Oracle® Retail Promotion Intelligence and Promotion Planning and Optimization

Implementation Guide

Release 13.1.1

September 2009

ORACLE®

Oracle® Retail Promotion Intelligence and Promotion Planning and Optimization Implementation Guide, Release 13.1.1

Primary Author: Judith Meskill

California, and imbedded in Oracle Retail Promotion Planning & Optimization application.

(viii) the software component known as **Style Report™** developed and licensed by InetSoft Technology Corp. of Piscataway, New Jersey, to Oracle and imbedded in the Oracle Retail Value Chain Collaboration application.

(ix) the software component known as **DataBeacon™** developed and licensed by Cognos Incorporated of Ottawa, Ontario, Canada, to Oracle and imbedded in the Oracle Retail Value Chain Collaboration application.

You acknowledge and confirm that Oracle grants you use of only the object code of the VAR Applications. Oracle will not deliver source code to the VAR Applications to you. Notwithstanding any other term or condition of the agreement and this ordering document, you shall not cause or permit alteration of any VAR Applications. For purposes of this section, "alteration" refers to all alterations, translations, upgrades, enhancements, customizations or modifications of all or any portion of the VAR Applications including all reconfigurations, reassembly or reverse assembly, re-engineering or reverse engineering and recompilations or reverse compilations of the VAR Applications or any derivatives of the VAR Applications. You acknowledge that it shall be a breach of the agreement to utilize the relationship, and/or confidential information of the VAR Applications for purposes of competitive discovery.

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

Contents

5 Staging and Loading Data

6 Database Configuration

7 Managing the Environment

8 Forecast Accuracy Indicator

9 Reports

10 Internationalization

Preface

The *Oracle® Retail Promotion Intelligence and Promotion Planning and Optimization Implementation Guide* provides an overview of the implementation process at a retail customer site.

Audience

This document is intended for administrators of the Promotion Intelligence and Promotion Planning and Optimization application.

Related Documents

For more information, see the following documents in the Oracle Retail Promotion Intelligence and Promotion Planning and Optimization documentation set:

- *Oracle Retail Promotion Intelligence and Promotion Planning and Optimization Release Notes*
- *Oracle Retail Promotion Intelligence and Promotion Planning and Optimization Installation Guide*
- *Oracle Retail Promotion Intelligence and Promotion Planning and Optimization Configuration Guide*
- *Oracle Retail Promotion Intelligence and Promotion Planning and Optimization Operations Guide*
- *Oracle Retail Promotion Intelligence and Promotion Planning and Optimization Data Model*
- *Oracle Retail Promotion Intelligence User Guide*
- *Oracle Retail Promotion Planning and Optimization User Guide*
- *Oracle Retail Promotion Intelligence and Promotion Planning and Optimization Licensing Information*

Customer Support

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

- https://metalink.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

If you are installing the application for the first time, you install either a base release (for example, 13.0) or a later patch release (for example, 13.1.1). If you are installing a software version other than the base release, be sure to read the documentation for each patch release (since the base release) before you begin installation. Patch documentation can contain critical information related to the base release and code changes that have been made since the base release.

Oracle Retail Documentation on the Oracle Technology Network

In addition to being packaged with each product release (on the base or patch level), all Oracle Retail documentation is available on the following Web site (with the exception of the Data Model which is only available with the release packaged code):

http://www.oracle.com/technology/documentation/oracle_retail.html

Documentation should be available on this Web site within a month after a product release. Note that documentation is always available with the packaged code on the release date.

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

Getting Started

This chapter introduces the Promotion Intelligence and Promotion Planing and Optimization product and provides an overview of the implementation process. It contains the following sections:

Introduction

Promotion Intelligence (PI) provides a collection of services and stored procedures that are used to assess the historical effectiveness of a set of promotions over a defined time period. The objective is to assess the incremental benefits from promoting items by comparing a promotion period to a baseline period. The resulting reports can help users to evaluate the performance of future promotions.

Promotion Planning and Optimization (PPO) is a web-based application that helps retailers to create and improve promotions. This process includes building offers, designing the ad layout, and producing a forecast to predict how the ad will perform. The user can leverage the information produced by Promotion Intelligence to make the best promotion decisions, using what-if analysis and predictive forecasting.

Figure 1–1, "Logical System Architecture" shows the relationship between Promotion Intelligence and Promotion Planning and Optimization. The overall process of the application, at a high level, is as follows:

1. Incoming data from the customer is provided to PI as text files and loaded into the database.
2. The Promotion Calc Engine (PCE) process the incoming data and produces forecast parameters.
3. Users, through the PPO UI, enter data into the system and use the application to run What If.
4. Reports are produced against the calculated metrics.
5. Outputs are exported to external systems.

Figure 1–1 Logical System Architecture

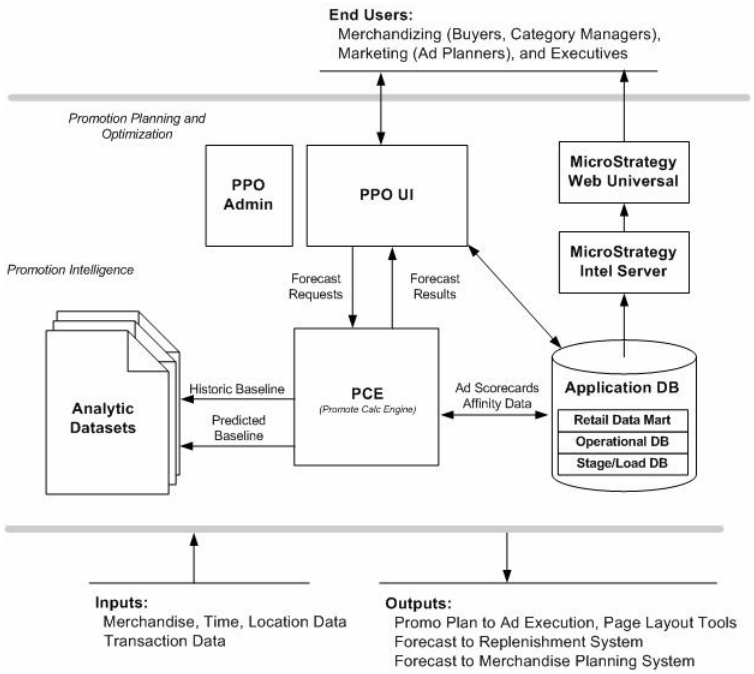
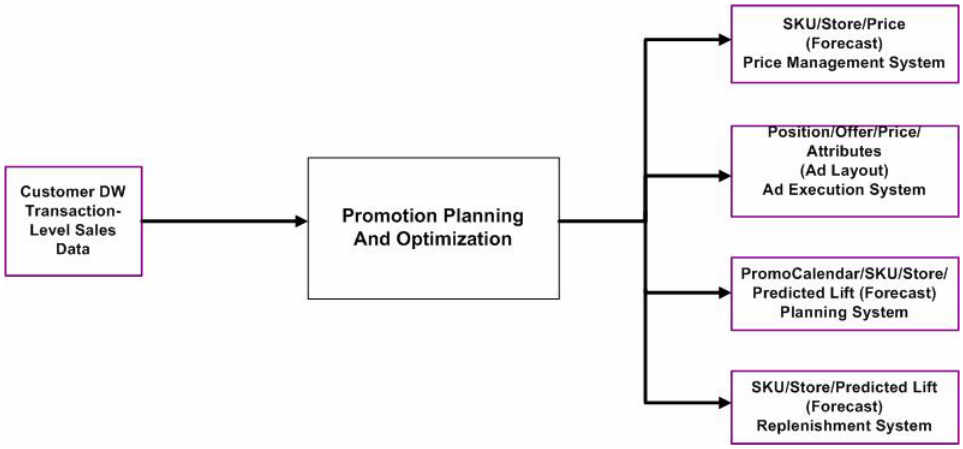


Figure 1–2, "Context Diagram" provides a greater level of detail regarding PPO.

Figure 1–2 Context Diagram



Documentation

The *Oracle® Retail Promotion Intelligence and Promotion Planning and Optimization Implementation Guide* is intended for use with the rest of the documentation set, as listed in the Preface. Here is a description of each of those documents.

The *Oracle® RetailPromotion Intelligence and Promotion Planning and Optimization Installation Guide* provides detailed instructions about planning the installation, setting up the database, setting up the application server, and installing the application.

The *Oracle® Retail Promotion Intelligence User Guide* and the *Promotion Planning and Optimization User Guide* both provide step-by-step procedures for all UI-related tasks.

The *Oracle® Retail Promotion Intelligence and Promotion Planning and Optimization Administration Guide* provides information about managing users and managing stores.

The Oracle® Retail *Promotion Intelligence and Promotion Planning and Optimization Configuration Guide* is a technical reference guide that provides technical details about all application configuration points.

The *Oracle® Retail Promotion Intelligence and Promotion Planning and Optimization Operations Guide* is a technical reference guide that provides technical details about the standard interface, the standard load process, PI tools, and administrative tools.

The *Oracle® Retail Promotion Intelligence and Promotion Planning and Optimization Release Notes* provides a high-level overview of new features and significant issues for the latest release of the application.

Implementation Process

The implementation process creates a client-specific configuration. The configuration is determined by a client's individual business requirements.

Roles

The implementation process typically involves people with the following roles/tasks:

- Database engineer, responsible for data staging and loading
- Analyst, responsible for calculating demand parameters
- Application engineer, responsible for UI configuration
- Production engineer, responsible for the weekly processes

PPO Implementation

The PPO implementation involves configuring the application to match the customer's business needs. This includes managing the formatting and scheduling of the data feeds from the customer, configuring the display of the UI, configuring the RMI server configuration, configuring export, integration with PI, defining and loading the customer's user profiles and store set definitions.

PI Implementation

The PI implementation involves the configuration of the analytical parameters by the analytics team in order to build models and compute ad effectiveness based on the customer's historical data. The process includes:

- Measuring the baseline performance – what would have happened in the absence of promotions - particular to that product but during a normal promotional week (not a dark week)
- Measuring the incremental sales of promoted products (sales above the baseline).
- Performing market basket analysis. The resulting affinity rules show which items sell together and how frequently they sell together.

Process

At a high level, the implementation process consist of these steps:

1. Gather business requirement from the customer. This will include a detailed discussion between the customer and Oracle about how the product works and

what the customer wants to accomplish with the product. The information collected must provide sufficient information for the configuration process to begin. This collection of business requirements and configuration will be an iterative process. Oracle must educate the customer about the data that must be supplied for the configuration. This is a critical discussion and should include information about how the data is supplied and the formats that are required for the data.

2. Configuration design. This step involves determining the data required from the customer for the analytical configuration and the application configuration.
3. The historical data required for the analytical process is obtained from the customer. The analytical calculation are performed iteratively to analyze history, build demand parameters, build lift models, build affinity models, and build predicted models. The resulting values are used for the application configuration.
4. Application configuration. This is a systematic iterative process that requires both historical data and current data. A subset of the configuration occurs and is tested and then other configuration points are added to the configuration and the testing is repeated.
5. User acceptance testing.
6. Training.
7. Transition to a production environment.

Production

Once PI-PPO has been configured and the UAT has been completed, the application can be set up in the customer's production environment. The environment is designed to meet the customer's performance requirements. The nightly and weekly processes are scripted and automated. These processes include a database back up, the loading of the nightly or weekly data, and a status report. Scheduling and process checks are included in the automation. The setting up of the production environment focuses on:

- data transfer mechanism
- data file encryption
- hardware requirements
- operational schedule

2

Installation

The *Oracle® Retail Promotion Intelligence and Promotion Planning and Optimization Installation Guide* provides instructions for the installation of both Promotion Intelligence and Promotion Planning and Optimization.

Introduction

Refer to this guide for information about the following topics:

- Planning Your Installation
- Setting up the Database
- Setting up the Application Server
- Installation
- Setting Up Single Sign–On
- Installation Order (if installing another Oracle Retail Application)

3

User Management

This chapter provides technical details about the user management functionality. It contains the following sections:

Introduction

User Management is a utility that lets you create, modify, and remove user accounts from a central location. The User Management utility is installed automatically when you install the application.

Each user who accesses the application must have a user account. Each user account is assigned one or more roles that determine the types of functions the user can perform with the application.

About User Roles and User Actions

Roles are defined by a specific set of user actions. The actions that define each role serve to delimit the activities a user can perform. All actions are self-contained. For example, Create does not imply View. So a role must include all the actions that are necessary for complete functionality.

Note that master data includes the hierarchy information (merchandise, location, and calendar), configuration information (data aggregation levels and other parameters), promotion attributes, store sets information, SKU lists, image information (images and associated mappings for items in the merchandise hierarchy), and dark periods information (time periods that must be excluded from baseline calculations)."

PI-PPO comes with a default set of actions, loaded into ACTION_TBL:

- PROMO_CREATE_CE – add and delete campaigns and events.
- PROMO_EDIT_CE – make changes to campaigns and events.
- PROMO_VIEW_CE – view campaigns and events.
- PROMO_CREATE_MD – add and delete master data.
- PROMO_EDIT_MD – make changes to master data.
- PROMO_VIEW_MD - view master data.
- PROMO_CREATE_PROMO – add and delete promotions.

- PROMO_EDIT_PROMO – make changes to promotions.
- PROMO_VIEW_PROMO – view promotions.
- PROMO_MANAGE_CATEGORY – edit the offers for a promotion.
- PROMO_MANAGE_MERCHANDISE – edit Like Item information.
- PROMO_MANAGE_VEHICLE – edit the definition and design of the promotion vehicle, category assignments, white space allocation, and workflow.
- PROMO_EXPORT_PROMO – provides access to the Export button, which is used to create xml and txt files of promotions. Necessary for access to the Export API functionality.
- PROMO_VIEW_REPORTS – launch the Promotion Intelligence reports.
- PROMO_ADMIN_DOC – only users assigned this action can log into the application when the server is in maintenance mode. Also provides access to the following commands: releaselocks, clearcache, refreshprops, refreshloggin, refreshbundle, refreshconfig, modestage, nodeprod, and modemaint.
- PROMO_APPROVE_OFFER – allows ad planners to approve or deny submitted offers.

PI-PPO comes with a default set of roles, loaded into ROLE_ACTION_TBL:

- PROMO_AD_PLANNER – a member of marketing who is responsible for the entire promotional calendar. This user can create and edit calendar events and create promotions.
- PROMO_BUSINESS_ADMIN – a business user who is responsible for activities such as data maintenance and template management.
- PROMO_CATEGORY_MANAGER – the person directly responsible for one or more categories of merchandise, assigned at a given level in the merchandise hierarchy.
- PROMO_EXEC – an executive who monitors promotion performance across all merchandise categories. Such a user would expect to monitor performance at both a high level and a low level, but would not need to edit or execute promotions.
- PROMO_MERCH_PLANNER – a merchandise planner who executes merchandising plans. Such a user is assigned responsibility at the Chain level.
- PROMO_AGENT – configure users to execute forecast and pre-planned import tasks.
- PROMO_VER_PLANNER – a version planner who executes version plans. Such a user is assigned responsibility at the Chain level.
- PROMO_VER_MANAGER – a version manager who manages version plans. Such a user is assigned responsibility at the Department level.
- PROMO_MERCH_WHATIF – a what-if manager who manages at the Department level.

The following table shows the default assignment of actions to roles in PI-PPO.

Table 3–1 Actions Assigned to Roles

PI-PPO Role	Assigned Actions
PROMO_AD_PLANNER	PROMO_CREATE_CE
	PROMO_EDIT_CE
	PROMO_VIEW_CE
	PROMO_CREATE_MD
	PROMO_EDIT_MD
	PROMO_VIEW_MD
	PROMO_CREATE_PROMO
	PROMO_EDIT_PROMO
	PROMO_VIEW_PROMO
	PROMO_MANAGE_VEHICLE
	PROMO_EXPORT_PROMO
	PROMO_VIEW_REPORTS
	PROMO_APPROVE_OFFER
PROMO_BUSINESS_ADMIN	PROMO_CREATE_MD
	PROMO_EDIT_MD
	PROMO_VIEW_MD
	PROMO_BUSINESS_ADMIN
PROMO_CATEGORY_MANAGER	PROMO_MANAGE_CATEGORY
	PROMO_MANAGE_MERCHANDISE
PROMO_EXEC	PROMO_VIEW_CE
	PROMO_VIEW_MD
	PROMO_VIEW_PROMO
	PROMO_MANAGE_CATEGORY
	PROMO_MANAGE_VEHICLE
	PROMO_EXPORT_PROMO
	PROMO_VIEW_REPORTS
PROMO_MERCH_PLANNER	PROMO_VIEW_CE
	PROMO_VIEW_MD
	PROMO_VIEW_PROMO
	PROMO_EDIT_PROMO
	PROMO_EXPORT_PROMO
	PROMO_VIEW_REPORTS
PROMO_AGENT	PROMO_ADMIN_DOC
	PROMO_VIEW_PROMO
	PROMO_CREATE_PROMO
	PROMO_EDIT_PROMO
	PROMO_VIEW_CE

Table 3–1 (Cont.) Actions Assigned to Roles

PI-PPO Role	Assigned Actions
	PROMO_CREATE_CE
	PROMO_EDIT_CE
	PROMO_VIEW_MD
	PROMO_CREATE_MD
	PROMO_EDIT_MD
	PROMO_MANAGE_MERCHANDISE
	PROMO_MANAGE_CATEGORY
	PROMO_MANAGE_VEHICLE
	PROMO_EXPORT_PROMO
	PROMO_VIEW_REPORTS
PROMO_VER_PLANNER	PROMO_VIEW_CE
	PROMO_VIEW_MD
	PROMO_VIEW_PROMO
	PROMO_EDIT_PROMO
	PROMO_CREATE_PROMO
	PROMO_EXPORT_PROMO
	PROMO_VIEW_REPORTS
PROMO_VER_MANAGER	PROMO_MANAGE_VEHICLE
PROMO_MERCH_WHATIF	PROMO_VIEW_CE
	PROMO_VIEW_MD
	PROMO_VIEW_PROMO
	PROMO_EDIT_PROMO
	PROMO_EXPORT_PROMO
	PROMO_VIEW_REPORTS

Default actions cannot be deleted.

Roles are assigned to users with restrictions that are defined at or above a specific node of the merchandise hierarchy and the location hierarchy. The scope of actions can be across the merchandise and location hierarchies.

The sample file, "Role Assignment Sample xml File" provides an illustration of defining the scope.

About User Management Roles

User accounts with user management roles have access to features such as creating users, assigning roles, removing user accounts, resetting passwords.

When a user with a user management role logs on, a link to the User Management utility appears on the Main Menu.

The following list describes the default User Management roles:

- UM_READ_ONLY_ADMIN – This role allows read-only access to the User Management utility. This role has privileges to view the list of users and their roles and hierarchy levels, but not to create new user accounts or modify or inactivate existing ones.
- UM_ROLE_ASSIGN_ADMIN – This role allows assigning new roles (and related hierarchy levels) to existing user accounts, but it does not allow the creation of new user accounts.
- UM_USER_ADMIN – This role allows creating new user accounts, but it does not allow the assignment of roles to the new accounts.

User Management Bulk Loader Utility

To create user accounts for a group of users all at one time, you can use the User Management bulk loader utility.

Prior to running the User Management bulk loader utility, you must:

- Set the jndi.properties. The jndi.properties file, which is located in <installed>/modules/tools/conf/jndi.properties, specifies the initial context factory and the url where the JNDI lookups are carried out.

  For WebLogic, typical values are:

  java.naming.factory.initial=weblogic.jndi.WLInitialContextFactory
  java.naming.provider.url=t3://localhost:7001
- Make sure that usermanagement.ear, suiteproperties.ear, and common4p.ear are deployed on the running application server.

Users and Roles

You need to create and validate (using a tool like XML Spy) three xml files containing entries for Users, Roles, and Role Assignments.

Note that the actions associated with roles must be created, using brmadmin.sh in order for the roles to be successfully created.

- The user file contains user names. All user names must be unique. The schema includes a flag that indicates whether or not the password should be hashed.
- The Roles file contains the possible roles that can be assigned. All role keys must be unique. The action key attributes must be loaded into the database before the bulk loader utility can be used. All elements and attributes must be lower case.
- The Role Assignment file contains user names and the role or roles associated with the user name. The user names must be loaded into the database before this file can be processed by the bulk loader utility. All elements and attributes must be lower case. The merchandise ID and the Location ID are provided by a pipe-delimited string of CLIENT_LOAD_ID, as found in the MERCHANDISE_HIERARCHY_TBL or LOCATION_HIERARCHY_TBL. For example, to assign a user to a certain department of merchandise:

  CHAIN COMPANY DIVISION DEPARTMENT merchandise attribute in .xml
  ---
  0 1 123 8765 1 | 123 | 8765
  0 1 22 789 1 | 22 | 789

The information in the three files is loaded into database tables by the bulk loader. (Users and Role Assignments can be added or modified via the application UI. Roles can only be added or modified via the bulkloader.)

The xml Files

The xml schemas and samples of the three required xml files can be found in <installed>/modules/tools/conf.

Table 3–2 User Management xml Files

Schema	Sample	Database Table
user-set.xsd	test_user_set.xml	USERS_TBL
role-set.xsd	test_role_set.xml	ROLES_TBL
role-assignment-set.xsd	test_assignment_set.xml	USER_RESOURCE_ROLE_TBL

Standard Load Prerequisites

Before you run the bulk loader, you must have run the standard load so that the merchandise hierarchy table (ASH_MH_TBL) and the location hierarchy table (ASH_LH_TBL) have been populated. (For more information on the standard load, see the *Oracle® Retail Promotion Intelligence and Promotion Planning and Optimization Operations Guide*).

4

PPO UI Configuration

This chapter provides as overview of the configuration of the Promotion Planning and Optimization UI. For more details, consult the *Oracle® Retail Promotion Intelligence and Promotion Planning and Optimization Configuration Guide*.

This chapter contains the following sections:

Introduction

The PPO product includes a configurable Graphical User Interface (GUI). Several configuration points can be used to modify GUI behavior.

<configroot>

<configroot> is the entry point directory that is used by the application to look up all the configuration files. This value has to be set at the application server (OAS) level. Refer to the *Promotion Installation Guide* for instructions on how to set it. It usually points to <install-dir>/config.

PPO Configuration File

GUI-wide properties are set in the promote.properties file, which is located in <configroot>/promote. This file is pre-populated during the installation process with installation-specific values. The complete list of properties in promote.properties can be found in the Configuration Guide.

Other properties relate to OAS 10.1.3.3 and contain the prefix oas. For example, oas.java.naming.provider.url has to be set to the correct (opmn or non-opmn) url, such as ormi://host:port.

The same is true for the configuration of the User Management application. Its properties are stored in <configroot>/usermanagement/usermanagement.properties.

To specify the time interval at which PI-PPO Planning updates the server session on browser-only user activity (that is, when a user click does not result in a server call), set promotekeepalive.interval to the desired interval (in seconds).

Configuring Display Strings

GUI resources such as labels and error messages are kept in the promoteResources.properties file, which is located in <configroot>/promote.

Note that all internationalization configuration settings are contained in promote.properties and all translated text for display in the UI are contained in promoteResources.properties.

The promoteResources.properties file is organized into functional sections, most of which define information presented to the user that should not be modified. Each section is preceded by a comment that defines either the purpose of the section (such as Error Messages) or the screen in the UI that the section details (such as Promotion Template).

The promoteResources.properties file also provides limited functionality to configure the columns and rows that appear in certain screens. Consult this file for more information about configuring columns and rows.

You can configure the following:

- Which columns or rows are displayed.
- The sort order, descending (-) or ascending (+), of specified columns. For example, sort.Offers=+position sorts the grid in ascending order based on the position column.
- Which metrics are displayed. You can select from two lists of available metrics: BASE_METRIC_COLUMNS, which is a list of common metrics, and METRIC_ COLUMNS, which is a list of additional available metrics.
- Column locking (defined using a pipe symbol).
- User defined fields can be added to the Offer Definition and Notes grids. Different types of fields, such as text or date) can be selected and client-defined labels can specified for the fields.

Configuring Export

The following style sheets are shipped with PI-PPO Planning:

- XmlExportTemplate.xslt, which is used to format the XML output of a promotion
- TxtExportTemplate.xslt, which describes the instructions for the TXT format.
- PromoOfferItemSumTxtTmpl.xslt, which provides promotion offer item details.
- PromoOfferPosSumTxtTmpl.xslt, which provides offer position details.

The location and naming of these files are specified in the promote.properties file, which is located in <configroot>/promote.

The following values must be specified:

Table 4–1 Export Configuration Values

Value	Description
export.root.path	Location of exported files for a push export
export.xml.template	Location of the XML format XSLT style sheet (e.g., <configroot>/config/promote /XmlExportTemplate.xslt)
export.txt.template	Location of the TXT format XSLT style sheet (e.g., <configroot>/config/promote /TxtExportTemplate.xslt)

Values for export.txt.template must be specified for all export types:

- promote.export.txt.template.promotion
- promote.export.txt.template.promotionofferitemsummary
- promote.export.txt.template.promotionofferpositionsummary

Pull Export Configuration

Two files must be configured for a pull export:

- promo-pullclient.properties – defines the defaults for the pull client
- promo-pullclient.log4j.properties – defines the Log4j configuration

These files are located in <installdir>/modules/tools/conf.

Example promo-pullclient.properties File

```
promote.pullclient.servlet.contextroot=promote
promote.pullclient.servlet.appname=export.do
promote.pullclient.protocol=http
promote.pullclient.host=localhost
promote.pullclient.port=8888
promote.pullclient.datemask=MM/dd/yyyy_HH:mm:ss
promote.pullclient.format=xml
promote.pullclient.command=list
promote.pullclient.timeout=10
```

No spaces are permitted for any of the assigned values. The date mask specifies only the input arguments format. The output format is specified in promote.properties.

Integration with Promotion Intelligence

The following configuration points must be set so that Promotion Intelligence reports can be open from PI-PPO Planning.

Auto Authentication Flag

The promote.properties file contains an auto-authentication flag called promote.report.auto_auth. The values for the flag are **true** and **false**.

When the flag is set to true, the Promo Planning/Intelligence integration uses the currently logged-in user's name and password when logging into MicroStrategy.

Report Links Configuration

Report mapping and report links must be defined in <configroot>/promote/promote-config.xml. A sample file is populated during the installation procedure. The XML schema definition file is located in <OAS-dir>/j2ee/home/applications/promote/xmlSchema/promote.xsd.

This configuration includes:

- The MicroStrategy server DNS name, port, protocol, and webapp name
- The organization of the MicroStrategy reports into groups and the list of reports that are included in each group
- Label displays
- Resource file mapping information

The following XML attributes are used in promote.xml:

- Connect attributes used in the construction of the URL for all links (protocol://server:port/webapp_path).
- Each reporting group has its own node. The name is used for the resource file mapping. The report request uses the param tag.
- For all report tags in group, sub-nodes are created in the GUI for the reporting area, using name, params and the common configuration from connect tag. A group with no reports does not have sub-nodes.
- If there is no params tag inside the group or report tag, then no link is provided.
- Groups cannot be nested inside other groups or reports.

Display Strings

The Promotion Planning and Promotion Intelligence GUI properties are located in promoteResources.properties. This file includes locale-specific labels and descriptions. The value name in promote-config.xml is used as the key in the resource file.

Here is an example, using "My Reports":

label.report.MyReports.name=My Reports

label.report.SharedReports.name=Shared Reports

label.report.SharedReports.AdPageAllocation.name=Ad Page Allocation

label.report.SharedReports.AdRoi.name=Ad ROI

Debug Messages

The log files are located in <configroot>/promote/promote.log4j.properties. The location of the file and the debug level can both be modified. If changes are made to these values, the application server must be restarted.

5

Staging and Loading Data

This chapter provides details about the staging and loading of data into the PI-PPO application. For details about the standard interface as well as configuration details, see the *Oracle® Retail Promotion Intelligence and Promotion Planning and Optimization Operations Guide*.

This chapter contains the following sections:

Data Standard Interface

The standard interface chapter of the *Oracle Retail Promotion Intelligence and Promotion Planning and Optimization Operations Guide* discusses the specific data and formatting requirements for the standard interface. Use these interface descriptions as the model for creating the required text files for the database.

Standard Load Process

The application provides two scripts that manage the process to stage, transform, and load data into the target database tables in the database. The data must be provided in flat files that meet the standard interface specifications. The variable length data in the files should be pipe-delimited. The files should be named to correspond to the names of the matching specification tables. For example, the calendar file should be named in a meaningful way (such as cal.txt) to correspond to ASH_CAL_TBL. No specific file extension is required for the input files.

The two scripts are located in %INSTALLATION_DIRECTORY%/modules/tools/bin. The first script, **pl_stage_file.sh**, stages the data from the flat files into the staging tables. The second script, **pl_load_data.sh**, loads the staged data into the database. These two scripts are used if you need to customize the load dependency tree.

Each script contains options that can be customized. You can customize the options in the following ways (which are listed in order of precedence, with the command line having the highest precedence):

- Using the command line options
- Setting the customization values as environment variables in env.sh
- Setting the customization values in the user's environment

If you do not need to customize the load dependency tree, you can use the following two scripts:

- **pl_stage_client.sh** <*full_path_to_product_directory*> *DatasetFilename*
- **pl_load_client.sh** <*full_path_to_product_directory*>

The **pl_stage_client.sh** script calls **pl_stage_file.sh**. The **pl_load_client.sh** script calls **pl_load_data.sh**.

Environment Customization File

Here is an example of the environment customization file (**env.sh**):

```
#This is the environment customization file.
#Please define all customization values here.

#The mail client and address to send all messages to:
#MAIL=mailx
#REPORT_ADDRESS=error_mail@your_domain.com

#Number of parallel processes to run load procedures:
PARALLEL=2

#Directory with data control files:
#CONTROLDIR=/ASHschema/controlfiles

#Directory to store logs:
#LOGDIR=/tmp/load_logs

#Directory to move old logs to.
#If this variable is not set, the logs will be overwritten.
This folder is not required to exist and will be created at the time
#of archiving the logs.
#
#If all old logs should be preserved, it is possible to
#archive the files into a new unique folder, such as:
#LOGDIR_ARCHIVE=
#/tmp/load_logs/archived_logs_'date +%Y%m%d_%H%M_%S'
#
#If only the archive of the previous run is important, then
#archive the files into the same folder, such as:
#LOGDIR=/tmp/load_logs/archived_logs

#Number of errors to allow during load
ERROR_THRESHOLD=50
```

Staging Script: pl_stage_file.sh

Usage: pl_stage_file.sh *[OPTION]... [FILE]...*
Loads the files into the database.

Options:

Table 5–1 pl_stage_file.sh Options

-a DIR	--logdir_archive=DIR	directory to archive old log files
-c DIR	--controldir=DIR	directory with data control files

Table 5–1 (Cont.) pl_stage_file.sh Options

-e NUM	--errorthreshold=NUM	number of errors to allow in load (for DB2, it is a warning threshold)
-l DIR	--logdir=DIR	directory to store logs
-r DIR	--configroot=DIR	configuration root directory
-f FILENAME	--paramfile=FILENAME	filename of DB parameter file in configroot
-h	--help	displays help and exits

Load Script: pl_load_data.sh

Usage: pl_load_data.sh *[OPTION]... [LOADPROCEDURE]...*
Runs the load procedures in the database.

Options:

Table 5–2 pl_load_data.sh Options

-a DIR	--logdir_archive=DIR	directory to archive old log files
-e NUM	--errorthreshold=NUM	number of errors to allow in load (overwrites the procedure's default limit)
-l DIR	--logdir=DIR	directory to store logs
-r DIR	--configroot=DIR	configuration root directory
-f FILENAME	--paramfile=FILENAME	filename of DB parameter file in configroot
-rlo	--runloadonly	do not run pre and post step
-ne	--noexit	does not exit with a value
-h	--help	displays help and exits

Standard Load Error Handling

The Standard Load verifies the records in each staging table. Each record that fails the verification is removed from the staging table and placed in another table so that the load can continue and so that the failed records can be reviewed.

If a load procedure fails and the threshold is exceeded, you will see the message "The specified error threshold has been exceeded for this load procedure." If this occurs, you should correct the existing data problem and re-run the load procedure as well as any child load procedures (as shown in "Standard Load Procedures Order" on page 5-4).

The table containing the failed records is assigned a name that corresponds to the associated staging table. For example:

Table 5–3 Failed Records Table Names

Staging Table	Failed Record Table
ASH_CAL_TBL	ASH_CAL_TBL_BAD
BEECH_OFFER	BEECH_OFFER_BAD

The "BAD" table into which the failed records are inserted has the same structure as the corresponding staging table with the addition of the following four columns:

Table 5–4 Bad Table Columns

Column Name	Description	Data Type	Maximum Length	Nullable (Y/N)
ERROR_ROWID	The row ID that corresponds to the row ID in the staging table	Row ID		N
ERROR_CODE	The code for the verification	Integer		N
ERROR_DESC	Description of the error	String	1000	
ERROR_TIME	The time the error occurred	Timestamp		N

It is possible to place a threshold on the number of failed records in any staging table that will trigger a termination of the load. The default threshold values are hard-coded into the application. In order to customize the threshold values, you must create a properties file and load it into the application.

Standard Load Procedures Order

The standard load should execute in the following order:

1. The staging of:
   - ASH_CAL_TBL
   - BEE_PERIODS_ATTR_TBL
   - ASH_CP_TBL
   - ASH_LHL_TBL
   - ASH_MHL_TBL
   - ASH_LHRENAME_TBL
   - ASH_LH_TBL
   - ASH_MHRENAME_TBL
   - ASH_MH_TBL
   - BEE_IMAGE
   - BEE_OFFER
   - BEE_USER_DEFINED_TYPE
   - BEE_USER_DEFINED_VALUE
   - BEE_VEHICLE_ATTR
   - BEE_VEHICLE
   - CLIENT_HIERARCHY_ACTIONS_TBL
   - STAGE_MH_ATTRS_TBL
   - ASH_LH_ATTRS_TBL
   - BEE_STORE_SETS
   - BEE_STORE_SUBSETS
   - BEE_STORE_SUBSET_ASGN

- BEE_STORE_SET_PRICE
- BEE_PR_LIKE_LOCATION
- BEE_PR_LIKE_MERCHANDISE
- BEE_SKU_LIST
- BEE_SKU_LIST_ITEM
- BEE_PROMO_ALLOC
- BEE_PROMO_CAMPAIGN
- BEE_PROMO_OFFER_ATTR
- BEE_PROMO_OFFER_MERCH
- BEE_PROMO_OFFER
- BEE_PROMO_STORE
- BEE_PROMOTIONS
- BEE_PROMO_FRCSTR
- ASH_PARAMETER_VALUES_TBL
- ASH_SEASONALITY_MAPS_TBL
- ASH_SEASONALITY_VALUES_TBL
- BEE_FUTURE_PRICE_COST
- BEE_MB_DETAIL
- WK_HIST_SALES_INV

2. These load procedures:
   - com.profitlogic.db.birch.LoadCalendars
   - com.profitlogic.db.birch.LoadLHKeyRename
   - com.profitlogic.db.birch.LoadMHKeyRename
   - com.profitlogic.db.birch.LoadMerchandiseHierarchy
   - com.profitlogic.db.birch.LoadLocationHierarchy
   - com.profitlogic.db.birch.LoadMHTbl
   - com.profitlogic.db.birch.LoadLHTbl
   - com.profitlogic.db.birch.LoadTClose
   - com.profitlogic.db.birch.LoadLTClose
   - com.profitlogic.db.beech.LoadTypeMaster
   - com.profitlogic.db.beech.LoadValueMaster
   - com.profitlogic.db.beech.LoadOfferMaster
   - com.profitlogic.db.beech.LoadImageMaster
   - com.profitlogic.db.beech.LoadVehicleMaster
   - com.profitlogic.db.beech.LoadVehicleAttributeMaster
   - com.profitlogic.db.birch.LoadMerchandiseAttributes
   - com.profitlogic.db.birch.LoadLHAttributes

- com.profitlogic.db.beech.LoadPromoLikeLocation
- com.profitlogic.db.beech.LoadPromoLikeMerchandise
- com.profitlogic.db.beech.LoadSkuListMaster
- com.profitlogic.db.beech.LoadSkuListItemMaster
- com.profitlogic.db.beech.LoadSkuListCleanupMaster
- com.profitlogic.db.beech.LoadStoreSets
- com.profitlogic.db.beech.LoadStoreSubsets
- com.profitlogic.db.beech.LoadStoreSubsetAssignments
- com.profitlogic.db.beech.LoadStoreSetPrice
- com.profitlogic.db.beech.LoadPromoCampaign
- com.profitlogic.db.beech.LoadHistoricPromo
- com.profitlogic.db.beech.LoadPromoVehicle
- com.profitlogic.db.beech.LoadPromoVehiclePage
- com.profitlogic.db.beech.LoadPromoVehicleLocation
- com.profitlogic.db.beech.LoadPromoVehicleAlloc
- com.profitlogic.db.beech.LoadPromoOffer
- com.profitlogic.db.beech.LoadPromoOfferMerchandise
- com.profitlogic.db.beech.LoadPromoVehiclePagePosOff
- com.profitlogic.db.beech.LoadPromoVehiclePagePositionOfferAttribute
- com.profitlogic.db.beech.LoadPromoForecaster
- com.profitlogic.db.birch.LoadParameters
- com.profitlogic.db.birch.LoadSeasonalities
- com.profitlogic.db.beech.LoadFuturePriceCost
- com.profitlogic.db.walnut.CreateMerchSummaryTbl
- com.profitlogic.db.walnut.CreatemerchPromoMBSummaryTbl
- com.profitlogic.db.walnut.CreateMerchMapView
- load_weekly_history_data.load_at_all_levels
- pr_load_sales_price
- com.profitlogic.db.beech.LoadMbDetail
- pr_promoitem_load
- pr_promolocation_load
- pr_process_mb_detail
- pr_load_promo_mb_summary.load_all
- pr_process_apc_summary
- pr_create_store_count_sum
- pr_analytics_pkg.refresh_effective_price

Standard Load Steps

Each procedure consists of the following sub-procedures:

1. Setup
2. Pre-load Verification. All n processes are run in parallel.
3. Finish Pre-load Verification.
4. Load. All n processes are run in parallel.
5. Post-load Verification. All n processes are run in parallel.
6. Finish Post-load Verification.
7. Tear-down.

6

Database Configuration

This chapter provides technical details about the database configuration. It contains the following sections:

Summary Configurations

Several configurations must be included in ASH_CP_TBL. These configurations specify the level of aggregation in the merchandise hierarchy that the application and the RDM require.

Table 6–1 Summary Configurations

INTERSECT_NAME	MERCHANDISE_ LEVEL	LOCATION_ LEVEL	Description
PROMOTE_TAE	SKU	DISTRICT	Identifies the Level at which TAE output is produced.
PROMOTE_DETAIL	SKU	STORE	Identifies the Level at which POS data is expected. It is assumed to be the STORE level.
PROMOTE_SUMMARY_1	CLASS	STORE	Identifies the Merchandise and Location levels of the first level of the summary.
PROMOTE_SUMMARY_2	DEPARTMENT	STORE	Identifies the Merchandise and Location levels of the second level of the summary.
PROMOTE_SUMMARY_3	DIVISION	STORE	Identifies the Merchandise and Location levels of the third level of the summary.
PROMOTE_AFFINITY_LEVEL	CLASS	CHAIN	The level of calculation of the APE summary.
PROMOTE_APC	CLASS	REGION	The level of calculation of the APC summary.
PROMOTE_ANALYSIS	SKU	COUNTRY	

Table 6–1 (Cont.) Summary Configurations

INTERSECT_NAME	MERCHANDISE_ LEVEL	LOCATION_ LEVEL	Description
PROMOTE_SCORECARD_ SUMMARY_1	SUBCLASS	STORE	Specifies the level of aggregation from the MH that is used to generate the totals for the scorecard by the MH.
PROMOTE_SCORECARD_ SUMMARY_2	CLASS	STORE	Specifies the level of aggregation from the MH that is used to generate the totals for the scorecard by the MH.
PROMOTE_MIN_LCD	DEPT	CHAIN	Defines the lowest level of the hierarchy that is available for display in the UI.
PROMOTE_PROMO_OFFER_ MH_SUMMARY	DEPT	STORE	Specifies the level of aggregation from the MH that is used to generate the totals for the Scorecard by Offer/Dept report.
PROMOTE_SCORECARD_ MERCH_OFF_AMT_SUMM_3	DEPT	STORE	Specifies the level of aggregation from the MH that is used to generate the totals for the Scorecard by MH and Offer Amt.
PROMOTE_SCORECARD_ MERCH_OFF_AMT_SUMM_2	CLASS	STORE	Specifies the level of aggregation from the MH that is used to generate the totals for the Scorecard by MH and Offer Amt.
PROMOTE_SCORECARD_ MERCH_OFF_AMT_SUMM_1	SUBCLASS	STORE	Specifies the level of aggregation from the MH that is used to generate the totals for the Scorecard by MH and Offer Amt.
PROMOTE_TAE_NONAD_ PART_LEVEL_1	SUBCLASS	CHAIN	Specifies the level of aggregation from the MH that is used for the TAE non-ad metrics. It is also used by the TAE process to identify the starting MH level that should be used to generate its output.
PROMOTE_TAE_NONAD_ PART_LEVEL_2	CLASS	CHAIN	Specifies the level of aggregation from the MH that is used for the TAE non-ad metrics. It is also used by the TAE process to identify the starting MH level that should be used to generate its output.
PROMOTE_TAE_NONAD_ PART_LEVEL_3	DEPT	CHAIN	Specifies the level of aggregation from the MH that is used for the TAE non-ad metrics. It is also used by the TAE process to identify the starting MH level that should be used to generate its output.
PROMOTE_MIN_BL_AGGR_ LEVEL	SUBCLASS	CHAIN	Specifies the lowest level that aggregated baseline data should be calculated for.
PROMOTE_MAX_BL_AGGR_ LEVEL	CHAIN	CHAIN	Specifies the highest level that aggregated baseline data should be calculated for.

The following non-PI and PPO entries are required for compatibility reasons:

Table 6–2 Intersect Names

INTERSECT_NAME	MERCHANDISE_LEVEL	LOCATION_LEVEL
OPTIMIZATION	SKU	STORE
WORKSHEET	DEPARTMENT	CHAIN
SALES	SKU	CHAIN
CLUSTER	CHAIN	CHAIN
DEFAULTLEVEL	CHAIN	CHAIN

The Cust_Parameter_Levels PL/SQL package provides an interface to the following values. For examples, see "IR Views" on page 6-3.

- getMerchandiseLevelDesc(in_intersect_name)
- getMerchandiseLevelSqc(in_intersect_name)
- getLocationLevelDesc(in_intersect_name)
- getLocationLevelSqc(in_intersect_name)

CLIENT_HIERARCHY_ACTIONS_TBL

The Client_Hierarchy_Actions_Tbl must be modified according to the levels of inventory aggregation required.

Table 6–3 Actions for Hierarchy Actions Table

Action Type	Action Name	Action Level Name	Action Level	Hierarchy Type	Description
SUITE	STORE	STORE	0	LOCATION	Identifies the level in the location hierarchy corresponding to physical STORE
PROMOTE	HIST_AGG_ MERCH_LEVEL_0	HIST_AGG_LEVEL_0	0	MERCHANDISE	Identifies the Lowest Merchandise Level at which History should be persisted
PROMOTE	HIST_AGG_LOC_ LEVEL_0	HIST_AGG_LEVEL_0	0	LOCATION	Identifies the Lowest Location Level at which History should be persisted

IR Views

The following views must be modified according to the level of summary needed. The view creation scripts are located in <installdir>/modules/Database/ROSEWOODSchema/install/oracle /ROSEWOODSchema/dictionary/views_ir. Example (found in the supplied sample KSInc dataset) are located in <installdir>/modules/pce/sample/ir_views/oracle.

Table 6–4 Modifying Inference Rules

View	Description
IR_OLF_CANDIDATES_VW	This view defines what merchandise nodes are the source of Offer level Forecast (OLF) aggregates.
IR_OLF_NODES_VW	This view defines what merchandise receives an OLF forecast.
IR_PBL_ATTRS_PURCHASE_X_VW	This view defines how the merchandise hierarchy is divided for the predicted baseline calculation.
IR_PR_DEFAULT_PRICE_ZONE_VW	This required view must be manipulated so that it references the primary store set for which pricing data is provide via the Store Set Prices interface. it is used to generate price data for other store sets or location hierarchy levels.
IR_PR_LOCATION_SUMMARY_X_VW	These views map each location summary level to its SKU.
IR_PR_MERCH_SUMMARY_X_VW	These views map each merchandise summary level to it SKU.
IR_PR_PROMO_ITEM_VW	This view exposes the attributes needed by the PCE for modeling.
IR_PR_PROMOTIONS_VW	This view exposes the attributes needed by the PCE for modeling.
IR_TREND_CANDIDATES	

Update the views using the following guidelines:

For ir_pr_merch_summary_X_vw. These views map each merchandise summary level to its SKUs. For example:

- CREATE OR REPLACE VIEW ir_pr_merch_summary_3_vw AS SELECT hierarchy3_pid parent_pid, merchandise_id, mod(merchandise_id,10) seas_cd FROM merchandise_tbl WHERE level_sqc = 6
- CREATE OR REPLACE VIEW ir_pr_merch_summary_4_vw AS SELECT hierarchy4_pid parent_pid, merchandise_id, mod(merchandise_id,10) seas_cd FROM merchandise_tbl WHERE level_sqc = 6
- CREATE OR REPLACE VIEW ir_pr_merch_summary_5_vw AS SELECT hierarchy5_pid parent_pid, merchandise_id, mod(merchandise_id,10) seas_cd FROM merchandise_tbl WHERE level_sqc = 6

For ir_pr_location_summary_X_vw. These views map each location summary level to its SKUs. For example:

- CREATE OR REPLACE VIEW ir_pr_location_summary_1_vw AS SELECT hierarchy1_lid, location_id FROM location_tbl WHERE level_sqc = Cust_ Parameter_Levels.getLocationLevelsqc('PROMOTE_ANALYSIS')
- CREATE OR REPLACE VIEW ir_pr_location_summary_7_vw AS SELECT hierarchy7_lid parent_lid, location_id FROM location_tbl WHERE level_sqc = Cust_Parameter_Levels.getLocationLevelSqc('PROMOTE_ANALYSIS')

PR_DB_PARAMS

When PPO and RDF are deployed in the same environment, each application uses the base demand forecast generated by RDF for forecasting. The field PREDICT_BASELINE_SOURCE_TYPE, which must be removed if the value generated by RDF is not used is part of the PR_DB_PARAMS table shown in Table 6–5, " PR_DB_PARAMS".

Table 6–5 PR_DB_PARAMS

Field Name	Description
LAST_CREATE_MISSING_PROMO_CT	The date of the last attempt to create missing promotion counts. Only promotions modified after this date can be fixed when the promotion counts are created again.
LAST_PROCESSED_MB_DATE	The last market basket date loaded.
MB_START_DATE	Obsolete.
MB_END_DATE	Obsolete.
DEFAULT_TABLESPACE	The tablespace that holds the tables created by the application.
DEFAULT_INDEX_TABLESPACE	The tablespace that holds the indexes created by the application.
PROMOTE_SCORECARD_TOP_NONOD	The number of records stored by the TAE NonAd contributor feature. This affects Scorecard reports.
APE_DFLT_NODE_DESCR	The name of the default APE node that receives miscellaneous affinity numbers during forecasting.
LAST_LOAD_MISSING_PROMO_SUM	The date of the last attempt to create missing promotion summaries. Only promotions modified after this date can be fixed when the promotion summaries are created again.
PREDICT_BASELINE_SOURCE_TYPE	The source data for externally provided predicted baseline values. This field must be removed if the external predict baseline is not supported.

7

Managing the Environment

This chapter provides reference material for systems administrators and others responsible for maintaining the Promotion Intelligence and Promotion Planning and Optimization application. It contains the following sections:

Locating Host Specific Parameters

The following files contain host specific settings. Many of these default to "localhost" but may be different in a production environment.

```
$installdir/config/SIT/sit-config.properties
$installdir/config/promote/promote.properties
$installdir/modules/tools/bin/storesetupdater.sh
$installdir/modules/tools/conf/jndi.properties
$installdir/modules/tools/conf/promote-cmdline.properties
$installdir/modules/pce/sample/deploy.sh
$installdir/modules/pce/sample/images/install_is.sh
$installdir/config/SIT/server-list.xml
```

Monitoring the PI and PPO Server

Besides the existing Oracle Application Server monitoring facilities for monitoring applications, the following URL can be used to provide a pulse check of the web component to ensure that it is running properly:

```
http://${WL_HOST}:${WL_PORT}/promote/service.do?command=status
```

If everything is running properly, this command will return the string "OK"

Monitoring Promotion Calc Engine (PCE) Servers

To monitor the PCE servers, execute the following Linux command on each PCE host:

```
bash$ pceserver.sh -query status
Querying PCE server status....
PCE server is up running.
```

If the PCE server is not responding, use the following command to retrieve the server load.

```
bash$ pceserver.sh -query load
Querying PCE server load....
PCE server is running 0 applications.
```

For a listing of additional PCE server tools, run the pceserver.sh command with no options. A list of available commands will be provided.

Monitoring Batch Jobs

The following section provides an introduction to batch jobs.

Overview

Batch operations are composed of tasks, steps, and agents.

- Tasks – tasks are aligned with the major batch operations. These must be scheduled using a management tool and typically are enabled on one host.
- Steps – each task is composed of a number of steps, such as waiting for a file, unzipping it, and processing it.
- Agents – tasks and steps are managed as a process outside the application server. Agents run within each application server.

Tasks

The following are the tasks currently configured in the system:

- weekly.load – this task is responsible for loading and processing the appropriate data passed to the application each week.
- weekly.restate – this task also runs weekly, but it is meant specifically for certain data files separate from weekly.load.
- nightly.load – running nightly, this task loads the data that needs to be processed on a nightly basis.
- nightly.export – this task also runs nightly and is responsible for preparing data for export back to the client.

Steps

Each task defines a list of steps required to execute the task. This list of steps is task specific and can be found within $AUTOMATION_BASE/config/<task>/Proccess.steps.

The Job Log Table

Weekly and nightly feeds log their activities to the PR_MGMT_TASK table. The table below lists the columns in this table.

Table 7–1 PR_MGMT_TASK Table Attributes

Column Name	Data Type	Nullable	Data Default	Column ID	Primary Key	Comments
RUN_ID	Number (32,0)	No			1	
TASK_NAME	Varchar2 (40 byte)	No			2	
TASK_ID	Number (32,0)	Yes			3	
STEP_NAME	Varchar2 (80 byte)	No			4	
STEP_START	Date	No			5	
STEP_END	Date	Yes			6	
STEP_ DURATION	Number (5.0)	Yes			7	
STEP_RESULT	Number (3,0)	No			8	
HOST	Varchar2 (200 byte)	No			9	
ARGUMENTS	Varchar2 (2048 byte)	Yes			10	
DESCRIPTION	Varchar2 (2048 byte)	Yes			11	

The application writes to this table with each step of its nightly or weekly batch cycle. The following is a brief example of what is written:

Table 7–2 Sample Nightly Batch Data within the PR_MGMT_TASK Table

RUN_ID	TASK_ NAME	TASK_ID	STEP_NAME	STEP_RES	HOST	DESC
1	nightly.load		start	0	dev-app-101	Michaels Arts and Crafts: START in nightly.load (Run ID 20071018-1312) Automation log file is at ../mdc/operations /logs/automation /nightly.load. 200710181312.log All the logs are at ../mdc /operations/logs
2	nightly.load	1	generic/ wait.for.files.sh	0	dev-app-101	Running step 1/21 (generic/wait.for. files.sh) Step ended

Table 7–2 (Cont.) Sample Nightly Batch Data within the PR_MGMT_TASK Table

RUN_ID	TASK_ NAME	TASK_ID	STEP_NAME	STEP_RES	HOST	DESC
3	nightly.load	1	update	0	dev-app-101	Beginning nightly processing...
...						
39	nighly.load	1	promote/ load/data.sh	0	dev-app-101	Running step 20/21 (promote/load.data.sh) Step ended
40	nightly.load	1	update	0	dev-app-101	Nightly load complete
41	nightly.load	1	end	0	dev-app-101	Michaels Arts and Crafts: SUCCESS in nightly.load (Run ID 200710181312)

As this example shows, each task is made up of many steps linked together using the TASK_ID column. Each step receives one row in the above table.

Understanding Job Failures

The critical column to watch is STEP_RESULT. If the step succeeds, there will be a zero in this column. A non-zero value represents a failure.

Operators should be alerted when a step fails. The following table illustrates an example failure.

Table 7–3 Sample Task Failures within the PR_MGMT_TASK Table

RUN_ID	TASK_ NAME	TASK_ID	STEP_NAME	STEP_RES	HOST	DESC
1	weekly.load		start	0	dev-app-101	Michaels Arts and Crafts: START in nightly.load (Run ID 20071024-1517) Automation log file is at ../mdc/operations /logs/automation /weekly.load. 200710241517.log All the logs are at ../mdc /operations/logs

Table 7–3 (Cont.) Sample Task Failures within the PR_MGMT_TASK Table

RUN_ID	TASK_ NAME	TASK_ID	STEP_NAME	STEP_RES	HOST	DESC
2	weekly.load	1	generic/ wait.for.files.sh	0	dev-app-101	Running step 1/21 (generic/wait.for. files.sh) Step ended
3	weekly.load	1	fail	2	dev-app-101	Michaels Arts and Crafts: FAILURE in weekly.load (RUN ID 200710241517)

The log file above demonstrates that the client failed to deliver the data files in the specified time window:

```
2007.10.24 15:17:17 Started processing weekly.load on 200710241517 with log at
../mdc/operations/logs/automation/weekly.load.200710241517.log.
2007.10.24 15:17:18 Running step generic/wait.for.files.sh (1/38):
2007.10.24 15:17:18 Step started at 2007.10.24 15:17:18.
2007.10.24 15:18:33 Have NOT received 'weekly.ash_lh_tbl.txt.gz'.
2007.10.24 15:18:33 Have NOT received 'weekly.ash_mh_tbl.txt.gz'.
2007.10.24 15:18:33 Have NOT received 'weekly.stage_mh_attrs_tbl.txt.gz'.
2007.10.24 15:18:33 Have NOT received 'weekly.bee_pr_like_merchandise.txt.gz'.
2007.10.24 15:18:33 Have NOT received 'weekly.bee_promo_offer_attr.txt.gz'.
2007.10.24 15:18:33 Have NOT received 'weekly.bee_promo_offer_merch.txt.gz'.
2007.10.24 15:18:33 Have NOT received 'weekly.bee_promo_offer.txt.gz'.
2007.10.24 15:18:33 Have NOT received 'weekly.bee_promo_store.txt.gz'.
2007.10.24 15:18:33 Have NOT received 'weekly.bee_promotions.txt.gz'.
2007.10.24 15:18:33 Have NOT received 'weekly.wk_hist_sales_inv.txt.gz'.
2007.10.24 15:18:33 Have NOT received 'weekly.bee_mb_detail.txt.gz'.
2007.10.24 15:18:33 Have NOT received 'weekly.data_ready.txt'.
2007.10.24 15:18:33 FATAL: Files are late and abort time has passed. Giving up.
2007.10.24 15:18:33 Step ended at 2007.10.24 15:18:33. Elapsed time is 75 seconds.
2007.10.24 15:18:34 Running procedure 'pr_mgmt.task_fail' with parameters '223, 2
, 'Demo Stores: FAILURE in weekly.load (Run ID 200710241517)''
2007.10.24 15:18:34 Ran procedure 'pr_mgmt.task_fail' successfully.
```

Locating Bad Data

A common failure is bad data received from the client. This can be manifested in several ways. Suppose the client mistakenly puts a header row on the input feed. The following table is a sample task log illustrating the failed step.

Table 7–4 Sample Bad Data Example

RUN_ID	TASK_ NAME	TASK_ID	STEP_NAME	STEP_RES	HOST	DESC
239	weekly.load	226	start	2	dev-app-101	Running step 13/38 (db/oracle/stage.file .sh Step ended

Using SQL Developer, the file that failed can be located as follows:

The failed step was a result of a bad weekly.bee_promotions.txt file. Open the log file and scan it for the failed task as illustrated in the example below:

```
2007.10.24 16:14:35 Running step db/oracle/stage.file.sh (13/38):
2007.10.24 16:14:35 Step started at 2007.10.24 16:14:35.
2007.10.24 16:14:35 Staging logs at
'../mdc/operations/logs/database/stage/weekly.bee_
promotions.txt.200710241614.log'.
2007.10.24 16:14:35 Bad records file at
'../mdc/operations/logs/database/stage/weekly.bee_
promotions.txt.200710241614.bad'.
2007.10.24 16:14:36 Staging file
'../mdc/operations/spool/automation/weekly.load/weekly.bee_promotions.txt' with
control file
'../install/modules/Database/BEECHSchema/install/oracle/BEECHSchema/controlfiles/b
ee_promotions.ctl'...

Load completed - logical record count 23.
2007.10.24 16:14:36 FATAL: sqlldr exited with an error.
2007.10.24 16:14:36 Step ended at 2007.10.24 16:14:36. Elapsed time is 1 seconds
```

The log files above demonstrate that the failure was a number format error in the actual offending record.

Restarting Jobs

Failed weekly batch jobs can be restarted using the $AUTOMATION_BASE/scripts/do.weely.load script.

Jobs can also be restarted at a certain step by providing the line number of the job to the task script. For example, running $AUTOMATION_BASE/scripts/do.weekly.load 7 would run weekly batch starting at step 7, assuming that the files have already been placed in their processing/spool directory.

Model Migration

To migrate a model that has been built from a development system to a production system do the following:

1. Capture the models to be migrated. Process them into intermediate files (.pmmlmv) using the script /install/modules/pce/bin/capture_model.sh.

2. Use a manual process to move the intermediate files from the source system to the target system.
3. Import the intermediate files in the target system and convert them into model files (.pmml). Deploy these migrated models using /install/modules/pce/bin/import_model.sh.

Parallel Processing

The two supported ways to run any process in parallel are:

- Manually run more than one process
- Run more than one process in operations

The Master Script ensures that the tasks identified as part of the configured SQL Templates are registered and executed only after the required pre-processing is complete. The Helper Process ensures that the master has registered the task. If not, it polls for the task in the cron process and the manual process exits without processing the task.

Manual Process

The following generic script acts as helper for the master:

process\scripts\common\process_exec_helper_manual_doit.sh.yapp.

This script is only responsible for executing the tasks in parallel. It checks the task table to identify if any process among the registered tasks is waiting for execution. If such a task is present, it executes the task or else terminates.

For example:

For lift models, the 1.build_lift_model_manual_helper.sh resides under \installdir\modules\pce\bin\build_lift_model. This script executes tasks in parallel with the Main Process. This ensures that only Master Scripts are responsible for registering the list of the tasks for that process and that helpers are only responsible for selecting the task each time.

Running More Than One Process in Operations

The following script should be used when a task is executed as a cron job. The cron job for the helper process should be scheduled to execute a few minutes before the master cron job. The following generic script is provided that acts as helper for the master:

process \scripts\common\process_exec_helper_doit.sh.yapp. This script is only responsible for executing the task runs in parallel. It checks the task table to identify if any process is pending to execute the registered task for the specific process; if present, it executes them. Helper processes wake every few minutes to determine if any registered process can be selected.

This script is only responsible for executing the task runs in parallel. It checks the task table to identify if any process among the registered tasks is waiting for execution. If such a task is present, it is executed. The helper process wakes every few minutes to determine if any registered process can be selected.

For example:

The Tae process has a master and helper script:

Master: \mdc\operations\automation\step.lib\promote\pce\tae.sh

Helper (internally calls process_exec_helper_doit.sh):
\mdc\operations\automation\step.lib\promote\pce\tae_helper_run.sh

The operation configuration steps for the automation scripts mdc\operations\automation\config are:

- Master – weekly.load
- Helper – weekly.helper.load

The master scripts for the model build are:

- Main product – build_lift_model.sh
  \installdir\modules\pce\bin\
- MDC – doit.sh
  \mdc\scripts\6.build_lift_models\4.build_and_measure

The master scripts for tae are:

- Main product – tae_run.sh
  \installdir\modules\pce\bin
- MDC – doit.sh
  \mdc\scripts\4.analyze_history\2.tae

The updates to the SQL templates for customization are:

For the operations that Tae Runs uses:

tae_run_sql_template.txt residing under
\mdc\operations\automation\step.lib\promote\pce\tae.

For the operations MDC Tae Runs uses:

tae_run_sql_template.txt residing under \mdc\scripts\4.analyze_history\2.tae.

Any updates to this template affect the tae runs in the main MDC scripts.

Other Maintenance Activities

The following are additional actions that can be taken to ensure proper maintenance of the production environment.

Starting and Stopping the Server

Use the standard Oracle Application Server facilities to accomplish this.

Starting and Stopping the PCE

Use the following pceserver.sh commands to control and monitor the PCE server:

```
$bash pceserver.sh -start
$bass pceserver.sh -shutdown
```

8

Forecast Accuracy Indicator

This chapter explains the evaluation of a forecast for accuracy. It contains the following sections:

Introduction

The Forecast Accuracy Indicator is an enhancement to the PCE forecast prediction that evaluates the accuracy of a forecast by comparing current forecast data with historical data.

A rule-based decision tree based on a statistical analysis is used in the determination of the forecast accuracy. The decision tree is configured by Analytical Services (AS), using the accuracy.properties file. This file is used to configure the rules and the values used in the decision tree.

The UI displays the results of the accuracy determination.

Configuration

You can configure the forecast accuracy feature as follows:

Use the property **com.netperceptions.kde.rmi.server.RGIndicatorFlag=true** to enable or disable the Forecast Accuracy Indicator in the PCE.

For information about configuring the default thresholds that the UI uses to control the display of Red, Yellow and Green confidences, see the following properties in promote.properties:

- promote.confidence.greenThreshold=70
- promote.confidence.yellowThreshold=30
- promote.confidence.redThreshold=0

See the Merchandise Thresholds standard interface for information about the configuration of different thresholds for different areas of the merchandise hierarchy.

Metrics

This section lists the metrics used by AS to configure the accuracy.properties file. The supported rule operators in this file are:

=, <, >, <=, >=, !=

The metrics listed in Table 8–1, " Model Metrics" use the following abbreviations:

Abbreviation	Definition
XXX	attribute name
Metric String	abbreviation used in rule
CR	hard-coded constant expression
VI	metrics that are pre-evaluated as part of PCE start-up
X	data type
#	data type
Boolean	true/false

Table 8–1, " Model Metrics" contains the metrics used in accuracy.properties

Table 8–1 Model Metrics

Metric ID	Metric Description	Metric String Abbreviation	Type	CR/VI=	Operator
PBL-1	Type of merchandise	PBL_MET.MERCH_TYPE	X	CR=[B/S]	=, !=
PBL-2	Maximum size of baseline window (# of weeks 5 or 9 for example)	PBL_MET.MAX_BL_PERIOD	#	CR=[?]	=, <, <=, >, >=, !=
PBL-3	Actual number of historic baseline weeks used for prediction	PBL_MET.TTL_GOOD_PERIODS	#	CR=[?]	=, <, <=, >, >=, !=
PBL-4	Number of dark weeks from all baseline window weeks	PBL_MET.DARK_PERIOD	#	CR=[?]	=, <, <=, >, >=, !=
PBL-5	Number of promotion weeks from historic baseline window	PBL_MET.PROMO_PERIOD	#	CR=[?]	=, <, <=, >, >=, !=
PBL-6	Number of clearance weeks from historic baseline window	PBL_MET.CLEARANCE_PERIOD	#	CR=[?]	=, <, <=, >, >=, !=
PBL-7	Number of gray weeks from historic baseline window	PBL_MET.GRAY_PERIOD	#	CR=[?]	=, <, <=, >, >=, !=
PBL-8	Average baseline sales of item during historic baseline window	PBL_MET.AVG_BL_SLS	#,#	CR=[?]	=, <, <=, >, >=, !=
PBL-9	Average baseline sales variance of item during historic baseline window	PBL_MET.AVG_BL_SLS_VAR	#,#	CR=[?]	=, <, <=, >, >=, !=
PBL-10	Future clearance indicator	n/a	n/a	n/a	n/a

Table 8–1 (Cont.) Model Metrics

Metric ID	Metric Description	Metric String Abbreviation	Type	CR/VI=	Operator
PBL-11	APC elasticity level of item	PBL_MET.PRICE_ELASTICITY_ LEVEL	#	CR=[?]	=, <, <=, >, >=, !=
PBL-12	APC seasonality level of item	PBL_MET.SEAS_INDX_LEVEL	#	CR=[?]	=, <, <=, >, >=, !=
PBL-13	Seasonality Index of item	PBL_MET.SEAS_INDX	#,#	CR=[?]	=, <, <=, >, >=, !=
PBL-14	APC Price elasticity used for the item	PBL_MET.PRICE_ELASTICITY	#,#	CR=[?]	=, <, <=, >, >=, !=
PBL-15	Number of weeks between forecast as-of-date and ad-date	PBL_MET.WEEKS_TO_AD	#	CR=[?]	=, <, <=, >, >=, !=
MSC-1	Holiday ad?	n/a	n/a	n/a	n/a
MSC-2	Fiscal month of ad date	PBL_MET.FISCAL_MO	#	CR=[?]	=, <, <=, >, >=, !=
MSC-3	Fiscal quarter of ad date	PBL_MET.FISCAL_QUARTER	#	CR=[?]	=, <, <=, >, >=, !=
MSC-4	Error between Monkey Model forecast and PCE forecast.	n/a	n/a	n/a	n/a
MSC-5	Was like item	PBL_MET.LIKE_ITEM_USED_FLG	#	CR=[0/1]	=, !=
MSC-6	Was an offer level forecast used for predict baseline	PBL_MET.AGGR_PBL_USED_FLG	#	CR=[0/1]	=, !=
MDL-1	Lift model level of item	MDL_MET.MERCHANDISE_ LEVEL	#	CR=[?]	=, <, <=, >, >=, !=
MDL-2	Lift model fitting error (MSE in pmml file)	MDL_MET.MSE	#,#	CR=[?]	=, <, <=, >, >=, !=
MDL-3	Lift model R2 (rsquare in pmml file)	MDL_MET.RSQUARE	#,#	CR=[?]	=, <, <=, >, >=, !=
MDL-4	Lift model intercept (intercept in pmml file)	MDL_MET.INTERCEPT	#,#	CR=[?]	=, <, <=, >, >=, !=
MDL-5	Lift model condition number (condNum in pmml file)	MDL_MET.CONDITION_ NUMBER	#,#	CR=[?]	=, <, <=, >, >=, !=
MDL-6	Lift model F-statistic (fvalue in pmml file)	MDL_MET.FVALUE	#,#	CR=[?]	=, <, <=, >, >=, !=
MDL-7	Lift model p-statistic (pvalue in pmml file)	MDL_MET.PVALUE	#,#	CR=[?]	=, <, <=, >, >=, !=
MDL-8-1	Is current value < minimum historic value for each numerical predictor, where XXX is discount/price_ratio	MDL_MET.XXX.NUM_MIN	#,#	VI	=, <, <=, >, >=, !=
MDL-8-2	Is current value < maximum historic value for each numerical predictor	MDL_MET.XXX.NUM_MAX	#,#	VI	=, <, <=, >, >=, !=
MDL-8-3	STDDEV of current value from mean historic value for each numerical predictor	MDL_MET.XXX.NUM_STDDEV & MDL_MET.XXX.NUM_MEAN	#,#	VI	=, <, <=, >, >=, !=

Table 8–1 (Cont.) Model Metrics

Metric ID	Metric Description	Metric String Abbreviation	Type	CR/VI=	Operator
MDL-8-4	P-Value of predictor coefficient for each numerical predictor	MDL_MET.XXX.PVALUE	#,#	VI	=, <, <=, >, >=, !=
MDL-8-5	Standard error of predictor coefficient (sbk in pmml file) for each numerical predictor	MDL_MET.XXX.STD_ERROR	#,#	VI	=, <, <=, >, >=, !=
MDL-8-6	Variance inflation factor of predictor coefficient (vif in pmml file) for each numerical predictor	MDL_MET.XXX.VIF	#,#	VI	=, <, <=, >, >=, !=
MDL-8-7	The value of the predictor coefficient for each numerical predictor	MDL_MET.XXX.COEFFICIENT	#,#	VI	=, <, <=, >, >=, !=
MDL-9-1	Value of the predictor coefficient for categorical variables	MDL_MET.XXX.COEFFICIENT	#,#	VI	=, <, <=, >, >=, !=
MDL-9-2	P-value of the predictor coefficient for categorical variables	MDL_MET.XXX.PVALUE	#,#	VI	=, <, <=, >, >=, !=
MDL-9-3	Standard error of the predictor coefficient (sbk in pmml file) for categorical variables	MDL_MET.XXX.STD_ERROR	#,#	VI	=, <, <=, >, >=, !=
MDL-9-4	Variance inflation factor of the predictor coefficient (vif in pmml file) for categorical variables	MDL_MET.XXX.VIF	#,#	VI	=, <, <=, >, >=, !=

9

Reports

This chapter describes the reports that are available and provides details about configuring reports. It contains the following sections:

Introduction

Use the Standard Reports GUI to create and share new reports. All reports are based on a standard template. Several pre-defined reports are available, including reports that provide information on General Trends, Product Categories, and Individual Products.

Available Reports

PI-PPO provides the following reports. Because of rounding issues, the calculations in reports may be inaccurate. Metrics are calculated using full precision numbers; however, the reports only display two decimals. Validating these values manually using the metrics displayed in the reports can result in different results that are caused by the rounding. To prevent this, increase the number of decimals displayed in the reports.

- Affinity (Pull) – This report provides information about the affinity products or items that tend to sell well with other items. This report shows the affinity relationship over a longer period of time. This report contains metrics similar to the Affinity report, but also contains an additional column, "Pull Indicator", that defines whether the relationship Likely, Unlikely, or Inconclusively drives sales between two items. It displays the affinity rules produced by the ARM application.
- Affinity Report – This report provides information about all affinity products and items that tend to sell with other items. It displays all of the affinity rules produced by the ARM application.
- Audit Trail Report – This report tracks changes made to a promotion at the user, date/time, and offer level. It also tracks changes to offers that affect the forecast, including Promotion dates, promotion phase changes, edits to vehicle types, added and deleted offers, offer status changes (submissions and approvals), and

any offer changes that affect the forecast (e.g. criteria, offer type, offer amount, demand drivers, forecast overrides, and position changes).

- Event Scorecard By Class Report – This report provides an analysis of the effect that individual classes have on the success of particular events. Viewers of this report also have the option of drilling into the metrics for Incr Allocated Non-Ad Sales, Incr Allocated Non-Ad GM, and Incr Allocated Non-Ad Units.
- Event Scorecard By Class/Offer Amount – This report provides an analysis of the effectiveness of different offer types and amounts. This report summarizes the offer type–amount performance within a class across multiple events. It enables a merchant to determine whether a %off discount has been more effective than a price point even if the effective discount was equivalent. Similarly, it can help determine whether a specific offer amount 25% or 30% off of a given offer type has been more effective historically. Viewers of this report also have the option of drilling into the metrics for Incr Allocated Non-Ad Sales, Incr Allocated Non-Ad GM, and Incr Allocated Non-Ad Units.
- Event Scorecard By Department/Offer Amount – This report provides an analysis of the effectiveness of different offer types and amounts. This report summarizes the offer type–amount performance within a department across multiple events. It enables a merchant to determine whether a %off discount has been more effective than a price point even if the effective discount was equivalent. Similarly, it can help determine whether a specific offer amount 25% or 30% off of a given offer type was more effective historically.
- Event Scorecard By Item Report – This report provides an analysis of the effect that individual items have on the success of particular events.
- Event Scorecard By Offer/Department Report – This report provides an analysis of the effect that each offer/department combination has on the success of particular events.
- Event Scorecard By Sub-Class Report – This report provides an analysis of the effect that individual Sub-classes have on the success of particular events. Viewers of this report also have the option of drilling into the metrics for Incr Allocated Non-Ad Sales, Incr Allocated Non-Ad GM, and Incr Allocated Non-Ad Units.
- Event Scorecard by Sub-Class/Offer Amount – This report provides an analysis of the effectiveness of different offer types and amounts. This report summarizes the offer type–amount performance within a sub-class across multiple events. It enables a merchant to determine whether a %off discount has been more effective than a price point even if the effective discount was equivalent. Similarly, it can help determine whether a specific offer amount 25% or 30% off of a given offer type has been more effective historically.
- Forecast Accuracy Report – This report compares the system and user (if one exists) predicted forecasts from a promotion created in Promotion Planning and Optimization against the sales results within Promotion Intelligence. Analysis is done only at the event level.
- Forecast Exception Report – This report provides information about changes in an offer's total forecast units. The changes in the forecast could be the result of system re-forecast process or a manual re-forecast by any user. The system has the ability to track forecast changes by units, sales or margin (one at a time).
- Model Accuracy Scorecard – This report provides information to help evaluate the efficiency of the predictive model.
- Overlapping SKUs Report – This report identifies cases where the same SKU exists in different offers in the same event. The specific offers and duplicate SKUs are

listed so that the user can correct the offers and avoid a pricing conflict in which the same SKU is promoted at different prices.

- TAE Assessment Report – This report provides information that can be used to evaluate the accuracy and completeness of the data generated by TAE.

Configuration Report Details

This section provides details about the two reports that are used during the configuration process. Details about the other reports can be found in the *Promotion Intelligence User Guide*.

Model Accuracy Scorecard Report

Use this report to evaluate the efficiency of the predictive model before you deploy it in the production environment. To test the model and produce this report, the PCE generates sales predictions for past ad events. Then, the application compares those predictions to the actual sales data gathered during the past ad event. The results are used to generate measures of error that can be used to evaluate the accuracy of the predictive model.

Report Prompts and Display

The report uses selected segments from the product hierarchy. The resulting report measures error only from the predictions that were generated for products that belong to the selected segment. This can be to evaluate how well a specific model predicts for a given segment. Results are grouped by the Model_ID that was used to produce the item/store level prediction.

Table 9–1 Model Accuracy Scorecard Report

Metric Number	Metric	Description
1	Model	The model's name from rdm
2	Focus Item	Focus Item (SKU Level)
3	Location	Location (Store Level)
4	Actual Quantity	Actual number of units sold
5	Predicted Quantity	Predicted number (generated by the model) of units sold
6	Chain Level Error	Mean Absolute Percent Error calculated from chain level aggregated units sold
7	Store Level Error	Total of Mean Absolute Percent Error calculated at the item/store level

TAE Assessment Report

Use this report to check the accuracy and completeness of TAE results. This report facilitates evaluating the data generated by the TAE process before integrating the data into the dataset. The report executes against the temporary table, PR_TAE_TEMP_METRIC.

Report Prompts and Display

The report prompts you to select the run ID, the merchandise hierarchy, and the ad event. Multiple data runs are identified by separate run IDs.

Table 9–2 TAE Assessment Report

Metric Number	Metric	Description
1	Run ID	Unique ID for TAE execution.
2	Focus Item	Item ID and description.
3	Promotion	Promotion description.
4	Ad Item MB Count	Count of ad market baskets that contain at least one focus item.
5	Item Baseline MB Count	Count of baseline market baskets that contain the item.
6	Index: Ad MBs to Baseline MBs	Count of item ad market baskets compared to the count of item baseline market baskets.
7	Status	OK. Bl_subst_code = 0 Substituted. Bl_subst_code = 1 No Result. Bl_subst_code in (2,3,4)
8	Substitute Item	Item used in substitution.

Changing MicroStrategy Summary Levels

PI-PPO reports use a default level (Department or MH level 4) of analysis. To change this level, do the following (demonstrated changing Summaries from Department (MH level 4) to Division (MH level 3):

1. Edit the Merchandise Level in Schema Objects/Attributes/Product Attributes
2. Select PI_ID and click Modify.

Summary Configurations

The PROMOTE_PROMO_OFFER_MH_SUMMARY parameter is a value for the INTERSECT_NAME in the ASH_CP_TBL standard interface. It specifies the level of aggregation for the merchandise hierarchy that is used to generate the totals for the scorecard by Offer/Department. The merchandise level should be the level that corresponds to the Department. The location level is not relevant to this aggregation.

The following summary configuration parameters specify the level of aggregation from the merchandise hierarchy that are used to generate the totals for the scorecard by merchandise hierarchy and offer amount.

INTERSECT_NAME	MERCHANDISE_LEVEL	LOCATION_LEVEL
PROMOTE_SCORECARD_MERCH_OFF_AMT_SUMM_3	DEPT	STORE
PROMOTE_SCORECARD_MERCH_OFF_AMT_SUMM_1	SUBCLASS	STORE
PROMOTE_SCORECARD_MERCH_OFF_AMT_SUMM_2	CLASS	STORE

MB Counts

The MB count is generated under the assumption that no overlap exists between promotions in the same event and that no overlap exists within events during the same calendar period. If this assumption is disregarded, double counting many occur when MB counts are done.

10

Internationalization

This chapter contains the following:

Introduction

Internationalization is the process of creating software that can be translated more easily. Changes to the code are not specific to any particular market. The Promotion Intelligence and Promotion Planning and Optimization application has been internationalized to support multiple languages.

This chapter describes configuration settings and features of the software that ensure that the base application can handle multiple languages.

Translation

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

- 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
- Demo data
- Training materials

Supported Languages

Promotion Intelligence and Promotion Planning and Optimization is available in the following languages:

- Chinese (Traditional)
- Chinese (Simplified)
- French
- German
- Italian
- Japanese
- Korean
- Portuguese (Brazilian)
- Spanish (Spain)

Promotion Intelligence and Promotion Planning and Optimization depend on both the browser settings and the regional settings to determine which language is being supported for a specific implementation.

The applications support a single language within a single installation. The applications do not support multiple currencies within a single installation.