Oracle® Retail Demand Forecasting

Implementation Guide

Release 14.1.2

E71747-02

February 2016

ORACLE®

Oracle Retail Demand Forecasting Implementation Guide, Release 14.1.2

Primary Author: Melissa Artley

condition of the agreement and this ordering document, you shall not cause or permit alteration of any VAR Applications. For purposes of this section, “alteration” refers to all alterations, translations, upgrades, enhancements, customizations or modifications of all or any portion of the VAR Applications including all reconfigurations, reassembly or reverse assembly, re-engineering or reverse engineering and recompilations or reverse compilations of the VAR Applications or any derivatives of the VAR Applications. You acknowledge that it shall be a breach of the agreement to utilize the relationship, and/or confidential information of the VAR Applications for purposes of competitive discovery.

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

Contents

3 Integration

4 Installation Consideration

5 Configuration Considerations

6 Batch Processing

7 Cross Promotion Effects Module (CPEM)

List of Figures

List of Tables

Send Us Your Comments

Oracle Retail Demand Forecasting Implementation Guide, Release 14.1.2.

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

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

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

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

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

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

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

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

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

Preface

The Oracle Retail Demand Forecasting (RDF) Implementation Guide describes postinstallation tasks that need to be performed in order to bring RDF online and ready for production use.

Audience

This Implementation Guide is intended for the RDF application integrators and implementation staff, as well as the retailer's IT personnel. This guide is also intended for business analysts who are looking for information about processes and interfaces to validate the support for business scenarios within RDF and other systems across the enterprise.

Documentation Accessibility

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

Access to Oracle Support

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

Related Documents

For more information, see the following documents in the Oracle Retail Demand Forecasting Release 14.1.2 documentation set:

- *Oracle Retail Demand Forecasting Configuration Guide*
- *Oracle Retail Demand Forecasting Implementation Guide*
- *Oracle Retail Demand Forecasting Installation Guide*
- *Oracle Retail Demand Forecasting Release Notes*
- Oracle Retail Predictive Application Server documentation

The following documentation may also be needed when implementing RDF:

- *Oracle Retail Planning Batch Script Architecture Implementation Guide*

Supplemental Documentation

The following document is available through My Oracle Support at the following URL:

https://support.oracle.com

Oracle Retail Demand Forecasting 14.1.2 Cumulative Fixed Issues (Note ID 2096104.1)

This document details the fixed issues and defects for all RDF, Curve, and Grade patch releases prior to and including the current release.

Customer Support

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

https://support.oracle.com

When contacting Customer Support, please provide the following:

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

Review Patch Documentation

When you install the application for the first time, you install either a base release (for example, 14.1) or a later patch release (for example, 14.1.2). If you are installing the base release, additional patch, and bundled hot fix releases, read the documentation for all releases that have occurred since the base release before you begin installation. Documentation for patch and bundled hot fix releases can contain critical information related to the base release, as well as information about code changes since the base release.

Improved Process for Oracle Retail Documentation Corrections

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

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

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

An updated version of the applicable Oracle Retail document is indicated by Oracle part number, as well as print date (month and year). An updated version uses the

same part number, with a higher-numbered suffix. For example, part number E123456-02 is an updated version of a document with part number E123456-01.

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

Oracle Retail Documentation on the Oracle Technology Network

Oracle Retail product documentation is available on the following web site:

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

(Data Model documents are not available through Oracle Technology Network. You can obtain them through My Oracle Support.)

Conventions

The following text conventions are used in this document:

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

1

Introduction

Oracle Retail Demand Forecasting (RDF) is a statistical and promotional forecasting solution. It uses state-of-the-art modeling techniques to produce high quality forecasts with minimal human intervention. Forecasts produced by the RDF system enhance the retailer's supply-chain planning, allocation, and replenishment processes, enabling a profitable and customer-oriented approach to predicting and meeting product demand.

Today's progressive retail organizations know that store-level demand drives the supply chain. The ability to forecast consumer demand productively and accurately is vital to a retailer's success. The business requirements for consumer responsiveness mandate a forecasting system that more accurately forecasts at the point of sale, handles difficult demand patterns, forecasts promotions and other causal events, processes large numbers of forecasts, and minimizes the cost of human and computer resources.

Forecasting drives the business tasks of planning, replenishment, purchasing, and allocation. As forecasts become more accurate, businesses run more efficiently by buying the right inventory at the right time. This ultimately lowers inventory levels, improves safety stock requirements, improves customer service, and increases the company's profitability.

The competitive nature of business requires that retailers find ways to cut costs and improve profit margins. The accurate forecasting methodologies provided with RDF can provide tremendous benefits to businesses.

For a more detailed overview of the functionality within RDF, see the *Oracle Retail Demand Forecasting User Guide*.

Contents of this Guide

This implementation guide addresses the following topics:

- Chapter 1, "Introduction": Overview of the RDF business workflow and skills needed for implementation.
- Chapter 2, "Implementation Considerations": Explanation of the factors to take into consideration before performing the implementation.
- Chapter 3, "Integration": Overview of integration and explanation of the RDF data flow and integration script.
- Chapter 4, "Installation Consideration": Information for the setup that must be done prior to building the RDF domain and installing RDF.
- Chapter 5, "Configuration Considerations": Information on the functional changes or enhancements that can be made for RDF.

- Chapter 6, "Batch Processing": Information on the RDF batch forecast process.
- Chapter 7, "Cross Promotion Effects Module (CPEM)": Information on the module for cross promotion effects.
- Chapter 8, "AutoSource": Information on the AutoSource utility.
- Chapter 9, "Forecast Approval Alerts": Information on the usage and configuration of Forecast Approval Alerts.
- Chapter 10, "Adding New Local Domains": Information on the process of adding new local domains.
- Chapter 11, "Internationalization": Translations provided for RDF.
- Appendix A, "RPAS and RDF Integration with RMS": RMS to RDF transformation programs, RDF to RMS extract programs, Grade (RPAS) to RMS extract programs, and Curve (RPAS) to Allocation extract programs.

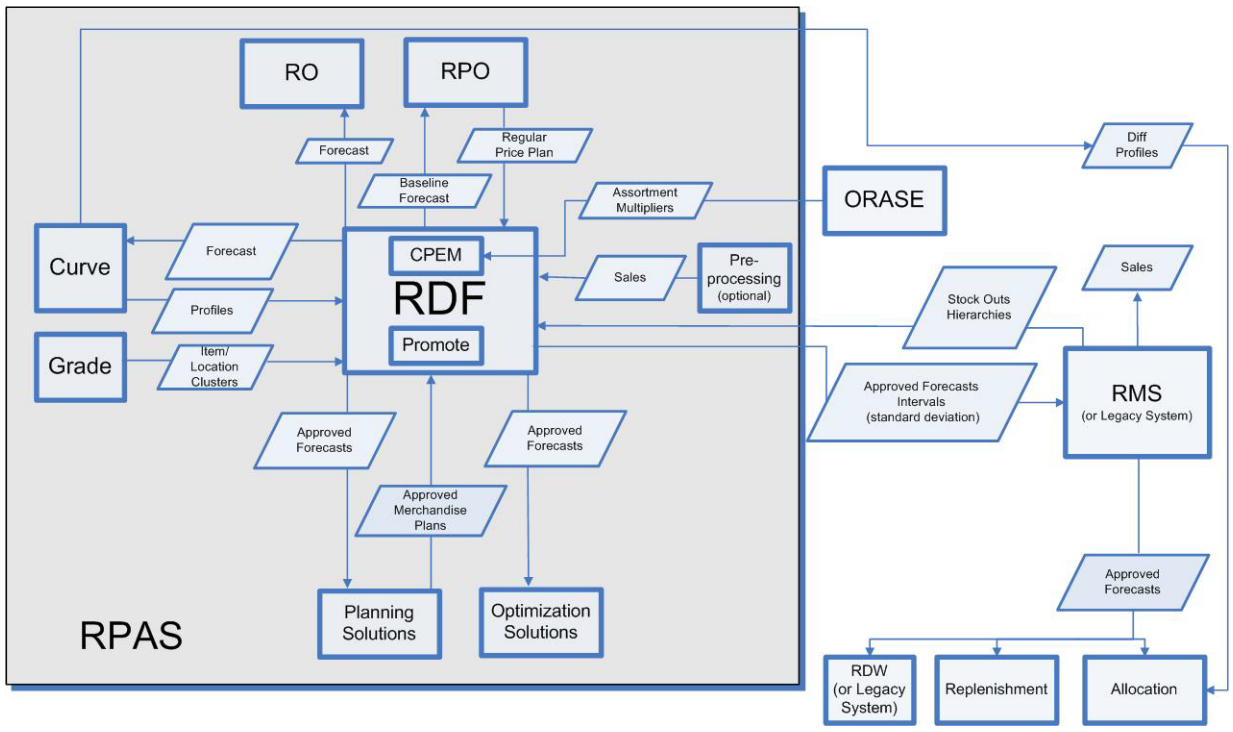
RDF and the Oracle Retail Enterprise

Oracle Retail has designed a forecasting solution separate from replenishment, allocation or planning. In order to provide a single version of the truth, it is crucial to free up the user's time and supply the tools to focus on the analysis of forecast exceptions, historical data, and different modeling techniques. This empowers the user to make better decisions, thus improving overall accuracy and confidence in the results of the forecast demand.

Within the Oracle Retail Enterprise, Oracle Retail Merchandising System (RMS) supplies RDF with Point-of-Sale (POS) and hierarchy data that is used to create a forecast. Once the forecast is approved, it is exported to RMS in order to calculate a recommended order quantity. Forecasts can also be utilized (no export process required) in any Retail Predictive Application Server (RPAS) solution to support merchandise, financial, collaborative, and price planning processes.

Figure 1–1 shows the interaction between RDF, RPAS, and other applications.

Figure 1–1 RDF and the Oracle Retail Enterprise

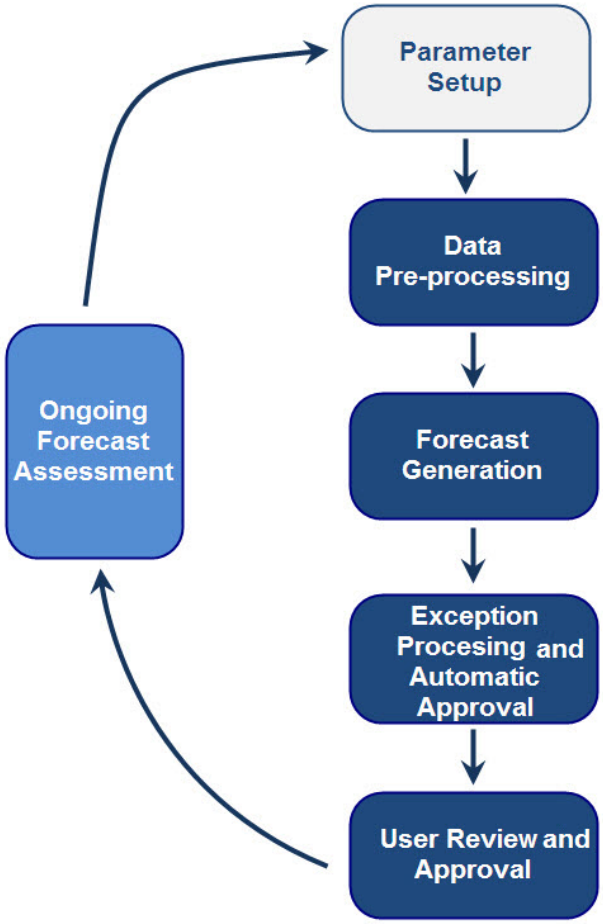


RDF Business Process Workflow

One of the challenges in retail forecasting is the data volumes. The RDF business process focusses on automation, accuracy and lends itself to easy analysis. RDF focuses on automation by automatically selecting best forecast methods and parameters as well as by automatic approval of forecasts that don't meet any exception criteria. Also, it allows users to analyze and manually approve forecasts. Forecast scorecarding allows users to monitor forecast accuracy over time and re-tune settings if necessary.

Figure 1–2 illustrates the RDF business process workflow.

Figure 1–2 Business Process Workflow



Parameter Setup

Following the initial setup, these parameters are not set on a scheduled basis, but are updated as needed.

- Preprocessing and alert parameter setup
- Sets forecast methods, parameters, and specifies source levels
- Sets history start and end dates

Data Pre-processing

For data pre-processing, RDF:

- Corrects for lost sales due to stock-outs
- Cleanses data for effects of promotions and short-term price changes (optional)
- Manual data-scrubbing (fake history and user history overrides)

Forecast Generation

For forecast generation, RDF:

- Computes demand parameters (seasonality, level, trend)
- Optimizes exponential smoothing parameters
- Allows you to select best forecast method for item/location or use Automatic Exponential Smoothing (AutoES)

Exception Processing and Automatic Approval

For exception processing and automatic approval, RDF:

- Evaluates forecast for exceptions based on specific alert criteria
- Automatically approves non-alerted forecasts
- Allows you to review flagged exception forecasts

User Review and Approval

For user review and approval, RDF:

- Reviews and analyzes forecasts, allowing for overrides if necessary
- Approves forecasts

Ongoing Forecast Assessment

For ongoing forecast assessment, RDF:

- Assesses user overrides versus system forecasts against actuals
- Assesses forecast quality and user adoption
- Retunes parameter settings as needed

Key Features of RDF

RDF provides the following features:

- Pre-processing to correct for stock-outs and other data anomalies
- Generation of forecasts
- Optimizes forecasting methods and exponential smoothing parameters
- Selects best forecasting methods and parameters to overcome data sparsity and reliability issues
- Generation of alerts and automatic approval of forecasts
- Allows you to facilitate review of analysis and approval of forecasting

Skills Needed for Implementation

A typical RDF implementation team has technical and application/business consultants in addition to other team members.

The technical and application/business consultants need to have a high level understanding of other applications that RDF can integrate with, which include:

- Advanced Inventory Planning (AIP)
- Analytic Parameter Calculator for Regular Price Optimization (APC-RPO)
- Analytic Parameter Calculator for Replenishment Optimization (APC-RO)

- Replenishment Optimization (RO)
- Retail Merchandising System (RMS)
- Retail Price Optimization (RPO)

In addition, both technical and application/business consultants need to have an understanding of RPAS, its calculation engine, and multi-dimensional database concepts.

Note: Staffing models and roles and responsibilities may vary from project to project, but following is a recommendation based on best practices.

Technical Consultant Role

The technical consultant is usually responsible for the following key areas in addition to other activities:

- Interface work
- Batch scripting
- RPAS/RDF domain partitioning

Note: The technical consultant should also be well versed in Unix, Shell scripting, and batch schedulers.

Application / Business Consultant Role

The application/business consultant is responsible for:

- Designing and configuring alerts
- Configuring pre-processing rules
- Any workflow/workbook customizations needed to meet retailers business process needs

Note: The application consultant should have a strong understanding of RPAS configuration rule language, RPAS configuration tools, RDF plug-in, and have experience configuring solutions on RPAS.

2

Implementation Considerations

The following information needs to be considered before implementing RDF:

- Input Data
- Hardware Space Impacts
- Domain Partitioning
- Patch Considerations
- Batch Scheduling
- Security
- Internationalization

Input Data

RDF uses the following required data:

Table 2–1 Required Data Files

File	Filename	Intersection
Weekly Regular Sales	rsal.ovr	item/store/week
Weekly Promotional sales	psal.ovr	item/store/week
Weekly Clearance sales	csal.ovr	item/store/week
Daily sales (if forecasting at day level)	dpos.ovr	item/store/day
Out of stock indicator	outind.ovr	item/store/week

The following are optional files:

- Promotion history
- Format and file details as specified during implementation time
- Filename varies by promotion name and intersection

Hardware Space Impacts

The following factors can affect hardware space requirements:

- Item—Number of distinct items.
- Store—Number of physical, Web, and other distinct retail outlets.

- Calendar—Number of historical and future time periods in the domain. This impacts the overall size of the environment.
- Workbooks—Amount of space used by workbooks. This is typically greater than the domain itself. The number of workbooks is related to the number of users.

Domain Partitioning

Partitioning is done to avoid competition for resources. Building a workbook and committing data are two processes that can cause contention.

How data is partitioned has an impact on the business process. The RDF domain is defined as a global domain. For performance reasons, a single domain is not recommended. There should be an even distribution of users across a set of local domains. For example, men's merchandise could be in a domain, women's merchandise in a domain, and children's merchandise in a domain. When a user is committing data in the men's merchandise domain, this will not affect the users in the women's or children's domains because of the use of partitioning.

Consider the following questions when defining the partitioning of the domain:

- How do I partition to meet my business needs?
- How do I partition my users?
- How do I create groups of users to further partition the solution?

Note: Domain partitioning is supported only along Product hierarchy (PROD). This is a standard RPAS hierarchy. Also source levels have to be below partion dimension, that is, if using Dept for source level forecasting, you have to partition at or below Dept.

In the GA configuration, group is a dimension label. The group dimension is a regular dimension in the product hierarchy, which the customer can rename or delete.

One of the major purposes of partitioning in RDF is to facilitate the parallelization of the batch process.

The wise selection of partition intersections can significantly reduce the batch time. Partition intersection selection should also consider business needs in such a way that contention issues are minimized.

Patch Considerations

With a new release, there are two types of patches that can affect the RDF domain:

- Changes to the code in the RPAS libraries.
- The configuration is not affected by this type of patch. For these types of changes, applying the patch is a straight forward process.
- Changes to the configuration.
- These types of changes can be more complex. If a retailer has customizations in the configuration, the customizations must be redone on the new configuration before the patch is installed.

Patching Process

Before patching an RDF domain, confirm that the necessary RPAS client, server, and Configuration Tools patch updates have been successfully applied. Refer to the *Oracle Retail Predictive Application Server Installation Guide* for RPAS installation instructions.

Batch Scheduling

RDF batch is typically scheduled to run end of day/end of week with the most updated feeds of sales history and foundation data. Some tasks or batch processes can be run on adhoc or as needed basis.

Following is a list of typical RDF batch tasks and scheduling considerations:

- Daily or weekly activities:
- Hierarchy Load
- Data Load
- Pre-processing
- Forecast Generation
- Alert Generation
- Commit batch (committing workbooks saved to be committed later)
- Auto Workbook build
- Adhoc/as needed
- AutoSource

Security

To define workbook template security, the system administrator grants individual users, or user groups, access to specific workbook templates. Granting access to workbook templates provides users the ability to create, modify, save, and commit workbooks for the assigned workbook templates. Users are typically assigned to groups based on their user application (or solution) role. Users in the same group can be given access to workbook templates that belong to that group alone. Users can be assigned to more than one group and granted workbook template access without belonging to the user group that typically uses a specific workbook template. Workbook access is either denied, read-only, or full access. Read-only access allows a user to create a workbook for the template, but the user cannot edit any values or commit the workbook. The read-only workbook can be refreshed.

When users save workbooks, they assign one of three access permissions:

- World—Allow any user to open and edit the workbook.
- Group—Allow only those users in their same group to open and edit the workbooks.
- User—Allow no other users to open and edit the workbook.

Note: A user must have access to the workbook template in order to access the workbook, even if the workbook has world access rights.

For more information on security, see the *Oracle Retail Predictive Application Server Administration Guide for the Classic Client*.

Internationalization

For more information on translation for RDF, see Chapter 11, "Internationalization".

3
Integration

This chapter describes the interaction between RDF and other applications and the script used to load demand data.

RDF is integrated with the following Oracle Retail applications listed by group:

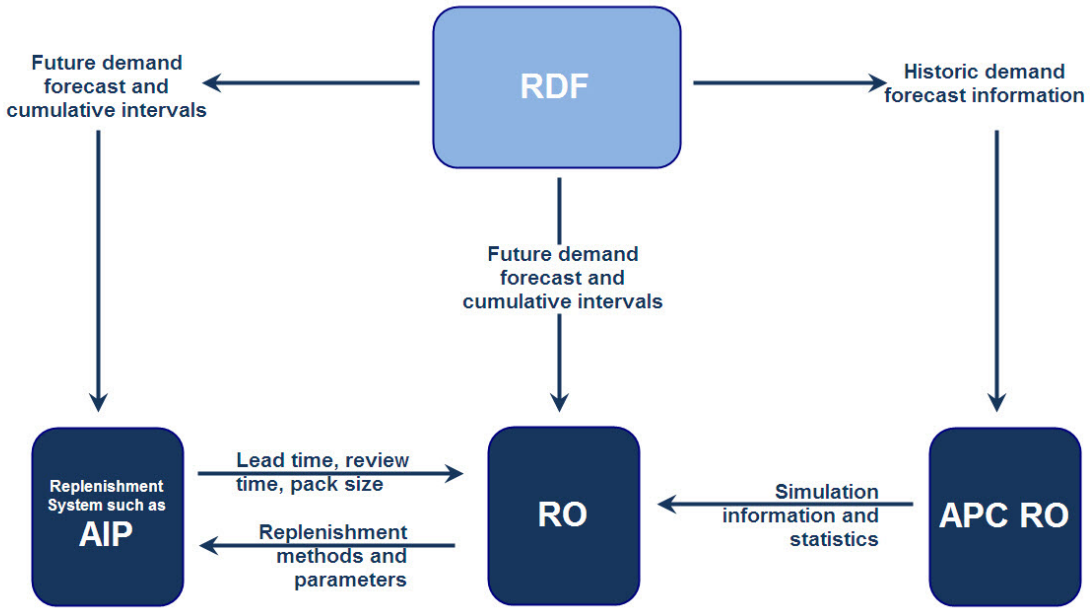
- Integrated Inventory Planning Suite Data Flow
  - Advanced Inventory Planning (AIP)
  - Analytic Parameter Calculator for Replenishment Optimization (APC-RO)
  - Replenishment Optimization (RO)
- RDF Supporting RMS Replenishment and Allocation
  - Retail Merchandising System (RMS)
- RDF Data Flow with RPO
  - Retail Price Optimization (RPO)
- APC-RPO, RPO, RDF Integration
  - Analytic Parameter Calculator for Regular Price Optimization (APC-RPO)
  - Retail Price Optimization (RPO)
- RDF to APC-RO Integration
  - Analytic Parameter Calculator for Replenishment Optimization (APC-RO)
- Integration Script
  - Integration scripts are used for moving data between applications.

Integrated Inventory Planning Suite Data Flow

Figure 3–1 shows the integration of the Integrated Inventory Planning Suite applications and the flow of data among those applications. Note that Figure 3–1 shows a replenishment system. This can be AIP or any other replenishment system. The demand forecasting application can be RDF or any other forecasting system. RDF forecasts are used as input to RO for simulation-determined replenishment parameters. RDF forecasts and associated statistics are used by AIP to plan time-phased replenishment.

This solution supports data sharing among these applications. Note that the data sharing functionality is not dependent on the presence of all these applications. The defined data sharing between any of the applications works for the entire suite as well as for a subset of the applications.

Figure 3–1 Integrated Inventory Planning Suite Data Flow



RDF Supporting RMS Replenishment and Allocation

Note: For detailed information about the RMS and RDF interface, see the following:

- Appendix A, "RPAS and RDF Integration with RMS"
- *Oracle Retail Merchandising System Operations Guide, Volume 1*

RDF integrates with Retail Merchandising System (RMS) to receive foundation data. In addition, it also sends weekly and daily forecasts to RMS (replenishment and allocation). These descriptions explain the data flows between RMS and RDF.

From RMS to RDF

The data flow from RMS to RDF includes:

- Product hierarchy
- Location hierarchy
- Calendar hierarchy

From RDF to RMS

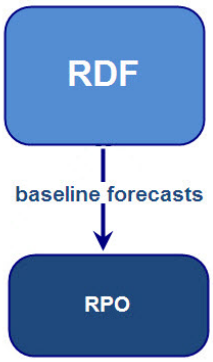
The data flow from RDF to RMS includes:

- Weekly and daily forecasts and cumulative intervals

RDF Data Flow with RPO

RDF sends baseline forecasts to RPO.

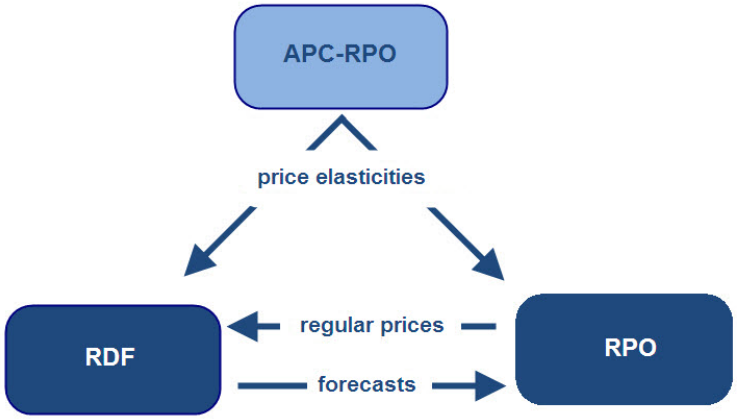
Figure 3–2 RDF Data Flow with RPO



APC-RPO, RPO, RDF Integration

This section describes the integration and data flow between APC-RPO, RPO, and RDF.

Figure 3–3 APC-RPO, RPO, RDF Integration



From APC-RPO to RPO

The data flow from APC-RPO to RPO sends:

- Item and cross item elasticities of items. RPO uses these elasticities to optimize prices.
- Maximum and minimum historical prices of items. RPO uses this data to optimize prices.
- Anchor prices of items. Anchor prices are the baseline prices that APC-RPO uses to calculate price elasticity. RPO uses the anchor prices to calculate price drift metrics.

- Maximum price increase and decrease percentages, self-item elasticity standard errors, and self-item elasticities t-statistics. RPO uses the maximum price increase and decrease percentages to setup up the default minimum and maximum price percentage change. Meanwhile, the RPO user can refer to the self-item elasticity standard error and t-statistics to adjust the price constraint.

From APC-RPO to RDF

The data flow from APC-RPO to RDF sends:

- Regular price self elasticities to RDF. The self elasticities, together with the price plan, allow RDF to calculate the item elasticity lift.
- Regular price cross-item elasticities to RDF. There are two types of cross-item elasticities: halo and cannibalization. These cross elasticities, together with the price plan, allow RDF to calculate the cross-item lift for both halo and cannibalization effects related to the corresponding elasticities.
- Anchor prices to RDF for reference purposes.
- Historical prices. RDF uses these to calculate the regular price lifts.

From RPO to RDF

The data flow from RPO to RDF sends the price plan that allows RDF to calculate the three components of the regular price lift: regular price self effect, regular price cannibalization effect, and regular price halo effect.

From RDF to RPO

The data flow from RDF to RPO sends forecasts to RPO. These forecasts represent the base demand at the item/store level. RPO aggregates the forecasts to the item/price zone level and uses that data to optimize prices.

RDF to APC-RO Integration

This section describes the integration process between RDF and APC-RO.

RDF's forecasts are important inputs for APC-RO. Forecasts are provided in two ways:

- Initial load
- Weekly updates

At initial load, a rolling 52 forecasts are generated and exported to APC-RO, each of the forecasts starts one week after another.

Secondly, as weekly updates, RDF exports currently forecast to APC-RO. The main purpose of the scripts is to generate RDF forecast in RDF GA domain and export the forecasts from a RDF domain and covert the forecasts into the format required by APC-RO.

Additionally, APC-RO provides to RDF a list of item/stores, and RDF only exports the forecasts for those item/stores to APC-RO.

Initial Load Process

The initial load process of RDF to APC-RO is as follows:

1. Loading the forecast export mask at item/store from APC-RO into a Boolean measure named apcroexptmask using loadmeasure binary.

2. Prepare the input files to generate forecasts.These two files are necessary:
   - A datelist file contains the list of desired forecast start date in the format of YYYYMMDD(for example, 20101130 and as shown in Example 3–1)
   - An input file to PreGenerateForecast binary (`pregen.xml`)

   The forecast start dates need to be seven (7) days apart as shown in Example 3–1.

Example 3–1 Datelist File

```
20100101
20100108
20100114
20100120
```

   The `pregen.xml` and `generate.xml` needs to have the right input for the desired forecast level and the desired path for the generate xml file input

3. Run `exportRDFtoAPCRO.ksh` to generate the output file for APC RO. This script performs the following steps:
   a. For each date in the dateList:

      Clear Approved Forecast and Approved Cumulative interval.

      Set forecast start date to the current date.

      Run PreGenerateForecast binary using the provided path of input file.

      For each local domain of the RDF domain: Run Generate using the provided path of input file. The provided path of inputfile needs to be the same as what specified in the provided path of input file for PreGenerateForecast.

      Call `exportRDFtoAPCROWeekly.ksh` to export the newly generated approved forecast, approved cumint and other outputs.

   b. Combine all the output created by the previous step into one file with the name specified by the `-out` argument.

Usage

exportRDFtoAPCRO.ksh

The usage of `exportRDFtoAPCRO.ksh` is as follows:

Script Name

`exportRDFtoAPCRO.ksh`

Required Arguments

The following table lists the required arguments for the script, `exportRDFtoAPCRO.ksh`.

Argument	Description
-d	Path to the master domain of a RDF domain (for example, `/user1/domain/RDF`)
-dateList	A file that contains a list of forecast start date in format YYYYMMDD (for example, `20101130`)
-PregenXML	Path to the pregeranteforecast input xml file (for example, `/user1/domain/RDF/pergen.xml`)

Argument	Description
-GenXML	Path to the generate input xml file (for example, `/user1/domain/RDF/generate.xml`, produced by the pregenerateforecast binary)
-ForecastLevel	Forecast level number in format xx, (for example, `01`, `06`)
-out	Output file name (for example: `./user1/domain/RDF/output/toapcro.csv`)

Optional Arguments

The following table lists the optional arguments for the script, `exportRDFtoAPCRO.ksh`.

Argument	Description
-DOWProfile	The optional day of week profile measure name. (for example, `apvp11`)
-mask	The mask measure name (for example, `RDFAPCROMsk01`)
-sundayIndex	Sunday's position (between 1 & 7) in the DOW hierarchy, The default is 2.
-switchOrder	Depending on your configuration, Prod hierarchy may be in front of or behind the LOC hierarchy, but in the output file the Prod ID is required to be always in front of Loc ID. So turn this switch either on or off if you see that the order in the output is incorrect. (For example, False since the default is True).

Integration Script

Integration scripts are used for moving data between applications. The following rules apply to integration scripts:

- The exportMeasure utility is used to export data in CSV (comma-separated values) format. This maintains the consistency of start and width attributes across different applications.
- Data exported from the source application is placed in the destination domain input directory.
- Export scripts must run before load scripts. They should be run in the batch window.
- The scripts have a command line argument to set the maximum number of processes that need to be run in parallel. Setting this argument can help speed up the performance of independent tasks on local domains. The default is 1.
- Do not hard-code domain paths. The paths are entered as command-line arguments.

Table 3–1 lists the integration scripts for RDF.

Table 3–1 Integration Script

Application	Script Name
AIP	rdf_e_aip_appf.ksh rdf_e_aip_cumint.ksh
RMS	rdf_e_rms.ksh

Usage

rdf_e_aip_appf.ksh

The usage of `rdf_e_aip_appf.ksh` is as follows:

Script Name

`rdf_e_aip_appf.ksh`

Required Arguments

The following table lists the required arguments for the script, `rdf_e_aip_appf.ksh`.

Argument	Description
-l	Forecast Level (01, 02, and so on)
-o	Output file name

Optional Arguments

The following table lists the optional arguments for the script, `rdf_e_aip_appf.ksh`.

Argument	Description
-d	Path to the current domain (for example, /user1/domain/RDF). If not specified, defaults to the current directory.
-f	Forecast Start Date, in the format YYYYMMDD. If not specified, defaults to the current system date.
-n	Number of time periods to include in the output. If not specified, defaults to the forecast length.
-s	Store column prefix.
-c	Calendar column prefix.

rdf_e_aip_cumint.ksh

The usage of `rdf_e_aip_cumint.ksh` is as follows:

Script Name

`rdf_e_aip_cumint.ksh`

Required Arguments

The following table lists the required arguments for the script, `rdf_e_aip_cumint.ksh`.

Argument	Description
-l	Forecast Level (01, 02, and so on)
-o	Output file name

Optional Arguments

The following table lists the optional arguments for the script, `rdf_e_aip_cumint.ksh`.

Argument	Description
-d	Path to the current domain (for example, /user1/domain/RDF). If not specified, defaults to the current directory.
-n	Number of time periods to include in the output. This is currently set to 1 in the script in all cases regardless of the value passed in.
-s	Forecast Start Date, in the format YYYYMMDD. If not specified, defaults to the current system date..
-p	Store column prefix.

4

Installation Consideration

This chapter describes the setup that must be done before building the RDF domain and the batch script that must be run to build the domain.

Note: For information on building the Curve and Grade domains, refer to the *Oracle Retail Predictive Application Server Installation Guide*.

Installation Dependencies

RPAS and RDF must be installed before setting up and configuring RDF:

- For information on installing RPAS, see the *Oracle Retail Predictive Application Server Installation Guide*.
- For information on installing RDF, see the *Oracle Retail Demand Forecasting Installation Guide*.

RPAS Installation

The Java-based RPAS installation programs that are included with the installation package are used to install the server-side RPAS components on UNIX operating systems.

The RPAS installer performs the following functions:

- Installs the RPAS server
- Installs the Configuration Tools on the server
  - On Windows, an InstallShield package is used to install the Configuration Tools.
- Defines the DomainDaemon port

RPAS Client Installation

The RPAS server installation package also includes the following RPAS clients:

- RPAS Classic Client: A Windows-based client interface for end users and system administrators of an RPAS domain.
- RPAS Fusion Client: A Web-based client developed using Oracle Application Development Framework (ADF).

Each RPAS client installation package includes a separate installer to help you install the client. For more information on installing the RPAS clients, refer to the *Oracle Retail Predictive Application Server Installation Guide*.

RDF Installation

In addition to the RPAS installer, the installation package also includes the Java-based RPAS installation program for the RDF application.

The RDF installer automates the following tasks:

- Installs the RDF mock install configuration
- Installs RDF plug-ins for the Configuration Tools
- Installs Language Translation files
- Creates a sample RDF domain

RDF Taskflow for the RPAS Fusion Client

The RDF installation software enables you to install the taskflow and online help files for the RPAS Fusion Client. In order to install the taskflow files, the RPAS Fusion Client must already be installed. For more information on installing the RPAS Fusion Client, see the *Oracle Retail Predictive Application Server Installation Guide*.

During the RPAS Fusion Client installation, the installer automatically sets up the RPAS domain connection configurations in the `ProfileList.xml` file. If you choose to set up the domain connection after the installation or set up an additional domain, you must manually set up the connection. For more information, see the *Oracle Retail Predictive Application Server Administration Guide for the Fusion Client*.

Environment Variables

In addition to the regular RPAS environment variables, including RPAS_HOME, you need to set up the following environment variables and export them.

Table 4–1 Environment Variables

Environment Variable	Description	Use
RIDE_HOME	The location of the configuration tools.	Mandatory for Installation and Patching
PATH	The standard operating system path. Running the script `$RPAS_HOME/rpaslogin.ksh` is the preferred method to update and export the PATH variable.	Mandatory
RDF_BATCH_TIMEOUT	This numeric value, set in seconds, controls the amount of time that the Batch Forecast workbook will wait for another Batch Forecast workbook process to finish. This allows several users in different local domains to start a Batch Forecast workbook simultaneously.	Optional

Files Needed to Build the RDF Domain

Before the domain is built, the following types of files need to be set up:

- Standard RPAS Hierarchy Files
- Required Data Files
- Optional Data Files

Standard RPAS Hierarchy Files

The following hierarchy files are needed:

- Calendar (CLND) Hierarchy File
- Product (PROD) Hierarchy File)
- Location (LOC) Hierarchy File
- Relative Calendar (RLTV) Hierarchy File

Note: As with all standard RPAS hierarchies, these hierarchies are configurable. For information about configuring these hierarchies, see Chapter 5, "Configuration Considerations".

Calendar (CLND) Hierarchy File

File name: clnd.csv.dat
File format: comma-separated values file

Table 4–2 describes the fields in the file:

Table 4–2 Calendar Hierarchy Fields

Field	Description
Day	Day ID
Day label	Day label
Week	Week ID
Week label	Week label
Mnth	Month ID
Mnth label	Month label
Qrtr	Quarter ID
Qrtr label	Quarter label
Fiscal Half	Fiscal Half ID
Fiscal Half label	Fiscal Half label
Year	Year ID
Year label	Year label
Day of Week	Day of Week ID
Day of Week label	Day of Week label
Day of Season	Day of Season ID
Day of Season label	Day of Season label
Week of Year	Week of Year ID
Week of Year label	Week of Year label
Week of Season	Week of Season ID
Week of Season label	Week of Season label

Example 4–1 CLND Format

```
20050130,01/30/2005,w01_2005,01/30/2005,JAN_2005,January 2005,Q1_
2005,Quarter 1 2005,H1_2005,2005 First Half, A2005, Year 2005, SAT,
Saturday,DOS01,DOS 01,WY01, Week 01, WS01,WOS 01
```

Note: After upgrading a domain or loading a new calendar hierarchy into the RDF domain, ensure that the current date measures value is within the new calendar hierarchy.

Product (PROD) Hierarchy File

File name: prod.csv.dat
File format: comma-separated values file

Table 4–3 describes the fields in the file:

Table 4–3 Product Hierarchy Fields

Field	Description
Item	Item ID
Item label	Item label
Parent	Parent ID
Parent label	Parent label
Grand Parent	Grand Parent ID
Grand Parent label	Grand Parent label
Subclass	Subclass ID
Subclass label	Subclass label
Clss	Class ID
Clss label	Class label
Dept	Department ID
Dept label	Department label
Group	Group
Group label	Group label
Division	Division ID
Division label	Division label
Supplier	Supplier ID
Supplier Label	Supplier label
Diff 1	Diff 1 ID
Diff 1 label	Diff 1 label
Parent Diff 1	Parent Diff 1ID
Parent Diff 1 label	Parent Diff 1 label
Grand Parent Diff 1	Grand Parent Diff 1 ID
Grand Parent Diff 1 label	Grand Parent Diff 1 label
Subclass Diff 1	Subclass Diff 1 ID
Subclass Diff 1 label	Subclass Diff 1 label
Class Diff 1	Class Diff 1 ID
Class Diff 1 label	Class Diff 1 label
Dept Diff 1	Dept Diff 1 ID
Dept Diff 1 label	Dept Diff 1 label

Example 4–2 PROD Format

```
10000010,10000010Leather Loafer - Black 6 B, 10000010, 10000010Leather
Loafer - Black 6 B, 10000009, 10000009Leather Loafer, 122, 122Loafer,
1312, 1312Casual, 1310, 1310Footwear Women's, 1300, Group 1, 1, All
Product, 1000, Supplier 1, 10000010_sml, 10000010Leather Loafer - Black 6
B Small, 10000009_sml, 10000009Leather Loafer Small, 122_sml, 122Loafer
Small, 1312_sml,,  1312Casual* Small, 1310_sml,,  1310Footwear Women's*
Small, _sml, Small
```

Location (LOC) Hierarchy File

File name: loc.csv.dat
File format: comma-separated values file

Table 4–4 describes the fields in the file:

Table 4–4 Location Hierarchy Fields

Field	Description
Str	Store ID
Str label	Store label
District	District ID
District label	District label
Regn	Region ID
Regn label	Region label
Area	Area ID
Area label	Area label
Chnl	Chain ID
Chnl label	Chain label
Company	Company ID
Company label	Company label
Store Format	Store Format ID
Store Format label	Store Format label
Store Class	Store Class
Store Class Label	Store Class Label

Example 4–3 LOC Format

```
1000, New York City, 1000, US, 1000, North America, 1000, The Americas,
1000, Bricks & Mortar, 100, JCB Trading Company, 4, 4, A, A
```

Relative Calendar (RLTV) Hierarchy File

File name: rltv.dat
File format: fixed width file

> **Note:** You should have 53 weeks set for this file.

Table 4–5 describes the fields in the file:

Table 4–5 Relative Calendar Hierarchy Fields

Field	Description
Relative Week	Numbered relative week
Relative Week Label	Relative week label
Relative Year	Numbered relative year
Relative Year Label	Relative Year Label

Example 4–4 RLTV Format

```
01 Relative Week 01RY_01 Relative Year
```

Required Data Files

The following data files are required:

- Sales history
- Promotion history (if using RDF causal)

File format: comma-separated values file

Table 4–6 lists examples of the required data files.

Table 4–6 Examples of Required Data Files

Required Data File	Example
Sales history	20090311,10000044,1000,8 20090415,10000044,1000,5
Promotion history (if using RDF causal)	20090311,10000044,1000,1 20090415,10000044,1000,1

Optional Data Files

The following data files are optional:

- Assortment Multipliers (if using Demand Transference Effect)
- Base Rate of Sales
- Forecast Start Date Override (if using Demand Transference Effect)
- Like items
- Out of Stock Indicators
- Price Elasticity (if using Regular Price Effect)
- Promo Cannibalization Change Ratio Source (if using Cross Promotion Effect)
- Promo Cannibalization Max Change Ratio Source if using Cross Promotion Effect)
- Promo Halo Max Change Ratio Source (if using Cross Promotion Effect)
- Promo Halo Ratio Source (if using Cross Promotion Effect)
- Promo Halo Prod to PROR Mapping (if using Cross Promotion Effect)
- Regular Price Prod to PROR Mapping (if using Regular Price Effect)
- Sister stores

File format: comma-separated values file

Example 4–5 Out of Stock Indicators Data File

```
20090311,10000044,1000,1
20090415,10000044,1000,1
```

Table 4–7 Examples of Optional Data Files

Optional Data File	Example
Promo Halo Prod to PROR Mapping (if using cross promo effect) File name: prod2pror01xb.csv.ovr Intersection: Halo Level + RHS Halo Level Type: Boolean measure	Assumption: Halo level is class 1410,1410,true 2410,2410,true 1114,1114,true 1141,1141,true 1231,1231,true 1312,1312,true 1322,1322,true
Approved Promo Halo Effect (if using cross promotion effect) File name: apphaloeff.csv.ovr Intersection: PROD_LOC_PROR (output from CPEM. Intersection decided by CPEM configuration) Type: Real	1047,1000,2891,0.14036448351423 1048,1000,2891,0.23036448351423 1047,2100,2891,0.13143344461036
Approved Promo Cannibalization Effect (if using cross promotion effect) File name: appcanneff.csv.ovr Intersection: PROD_LOC_PROR (output from CPEM. Intersection decided by CPEM configuration) Type: Real	401,1100,1,-0.19122357822331 401,2000,1,-0.18625095077102 111,1100,112,-1.3169973676138 1,1100,401,-0.38277054297883 1,2000,401,-0.34202226867616
Aggregated Promotional Lift Override (if using promotion effect and overriding promotion lift) File name: aprmliftovr01xb Intersection: user defined promotional lift override aggregate level (set to final level by default) Type: Real	w03_2002,10000494,1000,30 w05_2002,10000494,1000,40 w08_2002,10000494,1000,35 w16_2002,10000494,1000,60 w18_2002,10000494,1000,45 Note: Data intersection should match the defined override aggregate level.
Regular Price History (if using regular price effect) File name: regprice01xb Intersection: CLND_PROD_LOC (the same as final level) Type: Real	w31_2001,10000359,1000,3.99 w32_2001,10000359,1000,3.99 w33_2001,10000359,1000,4.59 w34_2001,10000359,1000,4.59
Price Lift Override (if using regular price effect and using price lift override) File name: prclifovr01xb Intersection: CLND_PROD_LOC (the same as final level) Type: Real	w03_2002,10000494,1000,5 w05_2002,10000494,1000,5 w08_2002,10000494,1000,-2 w16_2002,10000494,1000,-3 w18_2002,10000494,1000,5
Regular Price Prod to PROR Mapping (if using regular price effect) File name: prcprod2pror01xb.csv.ovr Intersection: ITEM_ITER Type: Boolean measure	10000328,10000328,true 10000536,10000536,true 10000537,10000537,true 10000539,10000539,true 10000629,10000629,true

Table 4–7 (Cont.) Examples of Optional Data Files

Optional Data File	Example
Price Elasticity (if using regular price effect) File name: rdfgamma.csv.ovr Intersection: PROD_LOC_PROR (output from RPO. Intersection decided by RPO configuration) Type: Real	10000359,1000,10000359,-0.45 10000360,1000,10000359,0.1 10000449,1000,10000359,0.15 10000452,1000,10000359,0.18 10000360,1000,10000360,-0.44
Assortment Multipliers (if using demand transference effect) File name: assmul01xb.csv.ovr Intersection: PROD_LOC_CLND (the same as final level forecast intersection) Type: Real	20010803,10000359,1000,1.2 20010921,10000359,1000,0.8 20010803,10000449,1000,0.8 20010921,10000449,1000,1.2 20010803,10000533,1000,0.9
Forecast Start Date Override (if using demand transference effect) File name: fmafstdt01xb.csv.ovr Intersection: PROD_LOC (the same as final level forecast method intersection) Type: Date	10000666,11005,20011005 10000666,11006,20010803 10000666,11007,20010907 10000666,11009,20011102
Base Rate of Sales File name: brsls01xb.csv.ovr Intersection: PROD_LOC (the same as final level forecast method intersection) Type: Real	10000666,11005,35 10000666,11006,30 10000666,11007,55 10000666,11009,40

Output from RDF to RMS and Retail Analytics

- Weekly forecasts and cumulative intervals
- Item/Store/Week
- Daily forecasts and cumulative intervals
- Item/Store/Day

5

Configuration Considerations

RDF is a statistical forecasting solution that uses state-of-the-art modeling techniques to produce high quality forecasts with minimal human intervention. Forecasts produced by RDF enhance the retailer's supply-chain planning, allocation, and replenishment processes, which enables a profitable and customer-oriented approach to predicting and meeting product demand.

Forecast information is often required for items at the lowest levels in a hierarchy. Problems can arise when historic sales data for these items is too sparse and too noisy to identify clear selling patterns. In such cases, generating a reliable forecast requires aggregating sales data from a low level up to a higher level in the hierarchy. After a forecast is generated at the higher level, the resulting data can be allocated (spread) back down to the lower level. This is based on the lower level's relationship to the total. Before you can spread forecast data back down to a lower level, you should have an understanding of the relationship between the lower level and the higher level dimensions. Frequently, an additional forecast is generated at the low level to help determine this relationship. This low level is called the final forecast level. Forecast data at this level might be sufficient to generate reliable percentage-to-whole information, but the actual forecast numbers are more robust when they are generated at an aggregate level. This aggregate level from which forecast data is spread is referred to as the source forecast level.

Some high-volume items may possess sufficient sales data for robust forecast calculations directly at the final forecast level. In these cases, forecast data that is generated at an aggregate level and then spread down to lower levels can be compared to forecasts that are run directly at the low level. Comparing the two forecasts, each generated at a different hierarchy level, can be an invaluable forecast performance evaluation tool.

The RDF solution may include multiple final forecast levels. Forecast data must appear at some final level for the data to be approved and exported to other systems.

Using the RDF plug-In, final and source forecast levels are defined for the RDF solution.

Note: The ability to configure the RDF solution may be limited. This is based on your licensing agreement.

Forecasting Calendar Hierarchy Requirement

With any RDF solution, configuration of the calendar hierarchy must always include a day dimension level name. There are no configuration requirements for the dimensions of the merchandise or location hierarchies.

Forecasting Limitations Using the Partition Hierarchy

Any dimension along the partition hierarchy that is used as an intersection to forecast must be unique across all domains. This requirement especially applies to Alternate Hierarchies. For example, if the forecast level is `supplier\str\week`, the Supplier dimension cannot have a supplier position that exists in multiple domains. However, additional support for clean partitioning of Alternate Hierarchies is provided through the RDF Transformation programs used to integrate RMS foundation data for RDF.

Causal Forecasting at Source Levels

RDF expects all promotional history to be pre-aggregated (externally or through custom RPAS rules) to each source level, when running causal forecasting at that level. It is possible to enable/disable promotions for each causal forecast level within the application.

Loc Hierarchy Limitation

RDF expects that the location hierarchy is called loc.

Forecasting Pre-Configuration Data Requirements

There are several parameters within the RDF configuration that may reference other measures that are configured external to the solution, specifically:

- Source Data
- Seasonal Profile
- Plan Data
- Spreading Profile

Prior to configuring an RDF solution, it is required that these measures already exist within the Project.

Source Data

The RDF plug-in populates a pick-list with all non-Boolean and non-string measures that have been created in the Project.

Plan Data

If the Plan Data that will be used to support Bayesian forecasting is being defined within another solution, this measure should already exist. The entry of this parameter is not required within the configuration, and it can be entered in the resulting domains.

Spreading Profiles and Seasonal Profiles

If Curve is used to produce Spreading Profiles or Seasonal Profiles to support your Forecasting solution, these profiles should already have been configured in the Curve solution. If these profiles are being defined external to Curve, these measures should already exist within the Project.

Registering the RdfFunctions Library

Prior to configuring the RDF Solution, register the RdfFunctions library to support proper validation of the RDF-specific rules:

Open the Function Library Manager and add RdfFunctions.

Note: If Promote is implemented, the following rules display as invalid; however these should be ignored:

- Rule: PREF_PlHolder
- RuleGroup: PREF_place
- Rule Group: PRMA_place
- Rule Group: PRPL_place

Editing Forecast Level Parameters

Note: For more information on Source Level Forecasting, see the *Oracle Retail Demand Forecasting User Guide*.

Table 5–1 lists and describes the forecast level parameters:

Table 5–1 Forecast Level Parameters

Forecast Level Parameters	Description
Default Source Level	Assigned only at the Final level, the Default Source Level is the primary level at which the aggregate, more robust forecast is run. The desired Source Level must first be created within the RDF configuration for it to be a selection in the pick-list. For more information on Source Level Forecasting, refer to the *Oracle Retail Demand Forecasting User Guide.* If no source level is required, the final level should be selected.
Enable Demand Transference	Assigned only at the final level. This is to indicate if demand transference is enabled on the level.
Enabled Demand Transference	Assigned only at the final level, the Enabled Demand Transference window defines if demand transference is enabled. If the check box is not enabled, demand transference function will not be applied to generate the forecast.
Extra Week Indicator	Assigned on both source level and final levels. The extra week indicator field should be populated with a measure name. This boolean measure indicates which week is the 53rd week. If the field is empty, then the 53 week processing is unavailable for that level.
Forecast Alert Intermediate Intersection	Assigned on the final level only. This field holds an intersection string that indicates the alert parameters' intermediate intersection. The alert parameters can be specified on three intersections: default, intermediate and override. When this field is empty, all GA alerts in the level are unavailable.
Forecast Method	The Forecast Method window displays all forecast generation methods that may be defined for a forecast level. The Default Forecast Method is also determined here. See "Forecast Methods" for a list of Forecast Methods that may be selected. See the *Oracle Retail Demand Forecasting User Guide* for more information on each method.
Intersection	The Intersection is the hierarchy dimensions that define the forecasting level.
Level Label	The Level Label is the level description that is viewed by the user once the domain is created. See "Level Labels" for additional information.
Level Name	The Level Name is the system-assigned level number when a forecast level is created. This is a read-only parameter.

Table 5–1 (Cont.) Forecast Level Parameters

Forecast Level Parameters	Description
Periodicity	Periodicity is the number of periods within the Calendar dimension, which are defined in the forecast level intersection. For example, if an intersection is defined at `week/item/store` the Periodicity value is 52 (since there are 52 weeks within a year).
Plan Data	Assigned only at the final level, Plan Data (sales plans) provide details of the anticipated shape and scale of an item's selling pattern. This information is required when Bayesian forecasting is used as a Forecast Method. The value in this parameter is a measure name.
Price Elasticity Data	Assigned only at the final level, the Price Elasticity Data specifies the measure to be used as the input data (for example, rdfgamma) to calculate the regular price change lifts. The value in this parameter is a measure name. For more information on the measure, refer to the Optional Data Files.
Promo Cannibalization Change Ratio Source	This measure is specified at an intersection higher than item/store. The content of the measure is visible if you roll up to All Products on the product hierarchy. At a lower intersection the cell appears as a hash mark. This parameter contains the name of the measure that determines the percentage of the promotional lift that is going to cannibalize related items.
Promo Cannibalization Data	Assigned only at the final level, the Promo Cannibalization Data specifies the measure to be used as the input data (for example, appcanneff) to calculate the cannibalization promotion lifts. The value in this parameter is a measure name. For more information on the measure, refer to the Optional Data Files.
Promo Cannibalization Max Change Ratio Source	This measure is specified at an intersection higher than item/store. The content of the measure is visible if you roll up to All Products on the product hierarchy. At a lower intersection the cell appears as a hash mark. This parameter contains the name of the measure that determines an item's maximum allowed drop in sales due to cannibalization. For instance if the sales of an item for a given period are 20 units, and the maximum allowed percentage is 20%, the drop in sales due to cannibalization for the period can not exceed 4 units.
Promo Halo Change Ratio Source	This measure is specified at an intersection higher than item/store. The content of the measure is visible if you roll up to All Products on the product hierarchy. At a lower intersection the cell appears as a hash mark. This parameter contains the name of the measure that determines the percentage of the promotional lift that is going to increase demand of complimentary items due to halo effect.
Promo Halo Data	Assigned only at the final level, the Promo Halo Data specifies the measure to be used as the input data (for example, apphaloeff) to calculate the halo promotion lifts. The value in this parameter is a measure name. For more information on the measure, refer to the Optional Data Files.
Promo Halo Max Change Ratio Source	This measure is specified at an intersection higher than item/store. The content of the measure is visible if you roll up to All Products on the product hierarchy. At a lower intersection the cell appears as a hash mark. This parameter contains the name of the measure that determines an item's maximum allowed increase in sales due to halo. For instance if the sales of an item for a given period are 15 units, and the maximum allowed percentage is 20%, the increase in sales due to halo for the period can not exceed 3 units.
Promo Lift Override Intersection	Assigned only at the final level, the Promo Lift Override Intersection window displays the intersection level allowed to override the promotion lift.
Return Forecast Parameters Intermediate Intersection	Assigned on the final level. This field should be populated with an intersection string. It is used to indicate the intermediate level of return forecast parameters. The return forecast parameters can be specified on three levels, default, intermediate and override. If this field is empty, then return forecast functionality is unavailable for this level.

Table 5–1 (Cont.) Forecast Level Parameters

Forecast Level Parameters	Description
Seasonal Profile	A seasonal profile provides details of the anticipated seasonality of an item's selling pattern. The seasonal profile is required in conjunction with the Profile-based Forecast Method. The seasonal profile can be generated or loaded, depending on your configuration. The value in this parameter is a measure name.
Source Data	Assigned only at the Final level, the Source Data is the measure to be used as the input data (for example, POS) for the generation of forecasts. The values in this pick-list are populated with all non-string and non-Boolean type measures that are configured in the Project.
Spreading Profile	Assigned only at the source forecasting level, the Spreading Profile is used to spread source level forecasts down to the final forecast level. The value in this parameter is a measure name, a profile level name, or any combination of these separated by commas. ■ If Curve is used to dynamically generate the spreading ratios, this parameter should be populated with the final profile level name (profile number) configured. For example: 01 (this is profile level 01). ■ If Curve is used to generate the static (manually approved) spreading ratios, this parameter should be populated with the Approved Profile measure. For example: `apvp11` (this is the Approved Profile for Curve level 11).

Level Labels

The Level Label is the level description that is viewed by the user once the domain is created.

- Level Labels may not exceed 40 characters.
- It is recommended, but not required, that Level Labels include the Level Name (the system-assigned level number).

  Within the Forecast Administration workbook, the Default Source Level may be edited. This pick-list is populated with the Level Name for all levels that are associated with a final level. Since this value can also be specified within this configuration, this recommendation may not be necessary if changes to the Default Source Level are not expected within the application.
- RPAS automatically places parentheses **( )** around Forecast Level Labels.

  The configuration specialist should not include these in the level label configuration or the installer will fail.

Correct Example	Incorrect Example
`1- item/str/week - Final`	`(1:itm/str/week - Final)`

- A hyphen (**-**) should not be used before or after the Forecast Level Label.

Correct Example	Incorrect Example
`1-itm/str/week - Final`	`-1:itm/str/week - Final-`

- A colon (:) should not be used at all in the Level Label.

Incorrect Example
`1: itm/str/week-`

Forecast Methods

The following is a list of Forecast Methods that may be selected. See the *Oracle Retail Demand Forecasting User Guide* for more information on each method.

- No Forecast
- Average
- Simple
- Intermittent
- Simple/Intermittent
- Trend
- Additive Seasonal
- Multiplicative Seasonal
- Seasonal
- AutoES
- Causal

> **Note:** See "About Causal Forecasting" for special conditions for Causal methods.

- Bayesian
- Profile-based
- LoadPlan
- Copy
- Components

About Causal Forecasting

The Causal method should be selected as a valid method only for levels in which causal forecasting will be used.

When enabling Causal as a valid forecast method for a source level, note that RDF Promotion variables need to be provided at the same dimension along the product and location hierarchies as the forecast level for which Causal forecasting is run (Final or Source). RDF Causal does not support aggregation of promotion variables along any hierarchies other than Calendar (Clnd). Aggregation of promotion variables along either or both of product or location hierarchies needs to be handled externally through configuration. Aggregation along the calendar hierarchy is support by RDF Causal, using specified aggregation and spread profiles. Refer to the *Oracle Retail Demand Forecasting User Guide* for details.

Floating Annual Events and Holidays

In an RDF causal implementation, an annual event that does not occur in the same period every year, but has a spike in demand associated with it, is handled by associating a causal factor to it. The system then determines the associated lift, and applies it to the relevant point in time in the forecast horizon.

In an RDF implementation, where there is no promote module, these events still exist. For instance, an increase for sales leading to Easter happens even if there is no

promotion specifically associated with Easter (chocolate bunnies sell more during Easter even though they are not specifically promoted). And because it does not happen in the same time period every year, the Easter spike appears randomly any time in March or April. The spike is baked in the seasonality of each item/store, making its prediction; timing and magnitude, inaccurate at best. RDF can support floating events and holidays, not only through its causal capabilities, but also for baseline forecasting. Refer to Generating Forecast for Floating Annual Events and Holidays for implementation details.

Generating an Internal Baseline for the Causal Forecast Method

RDF does not generate an internal baseline for the Causal Forecast method. A Causal External Baseline need to be specified. Two phase forecasting is recommended:

- Calculating Float Event Effect
- Component Forecasting.

Baseline Forecasting

Perform the following steps for the first phase of generating an internal baseline for the Causal Forecast method.

1. Data cleansing is a critical step for obtaining a good forecast. For baseline forecasting, at least three preprocessing stagings are needed:
   - Correction for out of stock effects
   - Removal of Outlier
   - Removal of promotional spike

   Note: All of these can be achieved using event based StdES. Out of stock indicator, outlier indicator and promotion indicator are required.

2. A final level needs to be configured in RDF to generate baseline forecast. The final level needs to be on the same intersection as the causal forecasting. One of more source levels can be set up for the final level based on data characteristics. The clean data from Step 1 is used as data source for both final level and source levels. The interim forecast shall be generated using simple method. The source level forecast can be generated using AutoES and then spread down to final level using interim forecast as ratio.

Causal Forecasting

Perform the following steps for the second phase of generating an internal baseline for the Causal Forecast method

1. For causal forecasting, at least three preprocessing stagings are needed:
   - Correction for out of stock effects
   - Removal of outliers
   - Removal of seasonal effects

   Note: The first two stages can be achieved using event based StdES. Out of stock indicator and outlier indicator are required here. The third stage is achieved using pregenerated seasonal indices.

2. A forecast level (final or source) can be set up to run causal. The forecast intersection needs to be on the same intersection as your baseline forecasting final level. The approved forecast generated from Calculating Float Event Effect is used as external causal baseline measure for this level. The clean data from Step 1 is used as a data source for causal forecasting.

Generating Forecast for Floating Annual Events and Holidays

There are two phases to generate a forecast for floating annual events and holidays:

- Calculating Float Event Effect
- Component Forecasting.

Calculating Float Event Effect

Perform the following steps for the first phase of generating a forecast for floating annual events and holidays.

1. Set float event indicator in the Floating Event Administration workbook, Floating Event Calendar Maintenance worksheet. Refer to the *Oracle Retail Demand Forecasting User Guide* for more information.
2. Run `gen_floatlift.ksh` batch from the master domain,

Note: There are 20 pre-configured float events. If you need more, add corresponding measures into the PreProcess solution.

The Float Event indicator must be set as Boolean type and configured at the same intersection as the forecast level in next step. The rules in the FloatEvent rule set also need to be updated.

Component Forecasting

Perform the following steps for the second phase of generating a forecast for floating annual events and holidays.

1. A forecast level (final or source) needs to be set up to run a Component forecast.Refer to the *Oracle Retail Demand Forecasting User Guide* for more information about the component forecast method.
2. Set the Components, Promotional Lifts as `precmbevnlift` which is the float event effect calculated in the first phase, Calculating Float Event Effect.

Autogenerating Hierarchies, Measures, Rules and Workbook Templates

The following is the process to autogenerate the hierarchies, measures, rules, and workbook templates that are required by RDF to support the forecasting configuration entered in the RDF plug-in.

The system automatically generates the following:

Item	Description
Hierarchies	The DATA hierarchy is updated with the flvl, fbrt and fmtr dimensions.
Measures	All measures necessary to support the base RDF solution are created.

Item	Description
Rules	All Rule Sets, Rule Groups, and Rules to support the base RDF solution are created.
Workbook Templates	All pre-defined workbook templates to support the base RDF solution are created.

You may continue to make changes to the RDF plug-in configuration, and the autogeneration process may be repeated as often as needed prior to the installation.

Deleting a Forecast Level

Deleting a forecast level causes the system-assigned enumerated values in the Level Name to renumber such that levels are in consecutive order starting with forecast level 01. Deleting a forecast level may impact any solution configuration that uses a specific level.

If the domain using the configuration has previously been installed, there is potential to lose data associated to a level that has been deleted or renumbered.

Configuring the Cloning Administration Workbook

The Product/Location Cloning Administration workbook allows users to specify clone products by a configurable dimension in the location hierarchy and clone stores by a configurable dimension in the product hierarchy. For example, users can specify a different clone item for a different region.

Editing the RDF GA Configuration

The autogeneration process creates hierarchies, measures, rules, and workbook templates that are required to support the essential RDF functionality. This base configuration is referred to as the GA Configuration. Certain changes to the GA Configuration are allowed. Once edits to the GA Configuration are made and the autogeneration process occurs again, valid changes to the configuration are preserved. There is nothing in the RPAS Configuration Tools to prevent invalid changes from being made.

> **Note:** When a custom taskflow exists in the configuration, the plug-in automation may cause the workbook template field to be blank. If this occurs, save the configuration and then re-open the configuration and the workbook template field is populated.

Table 5–2 outlines acceptable changes and restrictions.

Table 5–2 Editable Configuration

Editable Item	Description
RDF Solution Extension Name	The name assigned to the resulting RDF solution after autogeneration occurs cannot be edited.
Major and Minor Components	Additional major and minor components may be added to the RDF GA Configuration. The major and minor components that are part of the GA Configuration may not be edited. This restriction also applies to measure names and measure labels.

Table 5–2 (Cont.) Editable Configuration

Editable Item	Description
Rules	Additional Rule Sets, Rule Groups, and Rules may be added to the RDF GA Configuration. This includes support for adding new rules to existing GA configuration rule groups. It is recommended that new rules added to the GA configuration rule groups include cust (represents Custom) in the rule name. This allows for easy identification of Rules that are not part of the GA Configuration. Rule sets, rule groups, and rules that are part of the GA Configuration may not be renamed. Existing rules that are part of the GA Configuration may not be modified in any way.
Workbook Templates	Additional workbook templates may be added to the RDF GA Configuration. New measures and rules may also be added to the GA configuration workbook templates. This is done by adding new major and minor components, and adding new Rules to existing rule groups in the GA Configuration.

RDF Non-modifiable Hierarchies

The names of the hierarchies in this section cannot be changed.

Table 5–3 Non-Modified Hierarchies

Hierarchy Name	Hierarchy Label
DATA	Data Hierarchy
CLSH	Cluster
GRCH	Grade configurations
PRMH	Promotions
CSLH	Causal levels
OPMO	Overlapping Promotions

Calendar (CLND)

The Calendar hierarchy represents time in all RPAS solutions. It is a required hierarchy and must have a dimension named day (DAY).

Product (PROD)

The Product or Merchandise hierarchy represents the retailer’s merchandise (that is, merchandise that the retailer retails through its retail channels).

Location (LOC)

The Location hierarchy represents the retailer's retail locations and their roll-ups.

Configuring Causal for Short Lifecycle Merchandise

The Causal method in RDF is proven to work well for items with long lifecycle. However, the promotional forecasting generation process can be improved when forecasting items with short lifecycle or very pronounced seasonality patterns.

RDF creates a preseason baseline, which is adjusted in season using Bayesian and then promotion effects are applied on top of the baseline.

Procedure to Configure Causal for Short Lifecycle Merchandise

The following functionality is implemented in RDF's Common solution and Common rule set. The rule groups are:

- SLCHistBaseline
- SLCPrep_ROS

The measures are defined Common solution within the component called CausalSLC.

Perform the following procedure to implement the causal short life cycle functionality:

1. Calculate the Baseline:
   a. Deprice promotional sales using the deprice filter in preprocessing.
   b. Depromote the resulting output using the STD ES filter in preprocessing.
   c. Calculate a rate of sale for the depriced and depromoted time series.
   d. Aggregate the depriced and depromoted time series to a higher intersection; the result is a lifecycle curve.
   e. Spread the lifecycle curve calculated at the higher intersection to item/store using the rate of sale.

   The resulting curve is stretched/compressed and time shifted to fit the time frame of the new season. At this point you have generated the Bayesian plan, that is used in season to create the forecast baseline.
2. Causal Estimation
   a. Remove the seasonality bias:

      Combine the promotional lifts from Step 1b and the rate of sale from Step 1c to create the causal source that is processed by causal.

      Given the causal source and the promotional calendar, causal estimates the promotional lifts.
   b. If desired, the promotional lifts can be further refined and processed to make them more robust.

      The lifts calculated for every prod/loc are weighted and aggregated to the causal higher intersection.
3. Generating a final forecast
   a. Combine the baseline and promotional lifts calculated in Steps 1 and 2, respectively to generate the final forecast.
4. Configuring Daily Causal
   a. Use the daily final level.
   b. Set Causal Calculation Intersection and Causal Data Source at week level.
   c. Causal External baseline is provided at week level.
   d. Promotions can be provided at day or week level.
   e. Provide a week to day profile (based on DOW) to spread external baseline to day level prior to system forecast output. If no spread profile is provided, the default of 1/7 is used.

6

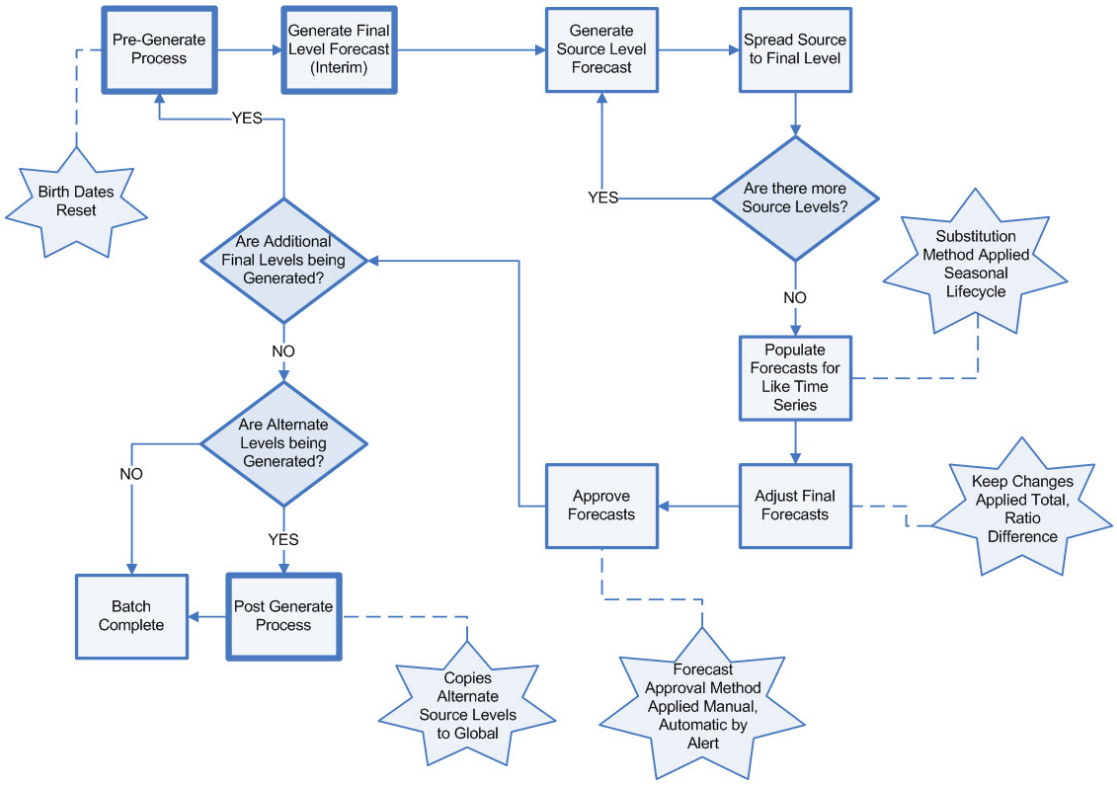
Batch Processing

This chapter describes the batch forecast process including:

- About RDF Batch Scripts
- PreGenerateForecast
- Executable: generate
- RDFvalidate
- UpdateFnhbiRdf
- Generate Halo Lift Effect Script
- Demand Transference PreProcessing Script

Figure 6–1 provides a high-level overview of the batch forecast process.

Figure 6–1 Batch Forecast Generation Process



About RDF Batch Scripts

Batch scripts, which are configured externally, are required by RDF to run processing before, during and after forecasting. This section outlines a sequence of common batch scripts that may need to be configured during RDF implementation to achieve desired processing to meet your business needs.

RDF Binaries

This section lists all RDF binaries with details and usage information

1. Hierarchy Loads

   Use the RPAS utility loadHier to load standard hierarchy files into RDF. The RPAS utilities positionBufferMgr and reconfigGlobalDomainPartitions may also need to be called by the loadHier script.

   Note that RPAS supports centralized hierarchy loads, which means that this script can be run from the master domain, in a global domain environment.

   Note: If user-defined hierarchies are present, then always run reshape arrays before continuing.

2. Measure Loads

   Use the RPAS utility loadMeasure to load sales history, out of stock information, promotion variables and other necessary data into RDF.

   Note that RPAS now supports centralized measure loads, which means that this script can be run from the master domain, in a global domain environment.

3. Preprocessing

   Performs any preprocessing calculations needed for forecasting. This script would invoke the preprocessing rule group that would be configured in the RDF configuration. Preprocessing performs any necessary scrubbing of historic data before forecast generation.

   This script must be run from local domains, if the preprocessing calculations involve RHS and LHS measures that are non-HBI, as would typically be the case for preprocessing calculations. Parallel running along local domains is possible.

4. Forecast Generation

   In order to generate forecasts, run the following RDF files:

   - PreGenerateForecast

     This file is run from a master domain and performed before running Generate.

   - Generate

     This file is run from local domains in a global domain environment.

5. UpdateFnhbiRdf

   This is an optional script, which is needed only if an alternate hierarchy dimension from the Product hierarchy is used as a dimension in a forecast level.

   This script needs to be run from the master domain, unless it is known that only one local domain has forecast data. Then calling this script from that local domain can save some time.

Note: If more than one local domain may have forecast data, then this script must be called from the master domain.

6. Alert Manager

   Use the RPAS utility alertmgr to evaluate alert conditions specified in the RDF configuration.

   With RPAS 12.0.6 and beyond, it is now possible to run alertmgr from local domains, followed by a final synchronizing run from the master domain, to synchronize alert hit counts at the master domain level. This can be achieved by running alertmgr -findAlerts from the local domains. After running alertmgr -findAlerts from the local domains, run alertmgr-sumAlerts from the master domain. The alertmgr -findAlerts process is more calculation intensive, but it can be performed in parallel at the local domain level.

7. Export Forecasts

   Use the RPAS utility exportData to export RDF forecasts from RDF for use by external systems. Users typically export the Approved forecasts from RDF. Also, refer to the RDF integration scripts (`rdf_e_rms.sh`) packaged with RPAS.

   This script is run from local domains.

8. Autoworkbook Build

   This script performs any necessary automated workbook builds as set up by the user. Automated workbook builds are set up by users to automate the workbook build process, so that they do not have to make the same wizard selections each time the workbook is built, and they do not have to wait for workbooks to build. The underlying RPAS utility used is wbbatch. This needs to be run from the local domains.

9. Generate Halo Lift Effect Script

   This script (genHaloLift.sh) is used to generate cross promotion halo lift effect. The script uses last approved forecast and the cross promotion halo effect matrix from CPEM to calculate the halo lift effect ratio, and this halo lift effect ratio will be used in the next forecast batch to calculate the cross promotion halo lift unit.

   This script is run from master domain

   Note: If this script is not run, then no Halo effects are incorporated in the forecast during the regular batch run, even when the Forecast Administration settings specify that Halo lifts should be produced.

10. Generate Demand Transference Effect Script

    This script (dtPreProcessingBatch.sh) is used to generate adjusted weekly demand transference effects. The script uses the integrated assortment multipliers and the effective dates to calculate the time-phased assortment multipliers. The time-phased assortment multipliers are used in the forecast batch to calculate the demand transference lift unit. The script also integrates the base rate of sales for the new item demand forecast.

    This script, dtPreProcessingBatch.sh, loads three measures based on the final level specified by the user. If no final level is specified, all the valid final levels will be iterated. Currently there is only data at final level 01 for the measures

fmafstdt01xb, brsls01xb, and assmul01xb. If desired, you can prepare data at the valid final levels for these three measures and specify the final level when running this script. For example, if you want to run the script at final level 06, then prepare data files for measures fmafstdt06xb, brsls06xb, and assmul06xb.

The following table lists the source for the measure data files where *{valid_final_level}* is the final level that you want to run on. The *{valid_final_level}* in the following measures will be replaced by the final level specified by you. If no final level is specified, this script will run on all the valid final levels.

Measure Data File	Source
fmafstdt*{valid_final_level}*xb	Forecast Start Date Override 1 - itm/str/week-Final
brsls*{valid_final_level}*xb	Base Rate of Sales 1 - itm/str/week-Final
assmul*{valid_final_level}*xb	Assortment Multipliers 1 - itm/str/week-Final

This script is run from master domain.

Note: If this script is not run, then no demand transference effects are incorporated in the forecast during the regular batch run, even though the Forecast Administration settings specify that demand transference lifts should be produced.

PreGenerateForecast

Used in a global domain or simple domain environment, PreGenerateForecast is an RDF executable that registers all measures with a birth date prior to forecast generation using generate. The first time PreGenerateForecast is run for a level, it registers the appropriate token measures for that level. If a global domain environment is implemented, PreGenerateForecast may be run against the Master or a Local domain. At either level, the necessary measures to produce the batch forecast are registered across all domains.

PreGenerateForecast requires an input file in the form of an XML. The XML is configured with the following values:

Value	Description
FinalLevel	The Final Level Number that is used to generate the forecast.
OutputFile	The name of the resulting file located at the root of the domain after PreGenerateForecast is run. The OutputFile includes the values set for FinalLevel and Override in addition to the birth date. This date is the Forecast Generation Date, and it is passed to the domains when generate is run. The date is produced in the following format: yyyymmddHhhMmm (Example: 20050327H13M36). When this birth date is selected in the Forecast Approval wizard, it is viewed as: (03/27/2005 13:36).
Override	A True or False value. When generate is passed a True value, the Next Run Date is ignored, and the batch forecast uses today's date as the Next Run Date; and the batch is run. When generate is passed a False value, the batch forecast will run if the Next Run Date is the same as today's date.

Note: When the Run Batch template is used to generate the batch forecast, PreGenerateForecast is run automatically. If a global domain environment is implemented, forecasts produced across Local domains using Run Batch cannot be aggregated in the Master domain because they do not share the same Forecast Generation Date.

PreGenerateForecast Usage

PreGenerateForecast -InputFile filename

InputFile is required.

The input file should be an XML file similar to Example 6–1:

Example 6–1 PreGenerateForecast Format

```
<Parameters>
   <Parameter>
      <Key>FinalLevel</Key>
      <Value>1</Value>
 </Parameter>

<Parameter>
    <Key>OutputFile</Key>
    <Value>MyOutput.xml</Value>
</Parameter>
<Parameter>
    <Key>Override</Key>
    <Value>true</Value>
    </Parameter>
</Parameters>
```

FinalLevel and OutputFile are required parameters of the XML file.

Override is an optional parameter of the XML file (default is **false**).

Other parameters may be included in the input XML file. They are passed through to the output XML file.

Return codes:

- 0 - Success (either ran pre-generate or did not need to run)
- 1 - Bad input
- 2 - Failure

To set the level of information provided, use -loglevel with values of: all, profile, debug, information, warning, error, or none. To disable timestamp header use -noheader.

Executable: generate

Used to produce the batch forecast, generate is an RDF executable. This executable requires as an input, the OutputFile resulting from PreGenerateForecast and is called `generate.xml`.

This binary runs RDF's batch process. Generate can take two optional inputs: level and override.

Usage

The usage of `generate -InputFile Filename` is as follows:

Script Name

`generate -InputFile Filename`

Parameters

The following parameters setting are included in the input file:

- birth
- startdate
- finallevel
- override

The override input must be True or False. The defaulted value is False if this option is not included in the input file. When override is False, `generate.xml` only starts the batch process if current time is later than the next run date in the domain. When the override is True, `generate.xml` starts the batch forecast regardless of the next run date.

The generate binary invokes code in the BatchForecast library to run the batch process.

`finalLevel` and `birth` are required parameters of the XML file. `override` (**false**) and `StartDate` (Default Forecast Start Date) are optional parameters of the XML file (defaults in parentheses).

Return Codes

The return codes include:

- 0—Success (either ran generate or did not need to run)
- 1—Bad input
- 2—Failure

To set the level of information provided, use -loglevel with values of:

- all
- profile
- debug
- information
- warning
- error
- none

To disable timestamp header use -noheader.

The input file should be an XML file that looks similar to the following example:

Example 6–2 XML Format for generate

```
<Parameters>
    <Parameter>
         <Key>Birth</Key>
         <Value>20041027H11M52</Value>
</Parameter>
```

```
<Parameter>
       <Key>StartDate</Key>
       <Value>20041027</Value>
</Parameter>
<Parameter>
        <Key>FinalLevel</Key>
        <Value>1</Value>
</Parameter>
<Parameter>
        <Key>Override</Key>
        <Value>true</Value>
   </Parameter>
</Parameters>
```

RDFvalidate

`RDFvalidate` automatically runs during the domain install, and it can also be run at any time against a Master or one subdomain. If run against the Master Domain, it checks the master and all subdomains. If run against a subdomain, it checks the Master and only the subdomain (not all other subdomains). This function verifies that:

- If there is a partition dimension, it must be along the product hierarchy.
- Domains are cleanly partitioned, this means that for the partition dimension, there exists only one position in each local domain, whether partitioning along the main or an alternate (or branch) product hierarchy.
- All data, measures, and levels are defined properly based on the partition dimension.
- Causal parameters are properly defined based on final, source, and causal levels.

Usage

The usage of `rdfvalidate -d pathToDomain` is as follows:

Script Name

```
rdfvalidate -d pathToDomain
```

To get this usage text, use -?, -help, or -usage. To get the version of this utility, use -version. To set the level of information provided, use -loglevel with values of: all, profile, debug, information, warning, error, or none. To disable timestamp header use -noheader.

RDF Validation

Table 6–1 displays the validation performed internally by the plug-in and the RDFvalidate utility.

Table 6–1 Internal Validation Performed by the Plug-in and RDFvalidate utility

Validation Area	Steps
Hierarchies and Dimensions	a. Verify day dimension exists on calendar hierarchy
	b. If there is a partition dimension, it must be along the product hierarchy.
For Final Levels	a. Intersection (fintxlxb) ■ Cannot be blank ■ Must be at or below all source level intersections ■ Must be at or below the partition dimension on the partition branch
	b. Seasonal profile (seasprofxlxb) can be either: ■ Blank ■ Measure name (only one) ■ Must be valid measure ■ Should be of type real ■ Measure intersection must be equal to the level intersection
	c. Source data (datasrcxlxb) must be a measure name (only one) ■ Must be a valid measure ■ Should be of type real ■ Measure intersection must be at or below the final level intersection
	d. Plan data (r fplanxlxb) must be either: ■ Blank ■ Measure name (only one) ■ Must be valid measure ■ Should be of type real ■ Measure intersection must be equal to the final level intersection

Table 6–1 (Cont.) Internal Validation Performed by the Plug-in and RDFvalidate utility

Validation Area	Steps
For Source Levels	**a.** Intersection (fintxlxb) ■ Cannot be blank ■ Must be at or above final level intersection ■ Must contain a dimension from the partition hierarchy ■ Must be either: ■ At or below the partition dimension on the partition branch. ■ On a branch of the partition hierarchy. If on a branch of the partition hierarchy, also check if domains are cleanly partitioned (executable only). This means for the branched dimension on the partition hierarchy, each position for that dimension can exist in only one sub-domain.
	b. Seasonal profile (seasprofxlxb) can be either: ■ Blank ■ Measure name (only one) ■ Must be valid measure ■ Should be of type real ■ Measure intersection must be equal to the level intersection
	c. Spreading profiles (sprdprofxlxb) ■ Can only be blank if source level intersection equals final level intersection ■ Must be comma-separated list of Curve levels and measure names (can be mixed) ■ If Curve level, must be a valid Curve level (final profile) ■ If measure: ■ Must be a valid measure ■ Should be of type real ■ Measure intersection must be at or higher than final level

Executable Only

Table 6–2 displays the validation performed internally by the RDFvalidate utility.

Table 6–2 RDFvalidate Utility

#	Executable Only	Steps
1	Domains are Cleanly Partitioned	**a.** Verify that there is only one partition dimension per subdomain.
2	For Final and Source Levels	**a.** Causal Aggregation Profile (aggxlxb) values should be either: ■ Blank ■ Measure name (one only) ■ Should be a valid measure ■ Should be of type real ■ The intersection of the measure must be at or above final level
		b. Causal Calculation Intersection (calcintxlxb) values should be either: ■ Blank ■ Intersection Intersection must be valid: ■ Must contain the calendar dimension ■ Must be at or above level intersection
		c. Causal Data Source (calcdtsrcxlxb) values should be either: ■ Blank ■ Measure name (one only) ■ Should be a valid measure ■ Should be of type real ■ The intersection of the measure must be at or above level intersection
		d. CausalHigher Intersection (cslhint) values should be either: ■ Blank ■ Intersection ■ Must be valid intersection ■ Must not contain the calendar dimension ■ Must contain a dimension from the partition hierarchy. ■ Must be at or above level intersection ■ Must be either: ■ At or below the partition dimension on the partition branch. ■ On a branch of the partition
	Note: If on a branch of the partition hierarchy, also check if domains are cleanly partitioned (executable only). This means that for the branched dimension on the partition hierarchy, each position for that dimension can exist in only one sub-domain.	

Table 6–2 (Cont.) RDFvalidate Utility

#	Executable Only	Steps
2. (continued)	For Final and Source Levels (continued)	**e.** Causal Spread Profile (spreadxlxb) values should be either: ■ Blank ■ Measure name (one only) ■ Should be a valid measure ■ Should be of type real ■ The intersection of the measure must be at or above final level
		f. Deseasonalized Demand Array (ddemandxlxb) values should be either: ■ Blank ■ Measure name (one only) ■ Should be a valid measure ■ Should be of type real ■ The intersection of the measure must be the level intersection less the calendar dimension
3.	For Final Levels only	**a.** Default History Start Date (defhstdt) values should be either: ■ Blank ■ A date within the calendar
		b. Forecast Start Date (dfxlxb) values should be either: ■ Blank ■ A date within the calendar

Promote Validation

Plug-in and Executable

1. Hierarchies and Dimensions:
   Check whether or not PTYP, FLVL, and PROM exist in Data Hierarchy. If not, create them.
2. Promotion Names:

   Check if promotion names have 1 to 4 characters.
3. Causal levels must be at or below the partition dimension on the partition branch.

UpdateFnhbiRdf

UpdateFnhbiRdf is required after Generate is run if an alternate hierarchy dimension from the Product hierarchy is used as a dimension in a forecast level. It performs the following functionality:

- Checks that certain measures are cleanly partitioned
- Copies corresponding cells (based on the partition) from each sub-domain to the master domain
- Runs automatically with the Run Batch wizard

- After ensuring that the FNHBI (Forced non-Higher Based Intersections) measures are cleanly partitioned, UpdateFnhbiRdf copies corresponding cells (based on the partition dimension) from each sub-domain into the master domain

Usage

The usage of `UpdateFnhbiRdf -d pathToDomain -InputFile filename` is as follows:

Script Name

```
UpdateFnhbiRdf -d pathToDomain -InputFile filename
```

To get this usage text, use `-?`, `-help`, or `-usage`. To get the version of this utility, use `-version`. To set the level of information provided, use `-loglevel` with values of: `all`, `profile`, `debug`, `information`, `warning, error`, or `none`. To disable timestamp header, use `-noheader`.

The `InputFile` format expected is as printed by the usage information. The timestamp or the birth key will have to be the same as the one output by pregenerateForecast, that is used by `generate.xml`.

Generate Halo Lift Effect Script

This script (genHaloLift.sh) generates the cross promotion halo lift effect. The script uses last approved forecast and the cross promotion halo effect matrix from CPEM to calculate the halo lift effect ratio, and this halo lift effect ratio will be used in the next forecast batch to calculate the cross promotion halo lift unit.

This script is run from master domain.

Usage

The usage of `genHaloLift.sh -d {MasterDomainPath} [-finallevel {FinalLevelString}] [ -maxprocesses {MaxProcesses} | -noparallel ] [-debug] [-u]` is as follows:

Script Name

```
genHaloLift.sh -d {MasterDomainPath} [-finallevel {FinalLevelString}] [
-maxprocesses {MaxProcesses} | -noparallel ] [-debug] [-u]
```

Required Arguments

```
[-d] : MasterDomainPath
```

Optional Arguments

Optional Arguments	Description
[-finallevel]	FinalLevelString [default = all]
[-maxprocesses]	MaxProcesses for parallel [default = 3]

Optional Flags

Flag	Description
[-noparallel]	Set MaxProcesses to 1

Flag	Description
[-debug]	Enable debug mode
[-u]	Show usage of this script

Example

```
genHaloLift.sh -u(Run the batch) genHaloLift.sh -d . -finallevel 01
```

Note: To get the usage of the script, use flag -u.The only required argument is for the path of a master domain:

```
-d {MasterDomainPath}
```

Although not required, but highly recommended for the efficiency considerations is:

```
-finallevel {FinalLevelString}
```

If it is not specified, the script goes over all the final levels of the domain.

Demand Transference PreProcessing Script

This script (dtPreProcessingBatch.sh) generates the adjusted weekly demand transference effects based on assortment multipliers per item/store, assortment multipliers decay factor and demand assortment multiplier applying period.

This script is run from master domain.

Usage

The usage of `dtPreProcessingBatch.sh -d {MasterDomainPath} [-noparallel | -maxprocesses {MaxProcesses}] [-finallevel {FinalLevelString}]` is as follows:

Script Name

```
dtPreProcessingBatch.sh -d {MasterDomainPath} [-noparallel | -maxprocesses
{MaxProcesses}] [-finallevel {FinalLevelString}]
```

Required Arguments

[-d] : MasterDomainPath

Optional Arguments

Optional Arguments	Description
[-finallevel]	FinalLevelString [default = all]
[-maxprocesses]	MaxProcesses for parallel [default = 3]

Optional Flags

Flag	Description
[-noparallel]	Set maxProcesses to 1

Example

```
dtPreProcessingBatch.sh -d . -noparallel -finallevel 01
```

Note: To use the script, run the script without any arguments.

The only required argument is the path of a master domain:

```
-d {MasterDomainPath}
```

Although not required, but highly recommended for the efficiency considerations is:

```
 -finallevel {FinalLevelString}
```

If it is not specified, the script goes over all the final levels of the domain.

```
-d
```

7

Cross Promotion Effects Module (CPEM)

Cross Promotion Effects Module (CPEM) is used to generate the crossing promotion effects between merchandise and location. There are two types of cross effects that are calculated at corresponding levels:

- Cannibalization effects
- Halo effects

The cannibalization level and Halo level are to define at which product and location levels the cannibalization or Halo cross effects are produced. The Halo level should be higher than cannibalization level along hierarchy.

RDF was launched with Promotional Halo and Cannibalization capabilities in January 2013. The functionality has had limited adoption due to various factors including necessary data required to support reliable results, market maturity and implementation complexity. In an attempt to share lessons learnt with our experience so far, we would recommend the following to our customers considering implementing this functionality:

Starting Conditions

Both Halo and Cannibalization deal with the cross item effects of running promotions, that is, the impact of promoting an item on the sales of other related items. The most important inputs for estimating these cross effects are a robust baseline forecast and self-promotion effects from RDF Causal. Hence, we recommend that CPEM be considered for implementation only after RDF Causal has been up and live for a 12-18 month period, delivering reliable baseline and causal forecasts to drive business decisions.

- **Replacement SKUs** — All historical SKUs that are replacement or promotional variants of each other need to be grouped together into a Plan SKU. The Plan SKU level needs to be used within RDF for estimating both baseline forecasts and causal forecasts.
- **Appropriate pre-processing** — That corrects for missing sales periods and other factors that might bias the estimated baseline forecast and hence impact that quality of CPEM output.

Set Up for Cannibalization

Careful configuration and pre-implementation set up is required for Cannibalization to run. It is important to note that Cannibalization (though configurable) is designed to run at an item-group level (as defined in the following list) and not the individual SKU level.

Note: Cannibalization needs to stay inside a local domain.

- **Item-group or L1 Set Up** — This is the level at which Cannibalization is estimated. SKUs have to be grouped (outside the system and fed in). These are groups of items that are typically promoted together. For example, in the yogurt Category, SKUs of a particular brand, size, fat content but different flavor variants. An attribute analysis exercise, should inform this L1-grouping. What are the key combinations of attributes that need to be the same for all items in an L1-group and what attributes can be different? It is important to note that these attributes will vary significantly, from Category to Category.
- **Cannibalization-group Or L2 Set Up** — L2 is the level where the cannibalization is estimated. Only item group or L1 within a Cannibalization group are analyzed for possible Cannibalization, when one or more items are promoted. This could be a grouping of one or more classes or sub-classes. Note that L2 grouping needs to be a roll up of L1 groupings, that is, every L1 item-group needs to cleanly map to one and only one L2 grouping (many to one mapping from L1 to L2).
- **L1-pairing Set Up** — In addition to set up of L1 or Item-groups, RDF's CPEM also needs to be told of legal/possible L1-pairs. This requires careful analysis on a Category by Category basis. For example. SKUs within a certain regular price range could be considered Cannibalistic and hence valid L1-pairs. It is recommended that this analysis be conducted carefully, ideally in conjunction with Customer Decision Trees (CDTs) to ensure the correct pairings are set up for each Category.
- **Co-promotion Effects are Turned Off** — A system flag that effectively looks for conditions when both L1 pairings are promoted at the same time and accounts for it appropriately.

Halo effects

Current functionality is designed for a limited use case for businesses with minimal promotional activity. The solution aims to estimate Halo effects by observing the impact on sales of other subclasses when a subclass is promoted. As we vetted this with a broad set of retail use cases with significant promotional activities, we recognized that this functionality is not adequate. We are revisiting our approach to estimating Halo effects and welcome partnerships with retailers.

CPEM is a standalone RPAS instance and its hierarchies including calendar, product, location, and RHS product, are the same as the ones in RDF and all measure data files are imported from RDF. RDF prepares the CPEM needed measure data files based on the cannibalization and Halo level, so CPEM and RDF should have exactly the same cannibalization and Halo levels set for CPEM to work. In other words, if you change the cannibalization or Halo level in RDF (CPEM), then you should make the same changes in CPEM (RDF).

Functionality

CPEM performs the following functionality:

- Define and override the cross promotion estimation related parameters in the Effect Estimation Administration workbook.
- Run the CPEM batch script to produce both cannibalization and Halo promotion effects. The CPEM determines the regression based on the promotion, baseline sales, sales, and price information to produce the cross effects.

- Review and approve the produced crossing promotion effects in the Effect Estimation Review and approval workbook.
- Export the approval cross promotion effects into RDF.

Input

CPEM is a standalone RPAS instance and it is a single domain.

Hierarchy Files

The hierarchy files of Product, Calendar, location and RHS Product are the same as the RDF domain.

1. Calendar (CLND) Hierarchy File
2. Merchandise or Product (PROD) Hierarchy File
3. Location (LOC) Hierarchy File
4. RHS Merchandise or Product (PROR) Hierarchy File

Measures Data Files

The measures data files needed in CPEM are imported from RDF as listed in Table 7–1.

Table 7–1 CPEM Measure Data Files imported by RDF

Measure Name	Intersection	Description
slscann	Cannibalization Level + week	The aggregated weekly sales history at cannibalization level.
slsbaselinecann	Cannibalization Level + week	The aggregate weekly baseline sales history at cannibalization level.
normprccann	Cannibalization Level + week	The weekly normalized price at cannibalization level.
promoindcann	Cannibalization Level + week	Weekly Promotion indicator at cannibalization level.
slsHalo	Halo Level + week	The aggregated weekly sales history at Halo level.
slsbaselineHalo	Halo Level + week	The aggregate weekly baseline sales history at Halo level.
normprcHalo	Halo Level + week	The weekly normalized price at Halo level.
promoindHalo	Halo Level + week	Weekly Promotion indicator at Halo level.

Output

CPEM produces the cross promotion effects and then outputs the cross promotion cannibalization effects and Halo effects as listed in Table 7–2.

Table 7–2 Measure Data Files Output by CPEM

Measure Name	Intersection	Description
FinCannEff	Cannibalization Level + RHS Cannibalization Product	The cross promotion cannibalization effects.
FinHaloEff	Halo Level + RHS Halo Product	The cross promotion Halo effects.

Installation Considerations

The following sections provide an overview of CPEM installation.

Note: For additional information, refer to the chapter, "CPEM Installation" in the *Oracle Retail Demand Forecasting Installation Guide*.

Installation Dependencies

RPAS and RDF must be installed before setting up and configuring CPEM. For information on installing these products, refer to the *Oracle Retail Predictive Application Server Installation Guide* and *Oracle Retail Demand Forecasting Installation Guide*.

Environment Setup

Before downloading the installation package to the UNIX server, a central directory structure to support the environment needs to be created. This central directory is referred to as `<cpem_directory>`. Set `<cpem_directory>` to the full path name to the $CPEM_HOME variable.

CPEM Installer

The CPEM installer performs the following functions:

- Downloads the configuration and batch scripts into the `<cpem_directory>/config` and `<cpem_directory>/bin` directories
- Downloads a set of sample hierarchy and data files into the `<cpem_directory>/input` directory
- Builds a sample domain at `<cpem_directory>/domain/cpem`

Custom Domain Build

To do a custom build of a domain:

1. Update the `globaldomainconfig.xml` file with the correct domain paths.
2. If needed, update the default environment variables in `environment.sh`.
3. Run the `buildcpem.sh` script: `./ buildcpem.sh.sh`

CPEM Taskflow for the RPAS Fusion Client

The CPEM installation software enables you to install the taskflow and online help files for the RPAS Fusion Client. In order to install the taskflow files, the RPAS Fusion Client must already be installed. For more information on installing the RPAS Fusion Client, see the *Oracle Retail Predictive Application Server Installation Guide*.

During the RPAS Fusion Client installation, the installer automatically sets up the RPAS domain connection configurations in the `ProfileList.xml` file. If you choose to set up the domain connection after the installation or set up an additional domain, you must manually set up the connection. For more information, see the *Oracle Retail Predictive Application Server Administration Guide for the Fusion Client.*

Batch Processing

CPEM has a batch script to generate the cross promotion effects and export the final effects. The following batch scripts are available in CPEM:

- CPEM batch script
- Cross effect export script

Running the Batch Process

To generate the cross effects, run the `cpem_batch.sh` script in the domain.

```
./ cpem_batch.sh
```

To export the cross promotion effects, run the `exportCPEMtoRDF.ksh` script in the domain

```
./ exportCPEMtoRDF.ksh
```

8

AutoSource

The AutoSource binary may be used to determine the optimal source level for a product/location. For the final level specified, AutoSource produces a forecast using each source level. The source level that produces the best PAE (Percent Absolute Error) for a time series is selected as the Optimal Source Level. The AutoSource results may be accessed by the user through the Forecast Maintenance workbook. If the Optimal Source Level is to be used for a product/location, the `Use Optimal Source` parameter should be set to True.

The AutoSource binary invokes code in the BatchForecast library to run the batch process. AutoSource can take four inputs: mode (required), finallevels (required), today, and timelimit (required for ONCEONLY and CYCLE mode).

The AutoSource binary does the following:

- Provides a starting Source Level recommendation for new forecasting customers. The recommended Source Level can be applied to the Final Level, which would allow the user to be focused on other tuning activities.
- Is helpful for existing customers that are starting to forecast new businesses. AutoSource can be included as an activity in the customer's forecasting roll-out process.
- Is useful for merchandise groups that have shifting demand patterns due to business or market changes such as pricing and marketing strategy changes, or product realignment.

AutoSource uses the forecast horizon to compute the PAE (Percent Absolute Error). If the forecast horizon is changed from the default of 13 weeks, AutoSource starts forecasting that number of weeks back. For example, if you have a forecast horizon of 52 weeks, AutoSource starts its analysis 53 weeks before today. This approach can disallow Winters and Seasonal models if sufficient calendar is not available. If the forecast horizon is 52 weeks, you should have at least 3.5 years of history for AutoSource to be able to perform all of its analysis.

Unlike Generate, there is no interim forecast calculation in AutoSource. Instead, AutoSource attempts to generate an AutoES result at the final level, then uses that result to perform the source level spreads.

Note: If the time series data is dense enough at the final level, the spread is not based on a linear contribution to the source. It will not maintain the source shape, and it will make recommendations based on such spreads.

AutoSource makes an initial recommendation to all the product/location combinations with sufficient data to perform analysis. Subsequent PAE calculation and comparison only occurs to these product/location combinations. The product/location combination without sufficient data (total sales = 0 during history region or total sales = 0 during forecast evaluation region) will not get any recommendation.

Inputs to AutoSource Binary

AutoSource is invoked from a script or the command line. The binary inputs are detailed in Table 8–1 and Table 8–2:

Table 8–1 AutoSource Binary Input Descriptions

Binary Inputs	Description		Example
-d	Relative or absolute path to domain		-d /cygdrive/c/Domains/RDF/ldom2
-mode	Includes the following options:		
	RESTART	Resetting measures, such that the next run starts without prior information. This option does not actually kick off any source level optimization run. Use this option when a clean run is desired, and then run AutoSource with one of the following next modes.	-mode RESTART
	CYCLE	If AutoSource doesn't complete an optimization run due to the time limit, the next time it is run it picks up where it left last time. For instance, if there are 10 source levels and during one run AutoSource only evaluated 3 source levels, then the next time it runs it optimizes source levels 4 and up. CYCLE without a time limit will never finish. Once the last source level was evaluated, AutoSource starts with the first level again.	-mode CYCLE
	ONCEONLY	AutoSource completes the run or stops when the time limit is up	-mode ONCEONLY
-flvllist	A list of the final forecast levels to be optimized		-flvllist 1

Table 8–2 AutoSource Optional Binary Input Descriptions

Optional Binary Inputs	Description	Example
[-today]	Specifies the date when AutoSource stops the evaluation of the forecast error. The evaluation starts at the date given by today minus the number of periods specified in the forecast length. Hence the time interval over which AutoSource evaluates the forecast error is [today - forecast length, today]. The date should be in the RPAS format stored in the dim_day array.	-today D20010101
[-timelimit]	Time, in minutes allowed AutoSource to run. It is optional for RESTART mode, but required for ONCEONLY and CYCLE mode. It needs to be greater than 0 to allow AutoSource to run.	-timelimit 10,000
[-noclear]	This is a flag indicating if temporary information should be deleted. If not specified, the temporary information is deleted.	-noclear

Example 8–1 AutoSource Binary Example 1

```
AutoSource -d . -mode RESTART -flvllist 01,06 -today 20020101 -timelimit 10
AutoSource -d . -mode ONCEONLY -flvllist 01,06 -today 20020101 -timelimit 10
```

Example 8–2 AutoSource Binary Example 2

```
AutoSource -d . -mode CYCLE -flvllist 01,06 -today 20020101 -timelimit 10
```

If only running AutoSource periodically, then use the RESTART and ONCEONLY modes. If the run exceeds the time limit during a RESTART run, then ONCEONLY should be run. If you want to start from the beginning, RESTART and ONCEONLY should be run again.

If AutoSource is scheduled as part of the daily cron job, use CYCLE. CYCLE runs RESTART and then ONCEONLY consecutively.

Refer to the *Oracle Retail Demand Forecasting User Guide* for specifics pertaining to the Forecast Maintenance Workbook and picking optimal levels.

Note: For item/stores that are new or highly seasonal, AutoSource may not return the best recommendation since new items may not have an adequate sales history length and highly seasonal items may only sell for a short period during the year. For these items, you should not set the AutoSource recommendation as default at the final level. Only use AutoSource recommendations for item/stores that have an adequate sales history.

AutoSource Measures

The following AutoSource measures are available in the Forecast Maintenance workbook.

Optimal Source Levels

Displayed only at final levels, a value is populated in this field if AutoSource has been run on the final level. The AutoSource executable evaluates all levels associated to a final level and returns the Source Level that yields the optimal forecast results or lowest error.

Pick Optimal Level

Set only at final levels, a check mark in this field indicates that the batch forecast should use the Optimal Source Level selected by AutoSource.

The final level measure Optimal Source Levels is used for reference. The RDF user can view the optimal Source Level that was determined by AutoSource. This Source Level was chosen by generating forecasts at all Source Levels and determining the lowest forecast error (PAE) at the final level.

If the user would like to use the Optimal Source Level during forecast generation they can set the Pick Optimal Level Boolean measure to True.

If Pick Optimal Level is set to True, when forecast generation is run, the optimal Source Level is used. The Forecast Method set at the optimal Source Level and the additional associated forecast parameters is also used.

Usage

AutoSource -d pathToDomain -mode RESTART/ONCEONLY/CYCLE -flvllist lvlx,lvly

[-today] todayString (the same format as in dim_day)

[-timelimit] minutes (optional for RESTART mode, but required for ONCEONLY and CYCLE mode)

[-noclear]

To get this usage text, use -?, -help, or -usage.

To get the version of this utility, use -version.

To set the level of information provided, use -loglevel with values of: all, profile, debug, information, warning, error, or none.

To disable the timestamp header, use -noheader.

The mode input must be one of RESTART, CYCLE, or ONCEONLY.

The flvllist must be a comma separated list of final levels.

The today input must be the same format as in dim_day, For example, YYYYMMDD.

The timelimit is in minutes.

- RESTART: This mode initializes the system in preparation for a new Autosource batch process.
- ONCEONLY: This mode runs the Autosource batch process until it completes or until the timelimit has been reached (whichever comes first).

> **Note:** In order to run in ONCEONLY mode, RESTART mode has to be run first.

- CYCLE: This mode continuously runs the Autosource batch process by first running the RESTART mode, and then running ONCEONLY. The CYCLE mode allows the Autosource batch process to always use the latest data in determining the optimal source level for a prod/loc.

> **Note:** For Autosource usage examples, refer to Example 8–1 and Example 8–2.

9

Forecast Approval Alerts

This chapter describes Forecast Approval Alert configuration.

About Alerts

Alerts can be configured through the RPAS Configuration Tools or can be manually registered in the domain. The alert expressions require familiarity with the RPAS rule functions. Registering an alert with the alert category of FORECAST_APPROVAL allows RDF to use the alert expression during the batch forecasting process to determine if a time series is automatically approved. When this category of alert is registered, the pick lists for Default Approval Method (in Forecast Administration) and the Approval Method Override (in Forecast Maintenance) are updated to include the label of the alert.

Note that the label of the alert when displayed in the list has certain special characters replaced with a space. These characters are:

- comma
- colon
- right parenthesis
- left parenthesis

You can select the alert for any product/location.

The following steps are an example of Forecast Approval Alert configuration using the example domain that is provided in the release package:

Prerequisite: Build Global Domain

1. Option 1:

   Run PreGenerateForecast or Generate

   Option 2:

   Use regTokenMeasure to Manually Register Any Token Measures Needed to Support the Alert Expression
2. Register the Alert Measure
3. Register the Expression for the Forecast Approval Alert

Prerequisite

Build Global Domain

Using the Mock Install Configuration, build the global domain environment.

Step 1 (Option 1)

Run PreGenerateForecast or Generate

If using a pristine global domain or simple domain environment, token measures have yet to be registered in the domains. Since you do not know the specific birth date at configuration time, token measures allow for measures with birth dates (a time stamp applied during the batch) to be evaluated. The token measure that we are using in this example is System Forecast for level 1 (sf01). The registration of the token measures can be accomplished by running PreGenerateForecast (in a global domain environment) or Generate (in a simple domain environment). This removes the need to manually run `regTokenMeasure`.

Step 1 (Option 2)

Use regTokenMeasure to Manually Register Any Token Measures Needed to Support the Alert Expression

If you prefer to manually register the token measures, the regTokenMeasure must be run with -FNHBI option if in a global domain environment. This allows the token measures to have different values across subdomains. The token measure requires a value to the measure while registering. In Example 9–1, the token measure is registered in the Master Domain and are made to be equal to pos (Weekly Sales) since pos has the same base intersection (item/store/week) and data type (real) as the System Forecast for level 1.

Example 9–1 regTokenMeasure

```
C:\Domains\RDF>regTokenMeasure -d . -add sf01=pos -fnhbi
```

Note: Do not perform this step if the batch has already been generated since the batch will have automatically registered sf01.

Step 2

Register the Alert Measure

The next step in the process is to register the alert measure in the Master Domain. In Example 9–2, an alert with the name of rdf_a1_1 with label of Alert1level1 is being registered.

Example 9–2 Register the Alert Measure

```
C:\Domains\RDF>regmeasure -d $DOMAIN_DEST_DIR -add "rdf_a1_1" -label
"Alert1level1" -baseint "itemstr_" -db "data/myalerts" -type boolean -navalue
False
```

Step 3

Register the Expression for the Forecast Approval Alert

The alertmgr utility is used to register the alert and the alert expression. In the Example 9–3, the alert expression first filters out time series with low volume sales (items with forecasts less than three units). This alert compares the System Forecast in the first three weeks in the forecast horizon with last approved forecast for the same three weeks. If the values are within a 33% range, the full forecast horizon is set to automatic approval, otherwise the Alert is triggered. This is all done in batch, so the Alert Manager is not necessary to apply the alert. For intersections that do not qualify for automatic approval, the Approval Comment on the Approval Worksheet in the Forecast Approval workbook contains *refused by alert*. You may use the Alert Manager to insert this alert into the workbook to display the intersections that have the alert flag set to True.

Example 9–3 alertmgr Utility

```
C:\Domains\RDF> alertmgr -d . -register "rdf_a1_1" -category "FORECAST_APPROVAL"
-categoryLabel "Alert1level1" -expression "rdf_a1_
1=if(tssum(@sf01,index([clnd].[week],flookup(lfsXLXB.level([data].[flvl]+[prod].to
p),[data].[flvl].[flvl01])),
index([clnd].[week],flookup(lfsXLXB.level([data].[flvl]+[prod].top),[data].[flvl].
[flvl01]))+3)>=3.0,
abs(1-tssum(@sf01,index([clnd].[week],flookup(lfsXLXB.level([data].[flvl]+[prod].t
op),[data].[flvl].[flvl01])),
index([clnd].[week],flookup(lfsXLXB.level([data].[flvl]+[prod].top),[data].[flvl].
[flvl01]))+3)/(tssum(lappf01XB,index([clnd].[week],flookup(lfsXLXB.level([data].[f
lvl]+[prod].top),[data].[flvl].[flvl01]))+3,index([clnd].[week],flookup(lfsXLXB.le
vel([data].[flvl]+[prod].top),[data].[flvl].[flvl01])))+0.001))>.33,false)"
```

10

Adding New Local Domains

This chapter provides an overview on adding new local domains to an existing RDF global domain. New local domains can be added using the RPAS reconfigGlobalDomainPartitions utility. It is important to keep in mind that as new local domains are added, they must be added such that the RDF partitioning requirements continue to be met. This means each new local domain can only contain one position along the partition dimension.

When new local domains are added, the following additional scripts must be run, which are located in the /bin directory of $RPAS_HOME:

loadCurveParameters.ksh

This script is used to load the Curve data parameter measures including Profile Data Source, Default Source Profile, Default Profile Approval Method, Training Window Method, and Normal Value. This action is typically performed within the plug-ins at domain creation time, however, when you add a new local domain to an existing domain environment, the plug-ins are not run, and therefore this script performs that action manually.

Usage:

loadCurveParameters -d fullPathToDomain -s fullPathToNewSubdomain

loadRDFParameters.ksh

This script is used to load the RDF data parameter measures including Default Required Method, Default Source Level, Data Plan, Seasonal Profile, and Spreading Profile. This action is typically performed within the plug-ins at domain creation time, however, when you add a new local domain to an existing domain environment, the plug-ins are not run, and therefore this script performs that action manually.

Usage

```
loadRdfParameters -d fullPathToDomain -s fullPathToNewSubdomain
```

11

Internationalization

Internationalization is the process of creating software that can be translated more easily. Changes to the code are not specific to any particular market.

Oracle Retail applications have been internationalized to support multiple languages.

Refer to "Internationalization Considerations for RETL" for information about internationalization and RETL.

Translation

Translation is the process of interpreting and adapting text from one language into another. Although the code itself is not translated, components of the application that are translated include the following:

- Graphical user interface (GUI)
- Error messages

The following components are not translated:

- Documentation (online help, release notes, installation guide, user guide, operations guide)
- Batch programs and messages
- Log files
- Configuration tools
- Reports
- Demonstration data
- Training materials

The user interface has been translated into the following languages:

- Brazilian Portuguese
- Chinese (Simplified)
- Chinese (Traditional)
- Croatian
- Dutch
- French
- German
- Greek

- Hungarian
- Italian
- Japanese
- Korean
- Polish
- Russian
- Spanish
- Swedish
- Turkish

Note: For information about adding languages for the first time or for translation information in general, see the *Oracle Retail Predictive Application Server Administration Guide for the Classic Client*.

A

RPAS and RDF Integration with RMS

This appendix addresses RMS integration between RPAS and RDF.

Integration Approach with RMS

The strategy for the extraction of foundation data from Retail Merchandising System (RMS) is for the extract programs (RMSE) to provide flat files in a generic format. For each solution that uses this data, transformation scripts are used to reformat the data as needed to produce a file suitable for loading into the application. For the instances of data coming from RPAS to non-RPAS applications, extract programs are specific to the application in need of the data. Other scripting languages are then used (Perl or AWK) to perform additional data formatting.

This appendix summarizes the following:

Specifics on the usage of RMS extract programs (RMSE's) within the RDF transformation programs are beyond the scope of this document. See the *Oracle Retail Merchandising System Operations Guide* for more information on the RMS extract programs.

Note: For integration compatibility information, see the *Oracle Retail Predictive Application Server Installation Guide.*

Environment Variable Setup

In addition to any variables identified in the RMS integration documentation, the transformation or extract programs or both require the environment variables listed in Table A–1.

Table A–1 Environment Variables

Environment Variable	Description
$RPAS_INTEGRATION_HOME	Identifies the location of the integration scripts when `/common/header.ksh` is run. This variable is used for all integration scripts packaged with the ARPOPlatform except those included in RFX (see "$RDF_HOME").
$TO_RPAS	The staging area for the data to be loaded into RPAS. This directory should be located at the same level as the root of the RPAS domain. For example, if the domain RDF is located in Domains directory. (example: `/Domains/RDF)`, then $TO_RPAS should be located at the same level as RDF (example: `/Domains/to_rpas`).
$FROM_RPAS	The staging area for the data extract out of RPAS. This directory should be located at the same level as the root of the RPAS domain. For example, if the domain RDF is located in Domains directory. (example: `/Domains/RDF)`, then $FROM_RPAS should be located at the same level as RDF (example: `/Domains/from_rpas`).
$RDF_HOME	Identifies the location of the root of the RFX directory. The RFX directory packaged with the ARPOPlatform should be added to the location RFX directory packaged with the RMS RETL programs.
$RI_RMSVERSION	Identifies the version of RMS. If this variable is not set, the integration scripts assume an RMS version of 13. Set the value of this environment variable to 13.
RFX_HOME	The directory that holds the RETL executable files.
RETL_JAVA_HOME	The directory that contains Java. See the latest version of the *Oracle Retail Extract, Transform, and Load Programmer's Guide* for the version of Java that corresponds to the version of RETL used.
PATH	To run RETL, include these items in the PATH variable: $RETL_JAVA_HOME/bin:$RETL_JAVA_HOME/jre/bin:$RFX_ HOME/lib:$RFX_HOME/bin:$PATH
LIBPATH	To run RETL, include these items in the LIBPATH variable: $RETL_JAVA_HOME/jre/bin:$RETL_JAVA_ HOME/jre/bin/classic:$RFX_HOME/bin:$LIBPATH
RMS_RPAS_HOME	This variable needs to be set in order to maintain compatibility with RMS. The required value is `$RDF_HOME`.
RESOURCE_DIR	This directory contains resource information to support multi-language capability for calendar-related fields as well as the `Unassigned` value present in some schema. The default (and recommended) value is `$RDF_HOME/resources`.

Table A–1 (Cont.) Environment Variables

Environment Variable	Description
PREFERRED_LANG	This value indicated the language that the calendar labels are shown The default is english (_en), but any value from `$RDF_HOME/resources/SupportedLanguages.txt` where *column 4 = Yes* is supported, provided the corresponding retl_msgs.* file is present.
RDF_SCHEMA_DIR	This directory contains the RFX or RETL schemas. This variable is used when converting the `Unassigned` value in the schemas to a locale specific language. The default (and recommended) value is `$RDF_HOME/rfx/schema`.
RPAS_ALERTNAVIGATIONTHRESHOLD	This variable controls alerts and specifies the override for the default alert navigation threshold. The RPAS alertmgr api takes navigation threshold as an argument. If this navigation threshold is not passed into the api as an argument, then the value of the environment variable RPAS_ALERTNAVIGATIONTHRESHOLD is used. If RPAS_ALERTNAVIGATIONTHRESHOLD is not set then a default value of 5000 is used.
UPGRADE_BACKUP_DIR	This directory is used to backup existing schema information prior to converting the `Unassigned` value in the schemas to a locale specific language. The default (and recommended) value is `$RDF_HOME/schema_backup`.

RDF Transformation Programs

This section describes the RDF transformation scripts.

Common Program for All Transformations

The `rdft.ksh` script runs all of the necessary data extraction and transformation scripts (`rmse_*.ksh` and `rdft_*.ksh`, respectively) that are needed to produce the files to be loaded into RPAS/RDF/Planning. Most of these scripts are run in parallel (as background jobs).

Usage

```
rdft.ksh  [-x]  [-c] [-sd startDate] [-ed endDate] [-d dir]
```

Arguments

The following table lists and describes the arguments for these scripts.

Argument	Description
-x	This option causes the running of the RMS data extraction wrapper (`rmse.ksh`) to be skipped.
`-c`	This option causes FILE_DATE in rmse_config.env to be set to the current date instead of using VDATE.
`-sd`	This option sets the start date for optionally filtering out records based on date. Records with dates prior to this date are excluded from loading into RDF. The date needs to be in the format YYYYMMDD.
ed	This option sets the end date for optionally filtering out records based on date. Records with dates after this date are excluded from loading into RDF. The date needs to be in the format YYYYMMDD.

Argument	Description
-d	This option causes all programs run by rdft.ksh to be obtained from the *dir* directory.

Transformations of Merchandise Hierarchy Data

The rdft_merchhier.ksh script is the primary script used to build the data for RPAS from the RMS Merchandise Hierarchy tables. The schema used to produce the output file depends on the attributes and differentiator settings in RMS:

Case	Settings	Outcome
1	If PROD_ATTRIBUTES_ACTIVE = false and DIFFS_ACTIVE = false	Then rdft_merchhier.base.schema is used to produce the file. In this case, attributes and diff fields are not included in the merchandise hierarchy file.
2	If PROD_ATTRIBUTES_ACTIVE = true and DIFFS_ACTIVE = false	Then rdft_merchhier.attributes.schema is used to produce the file. This schema must be manually edited to support a specific attribute model and must be kept in sync with rmse_attributes.schema and rmse_attributes.ksh.
3	If PROD_ATTRIBUTES_ACTIVE = false and DIFFS_ACTIVE = true	Then rdft_merchhier.schema is used to produce the file. In this case, diff fields are included in the merchandise hierarchy file.
4	If PROD_ATTRIBUTES_ACTIVE = true and DIFFS_ACTIVE = true	Then an error results. In this release, the combination of diffs and attributes is not supported.

Intermediate schema and scripts which may be used (depending on configuration options) to produce the merchandise hierarchy file:

- rdft_diff.domain.schema
- rdft_merchdiff.domain.schema
- rdft_merchhier_diff_trans.ksh
- rdft_merchhier_split_by_domain.ksh
- rdft_clean_partition.ksh

Additional merchandise hierarchy support for issue domains is provided in rdft_item_loc.ksh. This script is designed to produce a full item list for issues domains, only containing items that exist in the warehouses.

Note: Issues-specific data transformation functionality is triggered based on the issues setting in RMS (ISSUES_ACTIVE must be set to True).

The script rdft_clean_partition.ksh was designed to mimic some of the functionality of local domains prior to local domains being supported by RPAS. This script is not relevant to versions of RDF 12.0 or later. It is retained for use by customers on earlier versions of RDF.

Transformations of Location Hierarchy Data

The rdft_orghier.ksh script is the primary script used to build the location data file needed for RPAS from the RMS Organizational Hierarchy Table.

The following constants may be modified in the script based on location hierarchy data requirements:

Modifiable Constant	Description
COMPANY_NAME	The label for the company position to be populated in the file.
COMPANY_ID	The name for the company position to be populated in the file.
STORE_CLASS_CONCAT	When set to True, causes the STORE_CLASS to be concatenated on the left of the STORE_CLASS_DESCRIPTION field in the final Store data output file.
ADD_AT_SIGN_TO_WH_DESC	When set to True, causes the WHSE_NAME field in the Warehouse output file to have an *at sign* "@" prefix.
LONG_WAREHOUSE_ RECORDS	When set to True, the Warehouse output records consists of 16 fields. If it is False, the records contain only four fields, WH, WHSE_NAME, COMPANY, and CO_NAME.

Intermediate schemas which may be used (depending on configuration options) to produce the location hierarchy file:

- `rdft_issues.schema`
- `rdft_issues_long.schema`
- `rdft_orghier_store.schema`

Note: Issues-specific data transformation functionality is triggered based on the issues setting in RMS (ISSUES_ACTIVE must be set to True).

Transformations of Calendar Hierarchy Data

The `rdft_calhier.ksh` script transforms the Calendar Hierarchy data extracted from RMS for loading into RPAS.

Configuration inputs to the `rdft_calhier.ksh` script include:

- **DATE_PREF** - The path to the file that contains text indicating whether the format of the Date Description field are mm/dd/yyyy or dd/mm/yyyy. See the *Oracle Retail Merchandising System Operations Guide* for date format options.
- **LAST_DOW** - The path to the file that contains a day of week name or abbreviation indicating which day of the week is considered to be the end of the week for the fiscal calendar being used at this installation.

The RMS to RDF calendar integration has been enhanced to allow calculation of several new quantities. These new quantities are present in `rdft_calhier.dat`, and include:

- dos (Day of Season)
- woy (Week of Year)
- wos (Week of Season)

In order for these values to calculate properly, the `rdft_clndhier.ksh` script looks to two files, `rfx/etc/first_day_of_season.txt` and `rfx/etc/first_week_of_season.txt`. Each file contains a single number representing the day or the week number from where the season starts.

For example, suppose the data coming from RMS is starting from the 2nd day of the 3rd week of season, then the first_day_of_season.txt has two (2) and first_week_of_season.txt has three (3) as starting day and starting week of the season.

If the files are missing then the season is assumed to start with first day of the first week of the season.

Transformations of Daily Sales and Issues Data

The rdft_daily_sales.ksh script produces the daily sales and issues data files based on regular, promotion, clearance, and issues.

The following constant may be modified in the script based on data requirements:

- DOM_START_COL

  Defines the starting column position of the Domain ID in the RETL output schema. This is needed by rdft_merchhier_split_by_domain.ksh to split the files by domain ID. If the OUTPUT_SCHEMA file is modified, the value of DOM_START_COL may also require modification from the default value.

Intermediate schemas which may be used (depending on configuration options) to produce either or both of the sales or issues data file:

- rdft_daily_sales.schema

Note: Issues-specific data transformation functionality is triggered based on the issues setting in RMS (ISSUES_ACTIVE must be set to True).

Transformations of Weekly Sales and Issues Data

The rdft_weekly_sales.ksh script produces the weekly sales and issues data files based on regular, promotion, clearance and issues.

The following constant may be modified in the script based on data requirements

- DOM_START_COL

  Defines the starting column position of the Domain ID in the RETL output schema. This is needed by rdft_merchhier_split_by_domain.ksh to split the files by domain ID. If the OUTPUT_SCHEMA file is modified, the value of DOM_START_COL may also require modification from the default value.

Intermediate schemas which may be used (depending on configuration options) to produce either or both of the sales or issues data files:

- rdft_weekly_sales.schema

Note: Issues-specific data transformation functionality is triggered based on the issues setting in RMS (ISSUES_ACTIVE must be set to True).

Transformations of Store Open Date Data

The rdft_open_date.ksh script produces the Store/Warehouse Opening Date data file.

Intermediate schema used to produce the store open date data files:

- rdft_open_date.schema

Transformations of Store Close Date Data

The rdft_close_date.ksh script produces the Store/Warehouse Closing Date data file.

Intermediate schema used to produce the store closing date data files:

- rdft_close_date.schema

Transformations of Out of Stock Indicator Data

The rdft_outofstock.ksh script produces the Store and Warehouse (issues) Out of Stock Indicator data extracted from RMS.

Intermediate schema and scripts which may be used (depending on configuration options) to produce the Out of Stock Indicator data file:

- rdft_outstock_split_by_domain.awk
- rdft_outofstock.schema
- rdft_outofstock_issues.schema
- rdft_outofstock_sales.schema

RDF Transformation Matrix

The following matrix identifies the transformation scripts and schemas used for each the hierarchy and data files produced for RDF:

Table A–2 RDF Transformation Scripts and Schema

Directory	Script or Schema Name	Merchandise Hierarchy	Location Hierarchy	Calendar	Daily Sales & Issues	Weekly Sales & Issues	Out of Stock Indicator	Store Open Dates	Store Close Dates
rfx/lib	rdft_merchhier_diff_ trans.ksh	X							
	rdft_merchhier_split_by_ domain.ksh	X							
	rdft_outofstock_split_by_ domain.ksh						X		
rfx/ schema	rdft_close_date.schema								X
	rdft_daily_sales.schema				X				
	rdft_diff.domain.schema	X							
	rdft_ merchierdiff.domain.schema	X							
	rdft_ merchier.attributes.schema	X							
	rdft_merchhier.base.schema	X							
	rdft_ merchhier.domain.schema	X							
	rdft_merchhier.schema	X							
	rdft_open_date.schema							X	
	rdft_orghier_issues.schema	X							
	rdft_orghier_issues_ long.schema	X							
	rdft_orghier_strore.schema	X							
	rdft_outofstock.schema						X		
	rdft_outofstock_ issues.schema						X		
	rdft_outofstock_sales.schema						X		

Table A–2 (Cont.) RDF Transformation Scripts and Schema

Directory	Script or Schema Name	Merchandise Hierarchy	Location Hierarchy	Calendar	Daily Sales & Issues	Weekly Sales & Issues	Out of Stock Indicator	Store Open Dates	Store Close Dates
	rdft_weekly_sales.schema					X			
rfx/src	rdft_ksh	X	X	X	X	X	X	X	X
	rdft_calhier.ksh			X					
	rdft_clean_partition.ksh	X							
	rdft_close_date.ksh								X
	rdft_daily_sales.ksh								
	rdft_item_loc.ksh	X							
	rdft_merchhier.ksh	X							
	rdft_open_date.ksh							X	
	rdft_orghier.ksh	X							
	rdft_outofstock.ksh						X		
	rdft_weekly_sales.ksh					X			

Loading Transformed RMS Data into RDF

This section describes the loading of transformed RMS data into RDF.

renameRETLFiles.ksh

After the transformation of the RMS hierarchy and history data, the data files must be combined and renamed to match the appropriate measure names expected by RDF. This is accomplished by a script, `renameRETLFiles.ksh`. Usage of this script is optional, but recommended. The script reads the data files produced by the RETL transformation, combines and renames the data files, and writes the output to the domain's input folder. Existing data in the domain's input folder is backed up prior to the writing the new files.

Usage

```
renameRETLFiles.ksh -dataDir <data dir> -domInput <domain's input dir>
```

Arguments

The script has two required arguments:

Argument	Description
-dataDir	Required. The path to the data directory. This directory contains the RMS data transformed by the RDF RETL scripts. This is typically $RDF_HOME/data.
domInput	Required. The input directory of the domain in which the transformed data is to be loaded. The directory is typically the input directory under the domain root.

loadRETLMeasures.ksh

This script is another convenience script. Its usage is optional. It calls loadHierarchy and loadMeasure on the files produced by the RETL transformation and renamed by `renameRETLFiles.ksh` to load data into a specified RDF domain. It loads:

- The Calendar, Merchandise, and Location hierarchies
- Standard measures such as daily sales (dpos), regular sales (rsal), promo sales (psal), and clearance sales (csal).
- Optional measures. These can be anything, but are typically user-specified, such as out of stock indicators (outage), store opening dates (stropen), store closing dates (strclose), and so on. The list is not limited to measures associated with the RETL transformation – any valid measure may be specified.

Usage

`loadRETLMeasures -d <domain path> [-m measure1] [-m measure2]` and so on.]

Arguments

The script accepts the following arguments:

Argument	Description
-d	Required. The path to the domain. This script (like all RPAS-based applications) assumes the data files are in the domain's input directory.
-m	Optional. The name of a measure to load. The script assumes the data file is in the domain's input directory

Common Programs for Extracts

This section describes the common programs for extracts.

config.ksh

The `config.ksh` script is a configuration directory that requires both the RMS version being integrated and the backup action to be defined.

Arguments

The following optional arguments are available:

Argument	Description
Name of the domain	Defaults to directory name
Number of the domain	Defaults to the 2 last digits of the directory name
Format of timestamp attached to logs and processed input files	Defaults to: (date +"%b%d%a%I%M%p") (example: Aug02Thu0111PM)
Data Drop	Defaults to ../../to_rpas
Data Export	Defaults to ../../from_rpas
Log Drop	Defaults to ./logs
Error Drop	Defaults to ./err
Reclass Data	Defaults to ../reclass_data

functions.ksh

The `functions.ksh` script file contains ksh functions that are used by scripts in [DOM]/scripts. It should be sourced, not run in order to preserve environment variables.

header.ksh

The `header.ksh` script file should be run at the beginning of any implementation-specific script to setup function libraries, environment, and platform-specific routines.

Extract of Forecast Data for RMS

This section describes the scripts that extract forecast data for RMS.

rdf_e_rms.ksh

The `rdf_e_rms.ksh` script extracts forecast demand value and standard deviation (cumulative interval) at both day and week aggregations from an RDF domain.

Arguments

This script accepts the following arguments:

Argument	Description
-t	<Domain Type> (S for sales, I for issues)
-w	<Data Width> ([7...18], defaults to 14)
-d	<Domain> (defaults to current directory)
-n	<Domain Number> (defaults to last two digits of domain)
-s	<Start Date> (format YYYYMMDD, if not set, defaults to RPAS_TODAY; if RPAS_TODAY not set, then defaults to system date)

Output Files

`${RPAS_EXPORT}/d<s|i>demand.<Domain Number> (demand at day)`

`${RPAS_EXPORT}/w<s|i>demand.<Domain Number> (demand at week)`

The following table provides information about the output file data format.

Field	Start	Width	Format
Day | EOW Day	1	8	Alpha
Product ID	9	25	Alpha
Location ID	34	20	Alpha
Demand	54	14	Alpha
Std. Dev. Demand	68*	14*	Numeric (floating point, 4 decimal digits with decimal)
* Width of Demand and Std. Dev. Demand may be overridden with the -w parameter; stated values Demand width and Std. Dev. Demand start and width are based on default width of 14.			

Note: The following must be defined in the shell environment prior to calling this script:

- RPAS_HOME
- RPAS_INTEGRATION_HOME

Editing for Simple Domains

If you have a simple domain, then the `functions.ksh` script needs to be edited. Under the CreateWeek2DayArray () function call, edit this line by removing the `-fnhbi` flag.

Edit from:

```
regmeasure -d $RPAS_DOMAIN -add WEEK2DATE -type date -baseint week -fnhbi
-db data/hmaint
```

to:

```
regmeasure -d $RPAS_DOMAIN -add WEEK2DATE -type date -baseint week -db
data/hmaint
```

Load of Extracted Forecast Data and Standard Deviations to RMS

This section describes the load of extracted forecast data and standard deviations to RMS.

rmsl_forecast.ksh

The `rmsl_forecast.ksh` script pulls the daily/weekly forecast items into RMS.

During the loading of each domain file the following steps are performed:

1. Truncate the partition in the RMS forecast table which corresponds to the domain ID.

   Note: Partition names should always be in the format: `[tablename]_[domainID]`

2. Append a domain field and insert the `domain_id` into each record.
3. Load the forecast data into the RMS forecast table.

Example A–1 rmsl_forecast.ksh Format

```
rmsl_rpas_forecast.ksh daily | weekly
```

Intermediate schemas which may be used (depending on configuration options) to produce the forecast data files:

- `rmsl_forecast_daily.schema`
- `rmsl_forecast_weekly.schema`

Extract of Diff Profile Data for Allocation

This section describes the extract of diff profile data for Allocation.

profile_e_alloc.ksh

The `profile_e_alloc.ksh` script extracts Curve diff profiles for use by Allocation.

Arguments

The script accepts the following arguments:

Argument	Description
-p	<Profile Number>
-m	<Mask Measure> (Optional mask; only positions for which the mask value is non-NA are exported.)
-w	<Data Width> ([7...18], defaults to 12)
-d	<Domain> (defaults to current directory)
-n	<Domain Number> (defaults to last two digits of domain)

Output file

```
${RPAS_EXPORT}/d1<Product Level>.<Domain Number>
```

Note: Where Product Level is the Aggregation intersection's Prod dimension.

The following table provides information about the output file data format.

Field	Start	Width	Format
Product ID	1	25	Alpha
Location ID	26	20	Alpha
Diff ID (optional)	46	36	Alpha
Quantity	82	12*	Numeric (floating point, 4 decimal digits, no decimal)*
Std. Dev. Demand	68*	12*	Numeric (floating point, 4 decimal digits with decimal)
* Quantity width may be overridden with the -w parameter.			

Note: The following must be defined in the shell environment prior to calling this script:

- RPAS_HOME
- RPAS_INTEGRATION_HOME

Extract of Store Grade Data for RMS

This section describes the extract of Store Grade data for RMS.

grade_e_rms.ksh

The `grade_e_rms.ksh` script extracts store grades for use by RMS.

Arguments

The script accepts the following arguments:

Argument	Description
-t	<Timestamp> (YYMMDDTTTT). This value corresponds to the timestamp of the Cluster Membership measure (clpm+<Timestamp>) to be extracted
-d	<Domain> (defaults to current directory)
-n	<Domain Number> (defaults to last two digits of domain)

Output File

`${RPAS_EXPORT}/gr<Timestamp>.<Domain Number>`

Note: The following must be defined in the shell environment prior to calling this script:

- RPAS_HOME
- RPAS_INTEGRATION_HOME

The following tables list the output file data formats:

Header Records
FHEAD
Line ID Number
GRADU

Detail Records	Data Type	Width	Description
FDETL	String	5	Always 'FDETL'
Line ID	Number	10	Line Sequence Identifier (generated by the script)
Grade Group ID	Number	8	This value corresponds to the first 8 characters of the Cluster Run Name measure (clnam+<user-defined name>) set by the user in the Generate Cluster wizard in Grade. For integration with RMS, the Cluster Run Name must be populated with only numeric characters.
Grade Group	String	120	This value corresponds to the first 120 characters of the Cluster Run Name measure (clnam+<user-defined name>) set by the user in the Generate Cluster wizard in Grade.
Grade Store	Number	10	Valid Grade Store derived from the point mapping measure (clpm+<user-defined name>). For integration with RMS, the Cluster Run Name must be populated with only numeric characters.
Grade ID	Number	10	Valid Grade ID derived from the point mapping measure (clpm+<user-defined name>). For integration with RMS, the Cluster Run Name must be populated with only numeric characters.
Grade Name	String	120	Valid Grade Name, also derived from the point mapping measure (clpm+<user-defined name>).

Footer Records
Line ID Number
FDETL Line Total Number

RDF Extract Matrix

The following matrix identifies the extract scripts and schemas used for each the data files produced for RMS.

Directory	Script or Schema Name	Forecasts and Standard Deviations	Diff Profiles
common	config.ksh		
	functions.ksh	X	
	header.ksh	X	X
curve	profile_e_alloc.ksh		X
grade	grade_e_rms.ksh		
plan	Plan_e_alloc.ksh		
	Plan_e_price.ksh		
	Plan_e_plcblwdm.ksh		
	Plan_e_ploblwdm.ksh		
rdf	rdf_e_rms.ksh	X	
	rmsl_forecast.ksh	X	
	rmsl_forecast_daily.schema	X	
	rmsl_forecast_weekly.schema	X	

Internationalization Considerations for RETL

These sections describe internationalization considerations for RETL:

- Calendar Data
- Unassigned Value in Schemas

Calendar Data

Calendar data (that is names of weeks and months) is translatable into Oracle supported languages. Enabling this feature requires setup of these two environment variables:

- export RESOURCE_DIR=$RDF_HOME/resources
- export PREFERRED_LANG=_en

Note that for PREFERRED_LANG, the default is english (_en), but any value from `$RDF_HOME/resources/SupportedLanguages.txt` where *column 4 = Yes* is supported, provided the corresponding retl_msgs.* file is present.

Once these variables are setup, then the translation takes place for RETL. No separate script is required.

Unassigned Value in Schemas

The `Unassigned` value in schemas can be converted to a locale specific equivalent. To enable this feature, set these environment variables:

- export RESOURCE_DIR=$RDF_HOME/resources
- export PREFERRED_LANG=_en
- export RDF_SCHEMA_DIR=$RDF_HOME/rfx/schema
- export UPGRADE_BACKUP_DIR=$RDF_HOME/schema_backup

Note that for PREFERRED_LANG, the default is english (_en), but any value from `$RDF_HOME/resources/SupportedLanguages.txt` where *column 4 = Yes* is supported, provided the corresponding retl_msgs.* file is present.

After the environment variables are set, then run this script: `$RDF_HOME/rfx/etc/update_schema_nullvalues.ksh`.

This script backs up existing schemas into UPGRADE_BACKUP_DIR and then replaces the Unassigned value with a locale specific equivalent. This update preserves field widths.