchy and are only able to communicate with neighboring layers. The application is divided into a presentation layer, a middle tier consisting of services and business objects, and a database access/driver layer.

The segregation of layers has the following advantages, among others:

- The separation of presentation, business logic, and data makes the software cleaner, more maintainable, and easier to modify.
- The look and feel of the application is easily updated because the GUI is not tightly coupled to the back end.
- A layered architecture has become an industry standard.
- Portions of the data access layer (DAL) can be radically changed without effecting business logic or user interface code.
- The application takes advantage of Java database connectivity (JDBC), minimizing the number of interface points that must be maintained.
- Market-proven and industry-standard technology is utilized (for example, JSPs, JDBC, and so on).

The diagram below illustrates the RPM architecture:
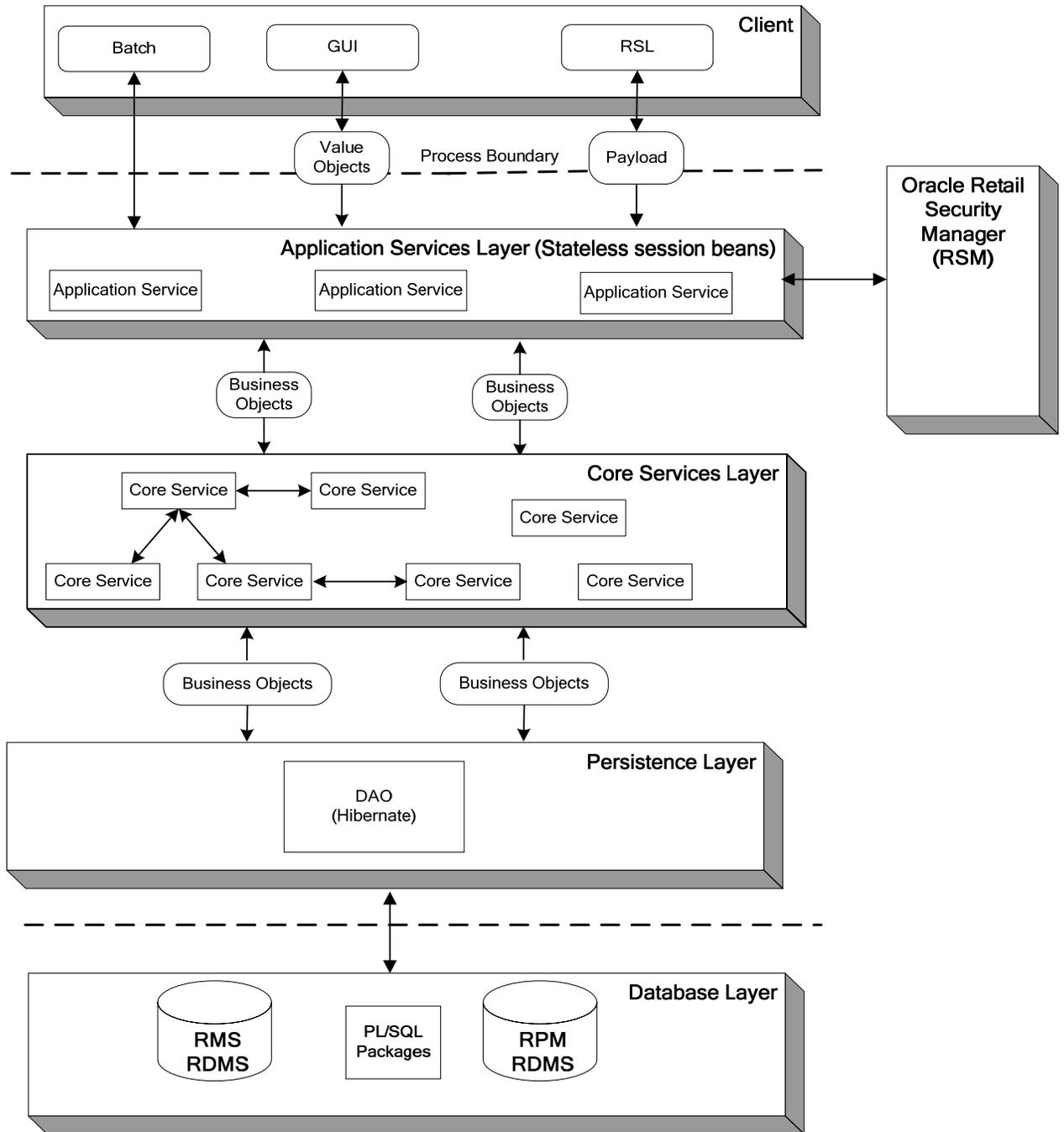
RPM exists on the same database schema as all the other Oracle Retail Merchandising Operations Management applications. As a result, there are a significant number of options for sharing information between applications

Information from the RPM application is shared with and retrieved from other Oracle Retail Merchandising Operations Management applications by reading directly from Oracle Retail Merchandising Operations Management application tables, by creating RPM views based on Oracle Retail Merchandising Operations Management application tables, by directly calling Oracle Retail Merchandising Operations Management

application packages, by allowing other Oracle Retail Merchandising Operations Management applications to call RPM packages, batch processes, and the RIB.

The diagram below illustrates the various applications that RPM interacts with in the merchandising footprint:
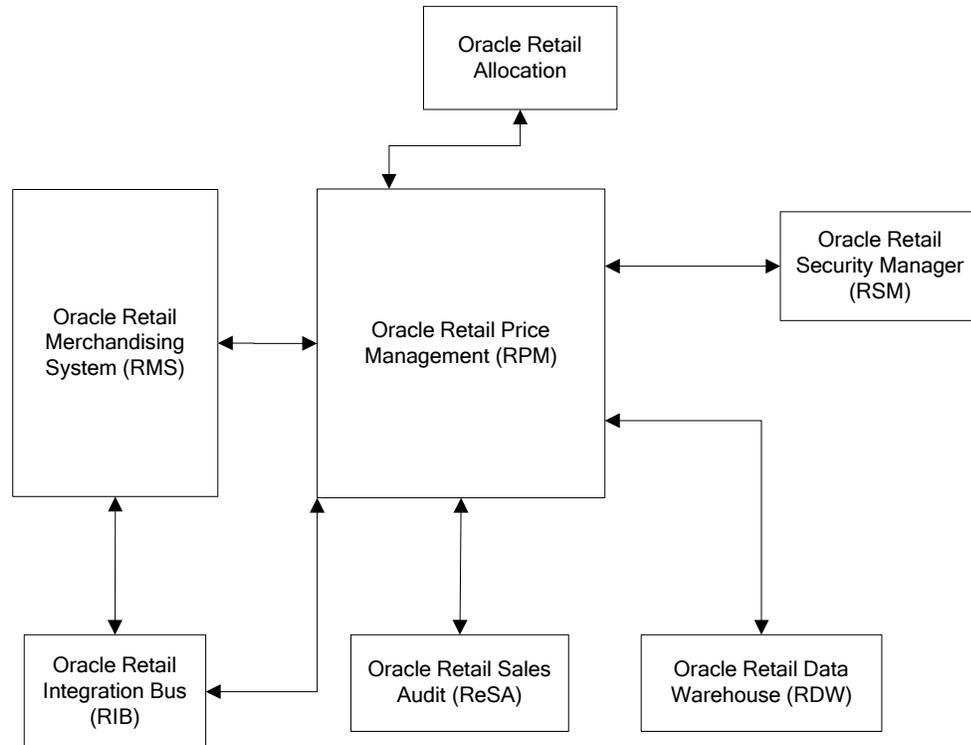
RPM and RMS/RTM/ReSA

RPM and RMS provide a client with flexible options for how to implement both solutions. RPM exists on the same database schema as RMS which allows information to be shared between applications from direct database reads, package calls, and batch processes. RPM also interacts with RMS by using the RIB. RPM offers the flexibility to forgo using the RIB for RMS and RPM installations through the use of a system setting. If RIB functionality is disabled, RPM uses API's to facilitate the exchange of information with RMS that would otherwise be conducted through the RIB. It is the client's responsibility to disable existing database triggers that support the RIB functionality should they choose not to implement it.

RPM provides RMS

RIB Implementation:

- **Regular Price Change Approval/Modification/Deletion –** If using the RIB implementation RPM publishes regular price change messages when a regular price change is created, modified, or deleted (for approved price changes). RMS subscribes to this message to generate (or remove if deleting) ticket request information for the regular price change request.

- **Promotional Price Change Approval/Modification/Deletion –** If using the RIB implementation RPM publishes promotional price change messages when a promotional price change is created, modified, or deleted (for approved promotions).

RMS subscribes to this message to generate (or remove if deleting) ticket request information for the promotional price change request.

- **Clearance Price Change Approval/Modification/Deletion –** If using the RIB implementation RPM publishes clearance price change messages when a clearance price change is created, modified, or deleted (for approved promotions). RMS subscribes to this message to generate (or remove if deleting) ticket request information for the clearance price change request.

- **Price Change Execution –** When regular, promotional, or clearance price changes are set to go into effect or end the PriceEventExecutionBatch owns the process. Once the pricing event has been processed by the batch program it updates pricing in RMS by interfacing with the RMSSUB_PRICECHANGE API in RMS.

- **Initial Pricing –** Initial pricing for items in RMS is dependant upon the primary zone group for the item defined in RPM and characteristics of that zone group. These characteristics include markup percent, markup percent type, and pricing guides. RPM provides this information to RMS through an API (MERCH_RETAIL_API_SQL).

- **Deal Creation –** RPM creates and associates Deals with regular, promotional, and clearance price changes. When this occurs RPM uses an RMS API (PM_DEALS_API_SQL) to create the deal in RMS.

Non-RIB Implementation:

- **Regular Price Change Approval/Modification/Deletion –** When a client is performing a non-RIB implementation, regular price change creation, modification, or deletion triggers a call to an RMS API to generate (or remove if deleting) the ticket request information.

- **Promotional Price Change Approval/Modification/Deletion –** When a client is performing a non-RIB implementation, promotional price change creation, modification, or deletion triggers a call to an RMS API to generate (or remove if deleting) the ticket request information.

- **Clearance Price Change Approval/Modification/Deletion –** When a client is performing a non-RIB implementation, clearance price change creation, modification, or deletion triggers a call to an RMS API to generate (or remove if deleting) the ticket request information.

- **Price Change Execution –** When regular, promotional, or clearance price changes are set to go into effect or end the PriceEventExecutionBatch owns the process. Once the pricing event has been processed by the batch program it updates pricing in RMS by interfacing with the RMSSUB_PRICECHANGE API in RMS.

- **Initial Pricing –** Initial pricing for items in RMS is dependant upon the primary zone group for the item defined in RPM and characteristics of that zone group. These characteristics include markup percent, markup percent type, and pricing guides. RPM provides this information to RMS through an API (MERCH_RETAIL_API_SQL).

- **Deal Creation –** RPM creates and associates Deals with regular, promotional, and clearance price changes. When this occurs RPM uses an RMS API (PM_DEALS_API_SQL) to create the deal in RMS.

RMS provides RPM with:

- **Foundation Data –** This information is essential to RPM functionality. To successfully setup price changes RPM needs to know the merchandise hierarchy, the organizational hierarchy, suppliers, and more. RPM is able to access this information via the RMS database.

- **Item –** Any price change created in RPM ultimately relates to an item/location. RPM needs to know all approved items currently in the merchandising system, the active, discontinued, or inactive item/location relationships for those items, suppliers with which the items are associated, and more. RPM is able to access this information via the RMS database.

- **Competitive Pricing Information –** RPM has the ability to create price changes based off competitive activity in the marketplace. RPM is able to access this information via the RMS database.

- **Deals –** Deals can be associated to price changes (including vendor funded promotions) in RPM. In order to associate price changes to an existing deal RPM needs visibility to the deals currently available in the system. RPM is able to access this information via the RMS database.

- **Event Notification –** There are certain events (outlined below) that occur in RMS that RPM needs to be notified of to ensure the appropriate processing occurs.

  - RIB Implementation:

    – **Store/Warehouse Creation -** When new stores and virtual warehouses are created in RMS, RPM needs to add them to a zone structure. To do this RMS provides RPM with the store and/or virtual warehouse being added, its pricing location, and its currency (to ensure it is the same as the zone it is being added to). When a client is using the RIB for an implementation, this notification process is handled through message publication and subsequent RPM subscription.

    – **Item/Location Creation –** When new item/location relationships are established RPM needs to verify that no future retail records currently exist, create an initial future retail record (for sellable items), and determine if there are any existing price changes that affect the item resulting in a future retail record for the price change as well. When a client is using the RIB for an implementation, this notification process is handled through message publication and subsequent RPM subscription.

    – **Item Modification –** This event is used to notify RPM when an item has been reclassified. The details of the reclassification are written to an item modification table in RPM for the next batch processing run. When a client is using the RIB for an implementation, this notification process is handled through message publication and subsequent RPM subscription.

    – **Department Creation –** This event is used to notify RPM when new departments are created in RMS. RPM creates aggregation level information for the new department using predefined system defaults. When a client is using the RIB for an implementation, this notification process is handled through message publication and subsequent RPM subscription.

- Non-RIB Implementation:
  - **Store/Warehouse Creation -** When new stores and virtual warehouses are created in RMS, RPM needs to add them to a zone structure. To do this RMS provides RPM with the store and/or virtual warehouse being added, its pricing location, and its currency (to ensure it is the same as the zone it is being added to). When a client is using a non-RIB implementation a store/warehouse creation event in RMS triggers an API call to RPM to execute the necessary processing via a batch process.
  - **Item/Location Creation –** When new item/location relationships are established RPM needs to verify that no future retail records currently exist, create an initial future retail record (for sellable items), and determine if there are any existing price changes that affect the item resulting in a future retail record for the price change as well. When a client is using a non-RIB implementation, an item/location creation event in RMS triggers an API call to RPM to execute the necessary processing via a batch process.
  - **Item Modification –** This event is used to notify RPM when an item has been reclassified. The details of the reclassification are written to an item modification table in RPM for the next batch processing run. When a client is using a non-RIB implementation, an item modification creation event in RMS triggers an API call to RPM to execute the necessary processing via a batch process.
  - **Department Creation –** This event is used to notify RPM when new departments are created in RMS. RPM creates aggregation level information for the new department using predefined system defaults. When a client is using a non-RIB implementation a department creation event in RMS triggers an API call to RPM to execute the necessary processing via a batch process.

RPM and RTM

No information exchanged.

RPM and ReSA

RPM provides ReSA with:

Promotion Information – RPM needs to provide ReSA with promotion information. This information is provided via a RETL extract program and is used by ReSA to validate promotional sales transactions.

ReSA provides RPM with:

No information is provided.

RPM and Oracle Retail Allocation

RPM provides Oracle Retail Allocation with:

Future Retail Price Data – Allocation uses future retail price data stored in RPM to provide the total retail value of an allocation. This interface is optional for a retailer and must be configured during implementation. Oracle Retail Allocation uses an API to access this information from RPM.

Promotions – Users in Oracle Retail Allocation can select a promotion from RPM to associate an allocation with.

Oracle Retail Allocation provides RPM with:

No information is provided.

RPM and ReIM

RPM provides ReIM with:

No information is provided.

ReIM provides RPM with:

No information is provided.

RPM and ARI

ARI is a monitoring system that interacts with any applications database (including RPM). As such it does not "use" any information from RPM, rather it monitors the RPM database for events defined by a client and notifies the client when said events occur.

RPM and RDW

RPM provides RDW with:

RPM needs to provide RDW with promotion information. This information is provided via a RETL extract program (prcilddm.ksh).

RDW provides RPM with:

No information is provided.

RPM Users and Security

RPM has its security managed through an external system, RSM. RSM is an application that provides a centralized method of authenticating and authorizing system users.

RSM leverages a Light Directory Access Protocol (LDAP)-compliant directory service to maintain and authenticate valid users. It is the system administrator's responsibility to ensure all users are setup in LDAP. If a client has an existing LDAP server where users are currently managed, RSM can be pointed to that server to eliminate the redundancy of maintaining multiple user names/passwords for the same user across a clients applications.

RSM provides a two-tier security structure that consists of Application and Data level security. RSM provides centralized administration screens for system administrators to create application and data level permissions.

Application Level Security

This allows applications to limit what business functionality users can access in the system and how they can interact with it when they can access it. In order to determine what business functionality a user has access to RSM uses Named Permissions.

- **Named Permissions -** These are pieces of business functionality around which the application has security. For example, if RPM has 'promotions' functionality surrounded by security, RPM creates a 'promotions' named permission. Named permissions data is sent to the RSM database during installation.

To determine what actions a user can perform for the Named Permissions RSM has actions defined for each Named Permission.

▪ **Actions –** These define how the user interacts with the functionality contained by the Named Permission. The type of actions that are attached to a Named Permission are as follows:

– **None –** Users associated to the role have access to the permission but no actions.

– **Edit –** Users associated with the role are allowed to see all secured information in a workflow.

– **View –** Users associated with the role are allowed to see all secured information in a workflow, but not make any changes to the data in the workflow.

– **Approve –** Users associated with the role are allowed to change the status of a workflow to Approved.

– **Submit –** Users associated with the role are allowed to change the status of a workflow from Worksheet to Submitted.

– **Emergency –** Users associated with the role are granted special access that goes beyond normal day-to-day access functionality. They can thus bypass normal delays in processing.

Using Named Permissions and Actions, RSM provides a requesting application (RPM in this case) with the application access information it needs to determine what security is enforced for each functional area within the application.

Data Level Security

This allows applications to limit user's access to information based on hierarchy (merchandise and location) permissions. Applications either provide the details of these types up front with SQL scripts or dynamically by implementing an RSM interface and exposing it to the RSM service. RSM does not understand application specific data (for example, RSM does not know the difference between departments and locations). To RSM, the data is a tag (for example, department) and a specific value (for example, 1000). This information is passed back to calling applications, and it is the applications responsibility to apply the data level permissions appropriately.

Once Application and Data Level security parameters are defined in RSM they still need to be associated to users based upon their business roles. To facilitate this process RSM allows a system administrator to create security roles which application and data level security can be associated with. These roles can then be associated with users which causes the users to inherit the security permissions associated with their security role. When a user logs onto RPM they are authorized to use the business functionality and data associated with their role. The Diagram below illustrates this concept:

John Smith

User

Role1

Role

Named permissions

Hierarchy permissions
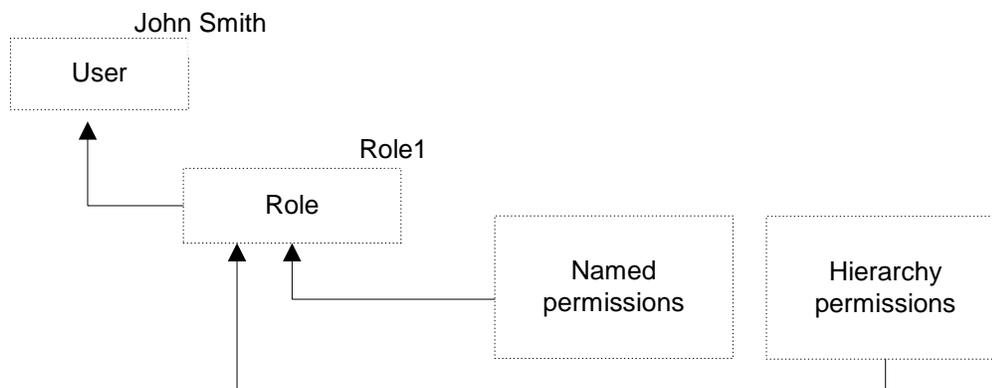
For additional details on RSM operation and use are found in the RSM Operations Guide and the RSM User Guide.

Internationalization

The RPM application is internationalized to support multiple languages. All text, date, formatting, and other locale specific information is placed in external files which are interfaced into the application. This allows the client to configure the application to meet their specific language needs.

The text in the .properties files below is translated so that the application functions in local settings. When a country code other than the default is used, the retailer populates the _xx.properties files below, where the retailer's applicable country equals xx.

- messages.properties and messages_xx.properties
- resources.properties and resources_xx.properties
- codes.properties and codes_xx.properties
- application_definition_rpm_messages.properties (contains labels that are displayed in the RSM GUI for data security setup)
- worksheet_column_names.properties and worksheet_column_names_xx.properties (contains the labels from the worksheet details table's columns)

In order to leverage internationalization the client machine's operating system also needs to be set to the appropriate locale.

Further information on Internationalization can be found in the RPM Operations Guide.

Oracle Retail Active Retail Intelligence (ARI) Overview

Exception Reporting

Oracle Retail Active Retail Intelligence (ARI) takes business rules that users are applying to reports in their daily tasks and applies them directly to the retail systems. This allows the correct people within the retail organization to be alerted when the business rule conditions have been met and informs them of what the appropriate actions might be to resolve it. This enables a retailer to apply the best practice to their entire organization by tying actions to information in an intelligent way.

A business example of an exception reporting rule is a rule that looks for orders that are past their ship date but have not been shipped yet. Rather than having an analyst scan a report to identify which orders have been shipped late a rule could be created in ARI to identify these orders for the analyst and the recommended steps to take to resolve the late shipping issue with the vendor.

Workflow Management

ARI maps out a retailer's business processes and notifies the user(s) responsible for those processes whenever new tasks require attention. ARI also lists the appropriate actions that may be taken to complete these tasks and allows users to take action directly from the screen where they receive the alert.

If the alert is not acted on within an acceptable timeframe, ARI can define an automated action that needs to take place. This can be to raise the priority of the alert or send it to a supervisor or take an automated system action to ensure that single users do not become a bottleneck.

A business example of a workflow management rule is a rule that notifies an inventory manager when orders are ready for approval. This gives the manager instant visibility to all orders that need to be approved for the day rather than requiring the manager to search for orders meeting this requirement. Additional logic can also be added to notify the manager's supervisor when orders awaiting approval are not approved after a pre-determined timeframe.

Enterprise Process Modification

ARI allows a retailer to modify process within Oracle Retail to more closely match their business processes. This is achieved in most cases without modifying the underlying retail system.

A business example of an Enterprise Process Modification is a client changing its transfer approval process from analysts being able to approve transfers to only managers being able to approve transfers. Rules could be developed in ARI to send alerts to both the analyst and manager when transfers are submitted to educate them on the new business process to help ensure a smooth transition.

ARI Integration with other Applications

ARI is a monitoring system that interacts with any applications database. As such it does not "use" any information from other retailing applications; rather it monitors the retailing application database(s) for events defined by a client and notifies the client when said events occur.

The only complication that can result is when a client is making database changes that affect the data mapping for rules already defined in ARI. In this case the rule needs to be changed to reflect the new data mapping.

ARI Security

Active Retail Intelligence provides no special security features or safeguards. Addressing any site-specific security issues involving Active Retail Intelligence is the customer's responsibility. Security settings in other applications with which Active Retail Intelligence interacts are not overridden or circumvented by Active Retail Intelligence. Whereas this is generally desirable, it is a consideration when determining to whom Active Retail Intelligence alerts are routed. Sending an alert to a user who does not have the privileges to take the actions necessary to resolve the event may prove frustrating and counter-productive. Users should be educated about this issue so that they avoid forward events that have actions with limited access as well.

At a data level, Active Retail Intelligence detection is necessarily done with full access privileges to all data. Individual users with data level security may see different values for some parameters (in particular those involving sums) than the values seen by Active Retail Intelligence. This may cause adverse effects such as a user looking at an event automatically causing it to close because the user's limited data access causes the event to see values that make Active Retail Intelligence think the exception is no longer an issue when in fact it still is. For this reason Oracle urges extreme caution when designing Active Retail Intelligence processes that involve users with limited data access. The consequences of missing alerts are great in an exception driven enterprise, so extra care is needed in the technical analysis of how such Active Retail Intelligence processes behave.

Appendix A: RMS RIB Related Tables and Triggers

RMS User Creation Script – A copy of the user creation script is available in the appendix of the RMS Installation Guide.

RMS Tables/Triggers affected by No RIB RMS/RPM implementation – Below is a list of the RMS tables, the triggers attached to the tables, and the staging tables that would be affected in a no RIB implementation of RMS and RPM. Careful analysis should be done to determine whether or not other applications will require these messages prior to disabling the triggers. If they do not, the triggers can be disabled.

> **Note:** Future implementations may require these staging tables and triggers. If clients are unsure if additional products are going to be implemented at a future time, it is recommended that the triggers be disabled rather than removed.

RMS Tables	RMS Table Triggers	RMS mfqueue tables
Deps	EC_TABLE_DEP_AIUDR	MERCHHIER_MFQUEUE
Class	EC_TABLE_CLA_AIUDR	MERCHHIER_MFQUEUE
Subclass	EC_TABLE_SCL_AIUDR	MERCHHIER_MFQUEUE
Store	EC_TABLE_STR_AIUDR	STORE_MFQUEUE
Wh	EC_TABLE_WHA_AIUR	WH_MFQUEUE

Appendix B: Partitioning

Establish Database Partitioning Strategy

Partitioning is mandatory for specific tables. **Please review this section in its entirety before proceeding with the installation.**

Requirements for mandatory and optional partitioning are defined in the Microsoft Excel spreadsheet located in INSTALL_DIR/ddl/part/RMS_partition_definition.xls. Since partitioning strategies are complex, this step should be implemented by an experienced individual who has a thorough understanding of partitioning principles and the data to be partitioned.

Use the Microsoft Excel spreadsheet to determine an appropriate partitioning strategy (INSTALL_DIR/ddl/part/RMS_partition_definition.xls). The "Partition Method" column indicates the recommended partitioning option(s) for each table. Refer to the information in this file to modify the DDL for partitioned tables. This can be done by manually changing the file INSTALL_DIR/ddl/rms12_part.tab or by implementing the process defined below. This file will be used later in the installation process.

> **Note:** Refer to Oracle10g Database Concepts Release 2 Chapter 18 "Partition Tables and Indexes" for further details regarding partitioning concepts.

Hash partitions: `To calculate the number of hash partitions and sub-partitions, enter values for the three parameters highlighted in yellow at the top of the RMS worksheet. Altering these values will update the "Number of Partitions" column for HASH partitioned/sub-partitioned tables. The values in these columns indicate the number of hash partitions/sub-partitions to create.`

Partition Factor: `This value is used to adjust the number of hash partitions. It is based on the number of active items per location and transactions per location/day. If the number of items/location and/or transactions/store/day is low, the value of partition factor should be high. This will calculate fewer hash partitions. The typical factor value ranges from 2 to 4 and in special cases, it can be 10 or more.`

> **Note:** Changing the items/location and transactions/store/day fields on the worksheet does not automatically impact the factor value. They are used as a point of reference only.

Sub-Partition Factor: `This value is used to adjust the number of hash sub-partitions. The partition strategy for historical information determines the value of this number. If the number of range partitions is high, the value of sub-partition factor should be high to control the number of sub-partitions. Typically, this value will be 2.`

Locations: `The total number of active stores and warehouses.`

Range partitions: `Determine the purging strategy for all of the tables that are RANGE partitioned. Each partition should have a range of multiple key values. For example, if the strategy were to have data available for one year and to purge it every three months, five partitions would be created. In this case, four 3-month partitions and a "max value" partition to contain all data greater than the defined ranges would result. Refer to the "Comments" column and update the value`

in the "Number of Partitions" column.   The value in this column indicates the
number of range partitions to create.

List partitions: The DAILY_ITEM_FORECAST and ITEM_FORECAST must be LIST partitioned.
If number of partition keys is relatively static, change the value in the
"Partition Method" column to LIST where allowed. This method will ensure that
each partition key has a separate partition and that none are empty. The "Number
of Partitions" column will be automatically updated with the proper number of
locations in the event the partition method is changed. The value in this column
indicates the number of list partitions to create.

Step 1: Modify partition_attributes.cfg

Modify INSTALL_DIR/ddl/part/partition_attributes.cfg based on the
partitioning strategy defined in RMS_partition_definition.xls. Changes to this file
should be made only as indicated.

partition_attributes.cfg file: (file is comma-delimited)

Sample Entry:

ITEM_LOC_HIST,EOW_DATE,RANGE,item_loc_hist.eow_date.date,64,LOC,HASH,item_
loc_hist.loc.number,64,RETEK_DATA

Field 1: Table Name - *Do not modify*

Field 2: Partition Key - *Do not modify*

Field 3: Partition Method - Modify based on value in "Partition Method" column
in RMS_partition_definition.xls - Valid values are RANGE, LIST, or HASH (case
sensitive)

Field 4: Partition Data Definition Filename - *Do not modify - This field is ignored
if Partition Method is not RANGE or LIST*

Field 5: Partition Hash Count – Modify based on value in "Hash Partitions
Calculated" column in RMS_partition_definition.xls. *This field is ignored if
Partition Method is not HASH*

Field 6: Sub-Partition Key - *Do not modify*

Field 7: Sub-Partition Method - Modify based on value in "Sub-partition
Method" column in RMS_partition_definition.xls - Valid values are LIST or
HASH (case sensitive)

Field 8: Sub-Partition Data Definition Filename - *Do not modify - This field is
ignored if Sub-Partition Method is not RANGE or LIST*

Field 9: Sub-Partition Hash Count - Modify based on value in "Hash Sub-
partitions Calculated" column in RMS_partition_definition.xls. *This field is
ignored if Sub-Partition Method is not HASH*

Field 10: Tablespace Name - *Optional. Default is RETEK_DATA*

Step 2: Modify Data Definition Files

Tables partitioned or sub-partitioned by RANGE or LIST have a corresponding
data definition file in the INSTALL_DIR/ddl/part/data_def directory and
should not be removed or renamed. These files are used to define the data
boundaries for each partition. Values must be entered in each file based on the
data type of the "Partition Key" column in RMS_partition_definition.xls. Refer
to the "Comments" column in this file for additional information. The value in
the "Number of Partitions" column indicates the number of entries to place in
the data definition file.

The format of a data definition file name is <table name>.<partition key column>.<partition key data type>, e.g., item_loc_hist.eow_date.date. When placing data into these files, enter one data partition value per line.

When entering varchar2 values in a data definition file, do not use quotation marks. When defining date values, use the DDMMYYYY format.

sampletable.action_date.date:

```
01012004
01012005
```
sampletable.state.varchar2:

```
Minnesota
Iowa
```
sampletable.location.number:

```
1000
2000
```
When using RANGE partitioning, the data definition files will use the "value less than" concept. For example, in sampletable.action_date.date above, the first partition will contain all data less than 01012004. The second partition will contain all data greater than or equal to 01012004 and less than 01012005. A third "MAXVALUE" partition will automatically be created for all data greater than or equal to 01012005.

When using LIST partitioning, the data definition files will use the "value equal to" concept. For example, in sampletable.state.varchar2 above, the first partition will contain all data equal to Minnesota. The second partition will contain all data equal to Iowa.

Step 3: Generate DDL for Tables – Run partition.ksh

Execute INSTALL_DIR/ddl/part/partition.ksh at the UNIX command prompt. This script will read configuration information from the partition_attributes.cfg file and generate the partitioned DDL file INSTALL_DIR/ddl/rms12_part.tab. This file will be used later during the installation process.

Sample output from partition.ksh:

```
<INSTALL_DIR>/ddl/part > ./partition.ksh
##########################################################################
# partition.ksh:
# This script will read the partition_attributes.cfg file and any
referenced
# data definition files and generate partitioned DDL.
##########################################################################
# The non-partitioned DDL file is ../rms120.tab.
# The partitioned DDL file that will be generated is ../rms120_part.tab.
##########################################################################
Checking partition_attributes.cfg for errors
Generating Partitioned DDL for DAILY_DATA
Generating Partitioned DDL for DAILY_ITEM_FORECAST
Generating Partitioned DDL for DAILY_SALES_DISCOUNT
…
partition.ksh has generated the DDL for partitioned tables in the
../rms12_part.tab file.
Completed successfully
```

C

Appendix C: Scripts

Load Seed Data at Time of Installation

The following table outlines Oracle-supplied data installation scripts and the tables populated by these scripts. Please note that some tables populated by these scripts may be modified for final configuration or updated with additional values prior to implementation.

Script Name	Calls the following scripts/packages	Tables Inserted
RIBDATA.SQL	Calls ALL_RIB_TABLE_VALUES.sql to insert into	RIB_ERRORS RIB_LANG RIB_TYPE_SETTINGS RIB_SETTINGS
	Calls RIB_DOCTYPES.sql which launches a sql loader session to insert into	RIB_DOCTYPES
RMSUOM.SQL	Inserts in to	UOM_CLASS
RMSCOUNTRIES.SQL	Inserts in to	COUNTRY
RMSSTATES.SQL	Inserts in to	STATE
RMSCURRENCIES.SQL	Inserts in to	CURRENCIES
STATICIN.SQL	Directly Inserts in to	SYSTEM_OPTIONS ADD_TYPE ADD_TYPE_MODULE COST_CHG_REASON DUMMY DEAL_COMP_TYPE DOC_LINK DYNAMIC_HIER_CODE INV_STATUS_TYPES INV_STATUS_CODES LANG MC_REJECTION_REASONS ORDER_TYPES PRICE_ZONE_GROUP SAFETY_STOCK_LOOKUP TRAN_DATA_CODES TRAN_DATA_CODES_REF TSF_TYPE VEHICLE_ROUND COST_ZONE_GROUP
	Calls ELC_COMP_PRE_HTSUPLD.SQL to insert into	CVB_HEAD ELC_COMP

Script Name	Calls the following scripts/packages	Tables Inserted
	Calls GENERAL_DATA_INSTALL_SQL. VAT_CODE_REGION to insert into	VAT_REGION, VAT_CODES, VAT_CODE_RATES
	Calls GENERAL_DATA_INSTALL_SQL. ADD_TYPE to insert into	ADD_TYPE
	Calls GENERAL_DATA_INSTALL_SQL. ADD_TYPE_MODULE to insert into	ADD_TYPE_MODULE
	Calls GENERAL_DATA_INSTALL.UNIT_OPTIONS to insert into	UNIT_OPTIONS
	Calls GENERAL_DATA_INSTALL_SQL.ELC_COMP _EXPENSES to insert into	CVB_HEAD CVB_DETAIL
	Calls GENERAL_DATA_INSTALL_SQL.ELC_COMP _EXPENSES, GENERAL_DATA_INSTALL_SQL.UP_CHARG E and GENERAL_DATA_INSTALL_SQL.BACKHAU L_ALLOWANCE to insert into	ELC_COMP
ADD_FILTER_POLI CY.SQL	Calls DBMS_RLS.ADD_POLICY function to implement fine grained access control.	
NAVIGATE.SQL	Inserts in to	NAV_ELEMENT NAV_SERVER NAV_COMPONENT EVIEW NAV_ICON
NAVROLE.SQL	Inserts in to	NAV_ELEMENT_MODE_ROLE
CODES.SQL	Inserts in to	CODE_HEAD CODE_DETAIL CODE_DETAIL_TRANS

Script Name	Calls the following scripts/packages	Tables Inserted
POPULATE_SEC_F ORM _ACTION.SQL	Inserts in to	SEC_FORM_ACTION
RESTART.SQL	Inserts in to	RESTART_PROGRAM_STATUS RESTART_CONTROL+C11
RTK_ERRORS.SQL	Inserts in to	RTK_ERRORS
RTK_REPORTS.SQL	Inserts in to	RTK_REPORTS
TL_COLUMNS.SQL	Inserts in to	TL_COLUMNS
WIZARD.SQL	Inserts in to	WIZARD_TEXT
CONTEXT.SQL	Inserts in to	CONTEXT_HELP
POPULATE_FORM_ LINKS.SQL	Inserts in to	FORM_LINKS
POPULATE_FORM_ LINKS_ROLE.SQL	Inserts in to	FORM_LINKS_ROLE
UOM_X_CONVERSI ON.SQL	Inserts in to	UOM_X_CONVERSION
VAR_UPC_EAN_LO AD.SQL	Inserts in to	VAR_UPC_EAN
MULTIVIEW_DATA .SQL	Inserts in to	MULTIVIEW_SAVED_45 MULTIVIEW_DEFAULT_45
RMSUOMCONV1.S QL	Inserts in to	UOM_CONVERSION
RMSUOMCONV2.S QL	Inserts in to	UOM_CONVERSION
CALENDAR.SQL	Inserts in to	HALF CALENDAR SYSTEM_VARIABLES PERIOD

Script Name	Calls the following scripts/packages	Tables Inserted
SA_SYSTEM_REQUIRED.SQL	Calls SA_METADATA.SQL to insert into	POS_TENDER_TYPE_HEAD SA_CC_VAL SA_REFERENCE SA_ERROR_CODES SA_EXPORT_OPTIONS SA_ERROR_IMPACT
	Calls SA_METADATA.SQL which in turn calls SA_REALM_TYPE.SQL to insert into	SA_REALM_TYPE
	Calls SA_METADATA.SQL which in turn calls SA_REALM.SQL to insert into	SA_REALM
	Calls SA_METADATA.SQL which in turn calls SA_PARM_TYPE.SQL to insert into	SA_PARM_TYPE
	Calls SA_METADATA.SQL which in turn calls SA_PARM.SQL to insert into	SA_PARM
RMS12RTM.SQL	Calls ENTRY_TYPE.SQL to insert into	ENTRY_TYPE
	Calls ENTRY_STATUS.SQL to insert into	ENTRY_STATUS
	Calls OGA.SQL to insert into	OGA
	Calls TARIFF_TREATMENT.SQL to insert into	TARIFF_TREATMENT
	QUOTA_CATEGORY.SQL	QUOTA_CATEGORY
	Calls COUNTRY_TARIFF_TREATMENT.SQL to insert into	COUNTRY_TARIFF_TREATMENT
	Calls HTS_HEADINGS.SQL to insert into	HTS_CHAPTER
BASE_FORM_MENU_ ELEMENTS.SQL	Calls all the xml.sql scripts for each form present in the application to insert into	FORM_ELEMENTS FORM_ELEMENTS_TEMP FORM_ELEMENTS_LANGS_TEMP FORM_MENU_LINK MENU_ELEMENTS MENU_ELEMENTS_TEMP MENU_ELEMENTS_LANGS_TEMP