Oracle® Retail Promotion Intelligence and Promotion Planning and Optimization

Operations Guide

Release 12.0.1

October 2006

Oracle Retail Promotion Intelligence and Promotion Planning and Optimization Operations Guide, Release 12.0.1

Contents

3 Standard Load

5 Market-Basket Analysis

6 Predictive Modeling

7 Affinity Modeling

8 Technical Reference

Preface

Oracle Retail Promotion Intelligence analyzes the results of past promotions and advertising and the affinity effects of products on one another to deliver insight into the performance of a promotional strategy.

Oracle Retail Promotion Planning and Optimization assists you in creating and improving your promotions. It allows you to leverage the information gained from Promotion Intelligence to make the best promotion decisions by using what-if analysis and predictive forecasting.

Promotion Planning and Optimization combines analysis, planning, and implementation components to give retailers the capability to achieve the highest return on their advertising, promotion, and inventory investments.

Audience

This document is intended for people responsible for product implementation.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at

`http://www.oracle.com/accessibility/`

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

Related Documents

For more information, see the following documents in the Oracle Retail Promote Release 12.0.1 documentation set:

- Oracle Retail Promotion Intelligence and Promotion Planning and Optimization 12.0.1 Release Notes
- Oracle Retail Promotion Intelligence and Promotion Planning and Optimization Installation Guide
- Oracle Retail Promotion Intelligence User Guide
- Oracle Retail Promotion Planning and Optimization User Guide
- Oracle Retail Promotion Intelligence and Promotion Planning and Optimization Configuration Guide
- Oracle Retail Promotion Intelligence and Promotion Planning and Optimization Sample Dataset Guide

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

The chapter contains the following:

- "About the Promotion Intelligence and Promotion Planning and Optimization Operations Guide" on page 1
- "What's In This Book" on page 1

About the Promotion Intelligence and Promotion Planning and Optimization Operations Guide

The *Promotion Intelligence and Promotion Planning and Optimization Operations Guide* provides details about the essential tasks involved in using the application: the staging and loading of data that is provided by the customer, historical analysis, market basket analysis, predictive modeling, and affinity modeling.

Note: this document also refers to the Promotion Intelligence and Promotion Planning and Optimization products as simply "Promote."

What's In This Book

The Promote Operations Guide addresses the following topics:

- Standard Interface
- Standard Load
- Historical Analysis
- Market Basket Analysis
- Predictive Modeling
- Affinity Modeling
- Technical Reference

2

Standard Interface

This chapter contains the following:

Introduction

An important part of getting Promote up and running in a production environment is the gathering and loading of enterprise data. Promote requires historical and weekly data to be loaded into the Promote database. The data must be provided in a standard format, as specified in the standard interface specification. The data can then be loaded according to the standard load procedure.

This chapter contains the standard interface specifications for the data that is loaded into Promote.

Promote Standard Interface Descriptions

This section details the data interface to the Promote application. The interfaces are described in alphabetical order.

Promote requires that customer data be provided in flat files containing pipe-delimited data organized so that the data can be loaded into Promote database tables that follow the formats specified here.

The following special characters are not allowed: colon, semi-colon, comma, forward slash, backward slash, any type of quote, any type of apostrophe, <, or >.

Three interfaces (Merchandise Hierarchy Levels, Location Hierarchy Levels, and Cross Product Information) that are required by Promote are only loaded once. The information contained in these three files is collected during discussions with specific clients; however, the files themselves are not provided by clients but are created and loaded as part of the initial Promote configuration. More information on these three interfaces is provided in Chapter 3, “Standard Load.”

The standard interface includes the following:

Table 2–1 Interface Specifications

Interface Specification	Required/Optional
APE Price Elasticity	Optional
APE Promotion Elasticity	Optional
Calendar	Required
Cross Products Information	Required
Images	Optional
Inventory	Required
Location Hierarchy	Required
Location Hierarchy CDA	Optional
Location Hierarchy Levels	Required
Location Hierarchy Rename	Optional
Merchandise Hierarchy	Required
Merchandise Hierarchy CDA	Optional
Merchandise Hierarchy Levels	Required
Merchandise Hierarchy Rename	Optional
Offers	Required
Promotion Allocation	Optional
Promotion Campaign	Optional
Promotion Offer	Required
Promotion Offer Attributes	Required
Promotion Offer Merchandise	Required
Promotion Offer Store	Required
Promotions	Required
Transaction Log	Required
UDE Type	Required

Table 2–1 (Cont.) Interface Specifications

Interface Specification	Required/Optional
UDE Value	Required
Vehicle	Required
Vehicle Attributes	Required

APE Price Elasticity Standard Interface Description

The APE price elasticity interface describes the APE price elasticity data generated by the Affinity Parameter Estimator (APE) component of Promote.

Data Fields

Five fields describe each record:

- DRIVER_APE_MERCH_NODE_EXT_ID - the external ID for the Driver Merchandise node.
- TARGET_APE_MERCH_NODE_EXT_ID - the external ID for the Target Merchandise node.
- LOC_LEVEL_DESC - the external ID for the external location level.
- LOC_CLIENT_LOAD_ID - the external ID for the location.
- ELASTICITY - the APE-calculated elasticity value.

An Example

The following table shows sample APE Price Elasticity data.

Table 2–2 Sample APE Price Elasticity Data

Driver	Target	Location Level	Location ID	Elasticity

Toys:
HIER3_KEY=1 80:
HIER4_KEY=2 17:
HIER5_KEY=3 17020:
	Toys:
	HIER3_KEY=1 80:
	HIER4_KEY=2 17:
	HIER5_KEY=3 17023:
		STORE	3451	0.4907

APE Promotion Elasticity Standard Interface Description

The APE price elasticity interface describes the APE promotion elasticity data generated by the Affinity Parameter Estimator (APE) component of Promote.

Data Fields

Five fields describe each record:

- DRIVER_APE_MERCH_NODE_EXT_ID - the external ID for the Driver Merchandise node.
- TARGET_APE_MERCH_NODE_EXT_ID - the external ID for the Target Merchandise node.
- LOC_LEVEL_DESC - the external ID for the external location level.
- LOC_CLIENT_LOAD_ID - the external ID for the location.

- PROMOTION_EXTERNAL_ATTR - a value generated by concatenating the source column name and its corresponding value.
- ELASTICITY - the APE-calculated elasticity value.

An Example

The following table shows sample APE Promotion Elasticity data.

Table 2–3 Sample APE Promotion Elasticity Data

Driver	Target	Location Level	Location ID	External Attribute	Elasticity
Toys: HIER3_KEY=1 80 HIER4_KEY=2 17 HIER5_KEY=3 17020	Toys: HIER3_KEY=1 80 HIER4_KEY=2 17 HIER5_KEY=3 17023	STORE	3451	VEHICLE:vehicle .circular	0.4907

Calendar Standard Interface Description

The calendar interface describes a retailer′s fiscal calendar. Each record in the file corresponds to a single fiscal week.

Data Fields

Seven fields describe each calendar record, which represents a fiscal week:

- EOP_CALENDAR_DT - the last day of the fiscal week, which is usually Saturday.
- FISCAL_YR - the number of the fiscal year for the record.
- FISCAL_QTR - the number of the fiscal quarter for the record.
- FISCAL_MO - the number of the fiscal month for the record.
- FISCAL_WK - the number of the fiscal week for the record.
- CALENDAR_WK - an alternative number for the calendar week for the record.
- SEASON - the number identifying the season associated with the calendar week.

An Example

The following table shows sample data for five weeks of a fiscal calendar.

Table 2–4 Sample Calendar Data

EOP Calendar Date	Fiscal Year	Fiscal Quarter	Fiscal Month	Fiscal Week	Calendar Week	Season
2004-02-07	2004	1	1	1	1	1
2004-02-14	2004	1	1	2	2	1
2004-02-21	2004	1	1	3	3	1
2004-02-28	2004	1	1	4	4	1
2004-03-06	2004	1	2	5	1	1

Technical Notes

The following list provides details to consider regarding the calendar data.

- The calendar must include all weeks, beginning with the earliest historical sales record and extending at least two years into the future.
- Each year included in the data must contain 52 - 53 weeks.
- The calendar file can be sent weekly or loaded all at once during the initial configuration of Promote. If provided all at once, it should contain all the historic data and extend at least three years into the future.
- Retailers can use the SEASON field to designate different seasons within the fiscal year. For example, a retailer might divide the fiscal year into two seasons.

Images Standard Interface Description

The images interface describes the data feed that is used by clients to import their image library. Promote maintains a catalog of references to the images, not the images themselves.

Data Fields

Thirteen fields describe an images record:

- NAME - The display name for the image.
- EXTERNAL_NAME - The ID for the image that is meaningful to the client. It is unique across all images.
- DESCRIPTION - An optional description of the image.
- FILE_NAME - The filename for the image.
- KEYWORDS - Keywords placeholder.
- FILE_SIZE - The size of the image file.
- WIDTH - The image width.
- HEIGHT - The image height.
- RESOLUTION - The on-screen resolution of the image.
- DEPTH - The depth of the image.
- FILE_TYPE_ENUM - The image file type. Must be JPEG (0).
- MERCH_CLIENT_LOAD_ID - The client-specific category ID.
- LEVEL_DESC - The client-specific merchandise hierarchy level description.

An Example

The following is an example of the data for an images record.

Table 2–5 Images Example Data

Name	External Name	Description	File Name	Keywords	File Size	Width	Height	Resolution	Depth	File Type Enum	Merch Client Load ID	Level Desc
CG Barbie Convertible	barbie caligirl convertible	Barbie car	barbie cgconvertible.jpg	barbie	1024	30	40			0	T000008493	SKU

Inventory Standard Interface Description

The inventory interface describes a client’s historical inventory data. This data feed is used for loading the data used by the Affinity Parameter Estimator (APE) component. Promote requires the first five fields.

Data Fields

Twenty seven fields describe an inventory record:

- MERCHANDISE_KEY - The key from the merchandise hierarchy for the item. All items must be at the same level in the merchandise hierarchy, which for Promote is the Item level.
- LOCATION_KEY - The key from the location hierarchy for the item. All items must be at the same level in the location hierarchy, which for Promote is the Store level.
- FISCAL_YR - The fiscal year of the sales record.
- FISCAL_WK - The fiscal week of the sales record.
- END_OH_QTY - The number of units of on-hand inventory at the end of the period.
- END_OO_QTY - The number of inventory units in transit to the location at the end of the period.
- UNIT_RTL - The item’s ticketed price at the end of the period.
- UNIT_CST - The item’s unit cost at the end of the period.
- INIT_RTL - The item’s ticketed price at the start of the season.
- RECEIPT_QTY - The total store receipts (in units) from the distribution centers and from transfers.
- GRSS_SLS_QTY - The gross number of new units sold for the item at the location. This excludes returns.
- GRSS_SLS_AMT - The gross dollar amount of new sales for the item at the location during the period. This excludes returns.
- NET_SLS_QTY - The net number of units sold of the item at the location. This includes returns.
- NET_SLS_AMT - The net dollar amount of sales for the item at the location during the period. This includes returns.
- TOT_DSC_AMT - The total discount amount.
- PROMO_MKDN_DSC_AMT - The total promotional markdown discount amount.
- SELLIT_MKDN_DSC_AMT - The total sell-it discount amount.
- CLR_DSC_AMT - The total clearance discount amount.
- FREIGHT - The freight cost.
- GRSS_PROFIT_AMT - The total gross margin (profit).
- DUMMY - A dummy field.
- POS_SLS_QTY - The number of new units sold of the item at the location during the period.
- POS_SLS_AMT - The dollar amount of the new sales for the item at the location during the period.

- MD_SALES_QTY - The units sold while on markdown.
- MD_SALES_AMT - The sales dollars of the units sold while on markdown.
- POS_MD_AMT - The total difference in weekly sales dollars between the promotional sales price and the inventory price.
- PERM_MD_AMT - Includes distribution center, on hand, in transit, and store on hand.

An Example

The following is an example of the data for an inventory record. Only the first five fields, which are required, are shown.

Table 2–6 Inventory Example Data

Merchandise Key	Location Key	Fiscal Yr	Fiscal Wk	End OH Qty
T0000084953	5773	2004	9	2568

Location Hierarchy Standard Interface Description

The location hierarchy interface describes how a retailer categorizes locations. The location hierarchy begins with the highest level, such as company or chain, and typically extends to the lowest level, the store. For example, a three-level location hierarchy might consist of Company, Region, and Store. Each entry (row) in the location hierarchy standard interface describes a specific location. In the example of a location hierarchy shown in Table 2–7, " Location Hierarchy Sample Data", each record describes the region and company of a specific store.

Data Fields

The location hierarchy can have up to twelve levels. Each level in the location hierarchy, just like the merchandise hierarchy, is described by three fields:

- HIERARCHY_ID - an identifier or value for the hierarchy level that is meaningful to the end user. It does not have to be unique.
- HIERARCHY_KEY - a key used to identify the location level that is unique across the chain for that level. It is used to reference the location in other data files.
- HIERARCHY_DESC - a description for the level that describes that level in the location hierarchy.

These three fields are required for each level of the location hierarchy that is used. For example, if a retailer's location hierarchy contains three levels, then the location hierarchy file will contain nine required fields. Any unused fields in the location hierarchy file should be present in the file as NULL (that is, consecutive delimiters) when the file is sent in delimited file format.

An Example

The following table shows sample data for a three-level location hierarchy that consists of Company, Region, and Store.

Table 2–7 ***Location Hierarchy Sample Data***

Hierarchy 1 (Company)			Hierarchy 2 (Region)			Hierarchy 3 (Store)		
ID	**Key**	**Desc**	**ID**	**Key**	**Desc**	**ID**	**Key**	**Desc**
1	1	Full Line	1	FL1	Northeast	1000	1000	New York
1	1	Full Line	2	FL2	Southeast	1001	1001	Atlanta
1	1	Full Line	2	FL2	Southeast	1010	1010	Charlotte
1	1	Full Line	3	FL3	Resort	1002	1002	Puerto Rico
2	2	Outlet	1	O1	Northeast	2000	2000	Philadelphia
2	2	Outlet	2	O2	Southeast	1003	1003	Atlanta

Technical Notes

The following list provides details to consider regarding the location hierarchy data.

- The best way to create a unique Key for each level in the location hierarchy depends on the retailer's hierarchy data. Whenever possible, the hierarchy Keys should not be dependent on higher levels in the hierarchy. In this way, Promote can automatically detect and handle hierarchy moves without additional data. For more information on how Promote manages location hierarchy changes, see "Location Hierarchy Rename Standard Interface Description" on page 9.
- The location hierarchy file must contain a record for each location that is referenced in any of a given week's data files.
- The location hierarchy must be described consistently throughout the data file: each hierarchy node must have the same hierarchy ancestors for all records in the file that describes the hierarchy node. In the example shown in Table 2–7 on page 9, the two records describing the hierarchy above Region FL2 are identical. Note that this consistency requirement applies to all three of the hierarchy fields (Key, ID, and Desc). Inconsistent values for hierarchy descriptions are a common reason why some location hierarchy records fail to load.
- Each node in a hierarchy can only have one parent node.
- The lowest level in the location hierarchy should be the level at which sales data is provided.
- The historical location hierarchy should contain a record for each location that is referenced in any historical sales records, even if the location is now closed. It is recommended that retailers provide a single location hierarchy file for all the historical data, rather than one file for each historical week.

Location Hierarchy CDA Standard Interface Description

The location hierarchy cda interface provides 24 additional optional attributes.

Location Hierarchy Rename Standard Interface Description

The location hierarchy rename interface facilitates moving locations within the location hierarchy. You can rename any node in the hierarchy by supplying the old node name, the new node name, and the level in the hierarchy. You cannot do this through the Location Hierarchy Standard Interface.

Merchandise Hierarchy Standard Interface Description

The merchandise hierarchy interface describes how a retailer categorizes merchandise. The merchandise hierarchy begins with the highest level, such as company or division, and typically extends to the style-color level. For example, a five-level merchandise hierarchy might consist of Division, Department, Class, Style, and Color. Each entry (row) in the merchandise hierarchy standard interface describes the hierarchy for a specific piece of merchandise. In the example of a merchandise hierarchy shown in Table 2–8 on page 10, the merchandise is an item of a specific color, and each row in the file describes the Division, Department, Class, and Style to which the specific color belongs.

Data Fields

The merchandise hierarchy can have up to fifteen levels. Each level in the merchandise hierarchy is described by three fields:

- HIERARCHY_ID - an identifier or value for the hierarchy level that is meaningful to the end user. It does not have to be unique.
- HIERARCHY_KEY - a key used to identify the merchandise level that is unique across the chain for that level. It is used to reference the merchandise in other data files.
- HIERARCHY_DESC - a description for the level that describes that level in the merchandise hierarchy.

These three fields are required for each level of the merchandise hierarchy that is used. For example, if a retailer’s merchandise hierarchy contains five levels, then the merchandise hierarchy file will contain fifteen required fields. Any unused fields in the merchandise hierarchy file should be present in the file as NULL (that is, consecutive delimiters) when the file is sent in delimited file format.

An Example

The following table shows sample data for a five-level hierarchy that consists of Division, Department, Class, Style, and Color. (The hierarchy descriptions are not included here):

Table 2–8 Merchandise Hierarchy Sample Data

Hierarchy 1 (Division)		Hierarchy 2 (Dept.)		Hierarchy 3 (Class)		Hierarchy 4 (Style)		Hierarchy 5 (Color)	
ID	**Key**	**ID**	**Key**	**ID**	**Key**	**ID**	**Key**	**ID**	**Key**
1	1	10	10	20	1020	1234	101234	9	101234509
1	1	10	10	20	1020	1234	101234	12	101234512
6	6	60	60	20	6020	1234	601234	12	601234512

In this example, the class, style, and color levels all have ID values that are not unique across the chain. Because of this, the Key values for these three levels cannot be the same as the ID values. The unique Key values for these three levels were created by combining values from higher levels in the hierarchy. The Key for the Class level was created by appending the Class ID to the Department Key. The Key for the Style level was created by appending the Style ID to the Department Key.

Technical Notes

The following list provides details to consider regarding the merchandise hierarchy data.

- The best way to create a unique Key for each level in the merchandise hierarchy depends on the retailer's hierarchy data. Whenever possible, the hierarchy Keys should not be dependent on higher levels in the hierarchy. In this way, Promote can automatically detect and handle hierarchy moves without additional data. For more information on how Promote manages merchandise hierarchy changes, see "Merchandise Hierarchy Rename Standard Interface Description" on page 11.
- The merchandise hierarchy file must contain a record for each product that is referenced in any other of a given week's data files.
- The merchandise hierarchy must be described consistently throughout the data file: each hierarchy node must have the same hierarchy ancestors for all records in the file that describes the hierarchy node. In the example shown in Table 2–8 on page 10, the first two records describe the hierarchy above Style 101234 in an identical way. Note that this consistency requirement applies to all three of the hierarchy fields (Key, ID, and Desc). Inconsistent values for hierarchy descriptions are a common reason why some merchandise hierarchy records fail to load.
- Each node in a hierarchy can only have one parent node.
- The lowest level in the merchandise hierarchy must be the level at which sales and distribution data are provided.
- The historical data files should include a record for each product that is referenced in any historical sales records, even if the product is inactive. It is recommended that retailers provide a single merchandise hierarchy file for all the historical data, rather than one file for each historical week.

Merchandise Hierarchy CDA Standard Interface Description

The merchandise hierarchy cda interface provides 24 additional optional attributes.

Merchandise Hierarchy Rename Standard Interface Description

The merchandise hierarchy rename interface facilitates reclassifying and moving merchandise within the merchandise hierarchy. Any node in the hierarchy can be renamed by supplying the old node name, the new node name, and the level in the hierarchy. This cannot be done through the Merchandise Hierarchy Standard Interface.

Offers Standard Interface Description

The offers interface contains the master data that describes a client's specific promotion (for example, a 2 for 1 promotion).

Data Fields

Eight fields describe an offer:

- NAME - The display name for the offer.
- INACTIVE - Activity flag. A value of 0 indicates the offer is active; a value of 1 indicates the offer is inactive.
- EXTERNAL_NAME - The ID for the offer that is meaningful to the client. It is unique across all offers.

- DESCRIPTION - An optional description of the offer.
- BUSINESS_RULE_CLASS_NAME - The instance of what class to use in the validation.
- TYPE_EXTERNAL_NAME - The name of the user-defined type.
- MODEL_CODE - The bit identifier of the offer. The value must be a power of 2 and is unique across the universe of all offers (for example, 0, 1, 2, 4, 8...).
- FORMAT - The output format for the offer (for example to put $ in front of the number).

An Example

The following is an example of the data for an offers record.

Table 2–9 Offers Example Data

Name	Inactive	External Name	Descrip-tion	Business Rule Class Name	Type External Name	Model Code	Format
% Off	0	offer.percent_off	% Off	com.profitlogic.promote.bean.rule.PercentOff OfferRule	ude.percent.off	1	{0}

Promotion Allocation Standard Interface Description

The promotion allocation interface provides a way to import historical space allocation usage. This applies only to promotions managed external to the application.

Data Fields

Four fields describe a promotion allocation:

- PROMO_EXTERNAL_NAME - The ID for the promotion that is meaningful to the client.
- MERCH_CLIENT_LOAD_ID - The client-specific category ID.
- LEVEL_DESC - The client-specific merchandise hierarchy level description.
- SPACE_ALLOCATION - The allocation for the given category.

An Example

The following is an example of the data for a promotion allocation.

Table 2–10 Promotion Allocation Example Data

Promo External Name	Merch Client Load ID	Level Desc	Space Alloca-tion
1-003-1-999000002	236	DEPARTMENT	0.1

Promotion Campaign Standard Interface Description

The promotion campaign interface describes a client's promotional data. This data feed provides Promote with promotional calendar information from other systems. It is also used to import historical data into the system for ad effectiveness analysis.

Data Fields

Five fields describe a promotion campaign.

- NAME - A display name for the campaign.
- DESCRIPTION - An optional description of the campaign.
- EXTERNAL_NAME - The ID for the campaign that is meaningful to the client. It is unique across all campaigns.
- BEGIN_DATE - The start date for the campaign.
- END_DATE - The end date for the campaign.
- INACTIVE - Activity flag. A value of 0 indicates the campaign is active; a value of 1 indicates the campaign is inactive.

An Example

The following is an example of the data for a promotion campaign.

Table 2–11 Promotion Campaign Example Data

Name	Description	External Name	Begin Date	End Date	Inactive
campaign 0001	BTS Campaign	Campaign for Back-to-School	2003-10-10	2003-10-17	1

Promotion Offer Standard Interface Description

The promotion offer interface describes all the offers in a promotion.

Data Fields

Twelve fields describe a promotion offer:

- NAME - The display name for the offer.
- EXTERNAL_NAME - The ID for the offer that is meaningful to the client.
- DESCRIPTION - An optional description of the offer.
- BEGIN_DATE - The start date for the offer.
- END_DATE - The end date for the offer.
- PROMO_EXTERNAL_NAME - The ID for the promotion that is meaningful to the client.
- OFFER_EXTERNAL_NAME - The ID for the offer that is meaningful to the client. It is unique across all offers.
- UDV_EXTERNAL_NAME - The actual user-defined type value.
- VALUE_INT - The integer value of the offer (either UDV_EXTERNAL_ NAME,VALUE_INT, or VALUE_DEC should be set).

- VALUE_DEC - The decimal value for the actual offer.
- PAGE_NUM - The page of the offer.
- POS_NUM - The position of the offer.

An Example

The following is an example of the data for a promotion offer.

Table 2–12 Promotion Offer Example Data

Name	External Name	Description	Begin Date	End Date	Promo External Name	Offer External Name	UDV External Name	Value Int	Value Dec	Page Num	Pos Num
Barbie Sale	PO-9010-1	CG Barbie Offer	2003-01-31	2003-02-02	1-001-1-9010	offer. percent. off	udev. percent _off.10			1	2

Promotion Offer Attributes Standard Interface Description

The promotion offer attributes interface describes the additional attributes for each offer (for example, page position: front, middle, and back).

Data Fields

Six fields describe a promotion offer attribute.

- PROMO_EXTERNAL_NAME - The ID for the promotion that is meaningful to the client.
- PROMO_OFFER_EXTERNAL_NAME - The ID for the promotion offer that is meaningful to the client. It is unique across all promotion offers.
- VEH_ATTR_EXTERNAL_NAME - The vehicle attribute name that is meaningful to the client.
- UDV_EXTERNAL_NAME - The actual user-defined type value.
- VALUE_INT - The integer value of the offer (either UDV_EXTERNAL_ NAME,VALUE_INT, or VALUE_DEC should be set).
- VALUE_DEC -The currency value for the actual offer.

An Example

The following is an example of the data for a promotion offer attribute.

Table 2–13 Promotion Offer Attribute Example Data

Promo External Name	Promo Offer External Name	Veh Attr External Name	UDV External Name	Value Int	Value Dec
1-001-1-999000000	LR-999000000-0 T0000099958	page_ location	udev.page _loca tion.front		

Promotion Offer Merchandise Standard Interface Description

The promotion offer merchandise interface describes the SKUs associated with an offer.

Data Fields

Seven fields describe a promotion offer merchandise record.

- PROMO_EXTERNAL_NAME - The ID for the promotion that is meaningful to the client.
- PROMO_OFFER_EXTERNAL_NAME - The ID for the promotion offer that is meaningful to the client. It is unique across all promotion offers.
- MERCH_CLIENT_LOAD_ID - The client-specific category ID.
- LEVEL_DESC - The client-specific merchandise hierarchy level description.
- FULL_PRICE - The price of the item.
- PROMO_PRICE - The promotion price of the item.
- COST - The actual cost of the item.

An Example

The following is an example of the data for a promotion offer merchandise record.

Table 2–14 Promotion Offer Merchandise Example Data

Promo External Name	Promo Offer External Name	Merch Client Load ID	Level Desc	Full Price	Promo Price	Cost
1-001-1-999000000	LR-999000000-0 T0000099958	T0000099958	SKU	24.50	18.37	12.25

Promotion Offer Store Standard Interface Description

The promotion offer store interface describes the stores on a promotion.

Data Fields

Three fields describe a promotion offer stores record.

- PROMO_EXTERNAL_NAME - The ID for the promotion that is meaningful to the client.
- LOC_CLIENT_LOAD_ID - The client-specific store ID.
- LEVEL_DESC - The client-specific store hierarchy level description.

An Example

The following is an example of the data for a promotion offer store record.

Table 2–15 Promotion Offer Store Example Data

Promo External Name	Loc Client Load ID	Level Desc
1-001-1-99 9000000	6493	STORE

Promotions Standard Interface Description

The promotions interface describes a client's promotions data. The data feed provides Promote with promotional calendar information from other systems. It is also used to import historical data into the system that is used for ad effectiveness analysis.

Data Fields

Ten fields describe a promotion record.

- NAME - The display name for the promotion.
- INACTIVE - Activity flag. A value of 0 indicates the offer is active; a value of 1 indicates the offer is inactive.
- EXTERNAL_NAME - The ID for the promotion that is meaningful to the client. It is unique across all promotions.
- DESCRIPTION - An optional description of the promotion.
- BEGIN_DATE - The start date for the promotion.
- END_DATE - The end date for the promotion.
- TOTAL_COST - The total cost allocated to the promotion.
- VEHICLE_EXTERNAL_NAME - The vehicle that is used when promoting items.
- PAGES - The number of pages for the vehicle.
- CAMPAIGN_ATTRIBUTE_EXTERNAL_NAME - The name of the campaign being used when promoting items.

An Example

The following is an example of the data for a promotion record.

Table 2–16 Promotion Example Data

Name	Inactive	External Name	Descrip-tion	Begin Date	End Date	Total Cost	Vehicle External Name	Pages	Cam-paign External Name
Circular for Week 20	0	promo0001	Standard Weekly Circular	2003-10-10	2003-10-17	120000.00	vehicle.circular	4	Campaign for Back-to-School

Transaction Log Standard Interface Description

The transaction log interface describes a client's basic transactional information. This data feed is used when using Promote's built-in data warehousing feature. Alternative configurations are available when leveraging a client's existing data warehouse.

Data Fields

Eleven fields describe a transaction log record:

- TRANS_ID - The unique identifier for the transaction.
- STORE_ID - The ID for the location of the transaction.
- TXN - The transaction date.
- QUANTITY - The number of a given item that were purchased in the market basket.
- PROD_ID - The ID of the product being sold.
- UNIT_COST - The per-unit cost of the sold product.
- EXT_COST - The market basket cost of the item.
- UNIT_PRICE - The per-unit price of the sold product.
- EXT_PRICE - The total market basket cost of the item.
- EXT_DISCOUNT - The amount, for this type of item in the market basket, that the price has been reduced, if the item is on promotion.
- EXT_GM - The amount of market achieved for this item.
- DISCOUNT_CODE - Discount flag. 0 = none; 1 = on Ad (item was promoted).

An Example

The following is an example of the data for a transaction log record.

Table 2–17 Values Example Data

Trans ID	Store ID	Txn	Quantity	Prod ID	Unit cost	Ext Cost	Unit Price	Ext Price	Ext Discount	Ext GM	Discount Code
0011175830094FFGF	459901	2006-02-28	2	89078460094	6.0	12.0	8.99	17.98	6.0	5.98	1

User Defined Type Standard Interface Description

The user defined type interface describes a client-defined type (for example, percent off and page units). The data feed typically provides Promote with user-defined information from other systems.

Data Fields

Four fields describe a user-defined type:

- TYPE_NAME - The display name for the type.
- INACTIVE - Activity flag. A value of 0 indicates the offer is active; a value of 1 indicates the offer is inactive.
- EXTERNAL_NAME - The ID for the type that is meaningful to the client. It is unique across all types.
- DESCRIPTION - An optional description of the type.

An Example

The following is an example of the data for a type record.

Table 2–18 Type Example Data

Type Name	Inactive	External Name	Description
% Off	0	ude.percent_off	Percent Off

User Defined Value Standard Interface Description

The user defined value interface describes a value for a client-defined type (for example, 5% for a Percent Off user-defined type). The data feed typically provides Promote with user-defined values information from other systems.

Data Fields

Seven fields describe a user-defined value:

- VALUE_NAME - The display name for the value.
- INACTIVE - Activity flag. A value of 0 indicates the offer is active; a value of 1 indicates the offer is inactive.
- EXTERNAL_NAME - The ID for the type that is meaningful to the client. It is unique across all types.
- TYPE_EXTERNAL_NAME - The name of the user-defined type.
- DESCRIPTION - An optional description of the type.
- ORDER_ID - The position of the element in an ordered list.
- EXTERNAL_CODE - The element's ID in the external system.

An Example

The following is an example of the data for a values record.

Table 2–19 Values Example Data

Value Name	Inactive	External Name	Type External Name	Description	Order ID	External Code
10 %	0	ude.percent_off.10	ude.percent_off	10 % Off	2	1

Vehicle Standard Interface Description

The vehicle interface describes a client's promotion vehicle (for example, circular or TV ad). The data feed typically provides Promote with vehicles information from other systems. It is also used to import historical data into the system for ad effectiveness analysis.

Data Fields

Six fields describe a vehicle:

- VEHICLE_NAME - The display name for the vehicle.
- INACTIVE - Activity flag. A value of 0 indicates the offer is active; a value of 1 indicates the offer is inactive.

- EXTERNAL_NAME - The ID for the vehicle that is meaningful to the client. It is unique across all vehicles.
- DESCRIPTION - An optional description of the vehicle.
- BUSINESS_RULE_CLASS_NAME - The instance of what class to use in the validation.
- MODEL_CODE - The bit identifier of the offer. The value must be a power of 2 and unique across the universe of all offers (for example, 0, 1, 2, 4, 8...).

An Example

The following is an example of the data for an vehicles record.

Table 2–20 Vehicles Example Data

Name	Inactive	External Name	Descrip-tion	Business Rule Class Name	Model Code
Circular	0	vehicle.circular	Circular	com.profitlogic.promote.bean.rule.CircularVehicleRule	1

Vehicle Attributes Standard Interface Description

The vehicle attributes interface describes the attributes of a client's vehicle (for example, pages and space allocation). The data feed typically provides Promote with vehicle attributes information from other systems. It is also used to import historical data into the system for ad effectiveness analysis.

Data Fields

Eleven fields describe a vehicle attribute:

- VEHICLE_ATTR_NAME - The display name for the vehicle attribute.
- INACTIVE - Activity flag. A value of 0 indicates the offer is active; a value of 1 indicates the offer is inactive.
- EXTERNAL_NAME - The ID for the vehicle attribute that is meaningful to the client. It is unique across all vehicle attributes.
- DESCRIPTION - An optional description of the vehicle attribute.
- ATTRIBUTE_LEVEL - The level at which to show the attribute. A value of 1 indicates vehicle; a value of 1 indicates item.
- VEHICLE_EXTERNAL_NAME - The ID for the parent vehicle that is meaningful to the client. It is unique across all vehicles.
- TYPE_EXTERNAL_NAME - The name of the user-defined type.
- MODEL - Indicates if the attribute is to be sent to the analysis engine. A value of 0 indicates do not send; a value of 1 indicates send.

- VISIBLE - Visibility flag. A value of 0 indicates invisible; a value of 1 indicates visible.
- ORDER_ID - Not used.
- FORMAT - The output format for the vehicle attribute (for example, to put Page label in front of the number).

An Example

The following is an example of the data for a vehicle attribute record.

Table 2–21 Vehicle Attributes Example Data

Vehicle Attribute Name	Inactive	External Name	Description	Attribute Level	Vehicle External Name	Type External Name	Model	Visible	Order ID	Format
Page Location	0	page_ location	Page Location	1	vehicle. circular	ude. page_ location	1	0		{0}

Promote Interface Specifications

The following tables provide ordered lists of the contents of each of the Promote interface specifications. The specifications are organized into alphabetical order.

APE Price Elasticity Specification (BEE_APE_PRICE_ELASTICITY_TBL)

Table 2–22 APE Price Elasticity Standard Interface Specification[1]

Attribute	Attribute Description	Data Type	Maximum Length	Nullable Y/N
DRIVER_APE_MERCH_NODE_EXT_ID	The external ID for the Driver Merchandise node.	String	200	Y
TARGET_APE_MERCH_NODE_EXT_ID	The external ID for the Target Merchandise node.	String	200	Y
LOC_LEVEL_DESC	The external ID for the external location level.	String	50	Y
LOC_CLIENT_LOAD_ID	The external ID for the location.	String	50	Y
ELASTICITY	The APE-calculated elasticity value.	Decimal	15,4	Y

[1] For Decimal, the requirement is a number of a certain defined length and with a certain number of decimal places. For example, (22,2) is a number that can be up to 22 digits long and that can have two digits after the decimal point.

APE Promotion Elasticity Specification (BEE_APE_PROMO_ELASTICITY_TBL)

Table 2–23 APE Promotion Elasticity Standard Interface Specification[1]

Attribute	Attribute Description	Data Type	Maximum Length	Nullable Y/N
DRIVER_APE_MERCH_NODE_EXT_ID	The external ID for the Driver Merchandise node.	String	200	Y
TARGET_APE_MERCH_NODE_EXT_ID	The external ID for the Target Merchandise node.	String	200	Y
LOC_LEVEL_DESC	The external ID for the external location level.	String	50	Y
LOC_CLIENT_LOAD_ID	The external ID for the location.	String	50	Y
PROMOTION_EXTERNAL_ATTR	A value generated by concatenating the source column name and its corresponding value.	String	200	Y
ELASTICITY	The APE-calculated elasticity value.	Decimal	15,4	Y

[1] For Decimal, the requirement is a number of a certain defined length and with a certain number of decimal places. For example, (22,2) is a number that can be up to 22 digits long and that can have two digits after the decimal point.

Calendar Specification (ASH_CAL_TBL)

Table 2–24 Calendar Standard Interface Specification

Attribute	Attribute Description	Data Type	Maximum Length	Nullable Y/N
EOP_CALENDAR_DT	Ending calendar date of the fiscal week (which is usually a Saturday).	Date in format YYYY-MM-DD	10	N
FISCAL_YR	Number of the fiscal year.	Integer	4	N
FISCAL_QTR	Number of fiscal quarter.	Integer	1	N
FISCAL_MO	Number of the fiscal month.	Integer	2	N
FISCAL_WK	Number of the fiscal week.	Integer	2	N
CALENDAR_WK	An alternative number for the calendar week (optional).	Integer	2	Y
SEASON	Season number associated with the week.	Integer	2	N

Images Specification (BEE_IMAGE_TBL)

Table 2–25 Images Standard Interface Specification

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
NAME	Display name for image.	String	40	N
EXTERNAL_NAME	The ID for the image that is meaningful to the client. Unique across the images.	String	40	N
DESCRIPTION	An optional description of the image.	String	1000	Y
FILE_NAME	The filename of the image.	String	250	N
KEYWORDS	Keywords placeholder.	String	1000	Y
FILE_SIZE	The size of the image file.	Integer	10	Y
WIDTH	The image width.	Integer	10	Y
HEIGHT	The image height.	Integer	10	Y
RESOLUTION	The on-screen resolution of the image.	Integer	10	Y
DEPTH	The depth of the image.	Integer	10	Y
FILE_TYPE_ENUM	The image file type. Must be JPEG (0).	Integer	10	Y
MERCH_CLIENT_LOAD_ID	The client-specific category ID.	String	50	Y
LEVEL_DESC	The client-specific merchandise hierarchy level description.	String	50	Y

Inventory Specification (WK_HIST_SALES_INV_TBL)

Table 2–26 Inventory Standard Interface Specification [1]

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
MERCHANDISE_KEY	The key from the merchandise hierarchy for the item.	String	25	N
LOCATION_KEY	The key from the location hierarchy for the item.	String	25	N
FISCAL_YR	The fiscal year of the sales record.	Integer	4	N
FISCAL_WK	The fiscal week of the sales record.	Integer	2	N
END_OH_QTY	The number of units of on-hand inventory at the end of the period.	Integer	12	N
END_OO_QTY	The number of inventory units in transit to the location at the end of the period.	Integer	12	Y
UNIT_RTL	The item's ticketed price at the end of the period.	Decimal	7,2	Y
UNIT_CST	The item's unit cost at the end of the period.	Decimal	7,2	Y
INIT_RTL	The item's ticketed price at the start of the season.	Decimal	7,2	Y
RECEIPT_QTY	The total store receipts (in units) from the distribution centers and from transfers.	Integer	12	Y
GRSS_SLS_QTY	The gross number of new units sold for the item at the location. This excludes returns.	Integer	12	Y
GRSS_SLS_AMT	The gross dollar amount of new sales for the item at the location during the period. This excludes returns.	Decimal	16,2	Y

Table 2–26 (Cont.) Inventory Standard Interface Specification [1]

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
NET_SLS_QTY	The net number of units sold of the item at the location. This includes returns.	Integer	12	Y
NET_SLS_AMT	The net dollar amount of sales for the item at the location during the period. This includes returns.	Decimal	16,2	Y
TOT_DSC_AMT	The total discount amount.	Decimal	16,2	Y
PROMO_MKDN_DSC_AMT	The total promotional markdown discount amount.	Decimal	16,2	Y
SELLIT_MKDN_DSC_AMT	The total sell-it discount amount.	Decimal	16,2	Y
CLR_DSC_AMT	The total clearance discount amount.	Decimal	16,2	Y
FREIGHT	The freight cost.	Decimal	16,2	Y
GRSS_PROFIT_AMT	The total gross margin (profit).	Decimal	16,2	Y
DUMMY	A dummy field.			
POS_SLS_QTY	The number of new units sold of the item at the location during the period.	Integer	12	Y
POS_SLS_AMT	The dollar amount of the new sales for the item at the location during the period.	Decimal	16,2	Y
MD_SALES_QTY	The units sold while on markdown.	Integer	12	Y

Table 2–26 (Cont.) Inventory Standard Interface Specification [1]

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
MD_SALES_AMT	The sales dollars of the units sold while on markdown.	Decimal	16,2	Y
POS_MD_AMT	The total difference in weekly sales dollars between the promotional sales price and the inventory price.	Decimal	16,2	Y
PERM_MD_AMT	Includes distribution center, on hand, in transit, and store on hand.	Decimal	16,2	Y

[1] For Decimal, the requirement is a number of a certain defined length and with a certain number of decimal places. For example, (22,2) is a number that can be up to 22 digits long and that can have two digits after the decimal point.

Location Hierarchy Specification (ASH_LH_TBL)

Table 2–27 Location Hierarchy Standard Interface Specification

Attribute Name	Attribute Description	Data Type	Maximum Length	Nullable Y/N
HIERARCHY1_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY1_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY1_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY2_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY2_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY2_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY3_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY3_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY3_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY4_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY4_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY4_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY5_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY5_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY5_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY6_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY6_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY6_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY7_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY7_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY7_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY8_ID	ID for this level of the hierarchy.	String	25	Y

Table 2–27 (Cont.) Location Hierarchy Standard Interface Specification

Attribute Name	Attribute Description	Data Type	Maximum Length	Nullable Y/N
HIERARCHY8_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY8_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY9_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY9_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY9_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY10_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY10_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY10_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY11_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY11_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY11_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY12_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY12_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY12_DESC	Description of this level of the hierarchy.	String	50	Y

LH CDA Specification (ASH_LH_CDA_TBL)

Table 2–28 Location Hierarchy CDA Standard Interface Specification[1]

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
LOCATION_KEY	Unique identifier for location hierarchy.	String	25	N
LOCATION_LEVEL	Level within the location hierarchy.	String	50	N
ATTRIBUTE1		String	100	Y
ATTRIBUTE2		String	100	Y
ATTRIBUTE3		String	100	Y
ATTRIBUTE4		String	100	Y
ATTRIBUTE5		String	100	Y
ATTRIBUTE6		String	100	Y
ATTRIBUTE7		String	100	Y
ATTRIBUTE8		String	100	Y
ATTRIBUTE1_DATE		Date in format YYYY-MM-DD	10	Y
ATTRIBUTE2_DATE		Date in format YYYY-MM-DD	10	Y
ATTRIBUTE3_DATE		Date in format YYYY-MM-DD	10	Y
ATTRIBUTE4_DATE		Date in format YYYY-MM-DD	10	Y
ATTRIBUTE5_DATE		Date in format YYYY-MM-DD	10	Y
ATTRIBUTE6_DATE		Date in format YYYY-MM-DD	10	Y
ATTRIBUTE7_DATE		Date in format YYYY-MM-DD	10	Y
ATTRIBUTE8_DATE		Date in format YYYY-MM-DD	10	Y
ATTRIBUTE1_NUMBER		Decimal	31,3	Y
ATTRIBUTE2_NUMBER		Decimal	31,3	Y
ATTRIBUTE3_NUMBER		Decimal	31,3	Y
ATTRIBUTE4_NUMBER		Decimal	31,3	Y
ATTRIBUTE5_NUMBER		Decimal	31,3	Y
ATTRIBUTE6_NUMBER		Decimal	31,3	Y
ATTRIBUTE7_NUMBER		Decimal	31,3	Y
ATTRIBUTE8_NUMBER		Decimal	31,3	Y

[1] For Decimal, the requirement is a number of a certain defined length and with a certain number of decimal places. For example, (22,2) is a number that can be up to 22 digits long and that can have two digits after the decimal point.

LH Rename Specification (ASH_LHRENAME_TBL)

Table 2–29 Location Hierarchy Rename Standard Interface Specification

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
OLD_LOCATION_KEY	Old unique identifier for location hierarchy.	String	25	N
NEW_LOCATION_KEY	New unique identifier for location hierarchy.	String	25	N
LOCATION_LEVEL	Level within the location hierarchy.	String	50	N

Merchandise Hierarchy Specification (ASH_MH_TBL)

Table 2–30 Merchandise Hierarchy Standard Interface Specification

Attribute Name	Attribute Description	Data Type	Maximum Length	Nullable Y/N
HIERARCHY1_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY1_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY1_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY2_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY2_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY2_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY3_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY3_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY3_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY4_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY4_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY4_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY5_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY5_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY5_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY6_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY6_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY6_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY7_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY7_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY7_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY8_ID	ID for this level of the hierarchy.	String	25	Y

Table 2–30 (Cont.) Merchandise Hierarchy Standard Interface Specification

Attribute Name	Attribute Description	Data Type	Maximum Length	Nullable Y/N
HIERARCHY8_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY8_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY9_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY9_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY9_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY10_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY10_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY10_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY11_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY11_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY11_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY12_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY12_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY12_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY13_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY13_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY13_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY14_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY14_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY14_DESC	Description of this level of the hierarchy.	String	50	Y
HIERARCHY15_ID	ID for this level of the hierarchy.	String	25	Y
HIERARCHY15_KEY	Key for this level of the hierarchy.	String	25	Y
HIERARCHY15_DESC	Description of this level of the hierarchy.	String	50	Y

MH CDA Specification (ASH_MH_CDA_TBL)

Table 2–31 Merchandise Hierarchy CDA Standard Interface Specification[1]

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
MERCHANDISE_KEY	Unique identifier for merchandise hierarchy.	String	25	N
MERCHANDISE_LEVEL	Level within the merchandise hierarchy.	String	50	N
ATTRIBUTE1		String	100	Y
ATTRIBUTE2		String	100	Y
ATTRIBUTE3		String	100	Y
ATTRIBUTE4		String	100	Y
ATTRIBUTE5		String	100	Y
ATTRIBUTE6		String	100	Y
ATTRIBUTE7		String	100	Y
ATTRIBUTE8		String	100	Y
ATTRIBUTE1_DATE		Date in format YYYY-MM-DD	10	Y
ATTRIBUTE2_DATE		Date in format YYYY-MM-DD	10	Y
ATTRIBUTE3_DATE		Date in format YYYY-MM-DD	10	Y
ATTRIBUTE4_DATE		Date in format YYYY-MM-DD	10	Y
ATTRIBUTE5_DATE		Date in format YYYY-MM-DD	10	Y
ATTRIBUTE6_DATE		Date in format YYYY-MM-DD	10	Y
ATTRIBUTE7_DATE		Date in format YYYY-MM-DD	10	Y
ATTRIBUTE8_DATE		Date in format YYYY-MM-DD	10	Y
ATTRIBUTE1_NUMBER		Decimal	31,3	Y
ATTRIBUTE2_NUMBER		Decimal	31,3	Y
ATTRIBUTE3_NUMBER		Decimal	31,3	Y
ATTRIBUTE4_NUMBER		Decimal	31,3	Y
ATTRIBUTE5_NUMBER		Decimal	31,3	Y
ATTRIBUTE6_NUMBER		Decimal	31,3	Y
ATTRIBUTE7_NUMBER		Decimal	31,3	Y
ATTRIBUTE8_NUMBER		Decimal	31,3	Y

[1] For Decimal, the requirement is a number of a certain defined length and with a certain number of decimal places. For example, (22,2) is a number that can be up to 22 digits long and that can have two digits after the decimal point.

MH Rename Specification (ASH_MHRENAME_TBL)

Table 2–32 Merchandise Hierarchy Rename Standard Interface Specification

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
OLD_MERCHANDISE_KEY	Old unique identifier for merchandise hierarchy.	String	25	N
NEW_MERCHANDISE_KEY	New unique identifier for merchandise hierarchy.	String	25	N
MERCHANDISE_LEVEL	Level within the merchandise hierarchy.	String	50	N

Offers Specification (BEE_OFFER_TBL)

Table 2–33 Offers Standard Interface Specification

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
NAME	Display name for the offer.	String	40	N
INACTIVE	Activity flag. 0 = active. 1 = inactive.	Integer	1	N
EXTERNAL_NAME	The ID for the offer that is meaningful to the client. Unique across all offers.	String	40	N
DESCRIPTION	An optional description of the offer.	String	1000	Y
BUSINESS_RULE_CLASS_NAME	Instance of what class to use in validation.	String	250	Y
TYPE_EXTERNAL_NAME	Name of user defined type.	String	40	N
MODEL_CODE	Bit identifier for offer. Must be power of 2 (e.g., 0, 1, 2, 4, 8...).	Integer	10	N
FORMAT	Output format for offer (e.g., to put $ in front of number).	String	40	N

Promotion Allocation Specification (BEE_PROMO_ALLOC_TBL)

Table 2–34 Promotion Allocation Standard Interface Specification [1]

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
PROMO_EXTERNAL_NAME	The ID for the promotion that is meaningful to the client.	String	120	N
MERCH_CLIENT_LOAD_ID	The client-specific category ID.	String	50	N
LEVEL_DESC	The client-specific merchandise hierarchy level description.	String	50	N
SPACE_ALLOCATION	The allocation for a given category.	Decimal	15,4	Y

[1] For Decimal, the requirement is a number of a certain defined length and with a certain number of decimal places. For example, (22,2) is a number that can be up to 22 digits long and that can have two digits after the decimal point.

Promotion Campaign Specification (BEE_PROMO_CAMPAIGN_TBL)

Table 2–35 Promotion Campaign Standard Interface Specification

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
NAME	A display name for the campaign.	String	40	N
DESCRIPTION	An optional description of the campaign.	String	1000	N
EXTERNAL_NAME	The ID for the campaign that is meaningful to the client. It is unique across all campaigns.	String	120	Y
BEGIN_DATE	The start date of the campaign.	Date in format YYYY-MM-DD	10	N
END_DATE	The end date of the campaign.	Date in format YYYY-MM-DD	10	N
INACTIVE	Activity flag. 0 = active. 1 = inactive.	Boolean (0,1)	1	N

Promotion Offer Specification (BEE_PROMO_OFFER_TBL)

Table 2–36 Promotion Offer Standard Interface Specification [1]

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
NAME	The display name for the offer.	String	40	N
EXTERNAL_NAME	The ID for the offer that is meaningful to the client.	String	120	N
DESCRIPTION	An optional description of the offer.	String	1000	Y
BEGIN_DATE	The start date for the offer.	Date in format YYYY-MM-DD	10	N
END_DATE	The end date for the offer.	Date in format YYYY-MM-DD	10	N
PROMO_EXTERNAL_NAME	The ID for the promotion that is meaningful to the client.	String	120	N
OFFER_EXTERNAL_NAME	The ID for the offer that is meaningful to the client.	String	120	N
UDV_EXTERNAL_NAME	The actual user-defined type value.	String	120	Y
VALUE_INT	The integer value of the offer.	Integer	8	Y
VALUE_DEC	The currency value for the actual offer.	Decimal	15,4	Y
PAGE_NUM	The page of the offer.	Integer	8	Y
POS_NUM	The position of the offer.	Integer	4	Y

[1] For Decimal, the requirement is a number of a certain defined length and with a certain number of decimal places. For example, (22,2) is a number that can be up to 22 digits long and that can have two digits after the decimal point.

Promotion Offer Attribute Specification (BEE_PROMO_OFFER_ATTR_TBL)

Table 2–37 Promotion Offer Attribute Standard Interface Specification [1]

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
PROMO_EXTERNAL_NAME	The ID for the promotion that is meaningful to the client.	String	120	N
PROMO_OFFER_EXTERNAL_NAME	The ID for the promotion offer that is meaningful to the client.	String	120	N
VEH_ATTR_EXTERNAL_NAME	The vehicle attribute name that is meaningful to the client.	String	120	N
UDV_EXTERNAL_NAME	The actual user-defined type value.	String	120	Y
VALUE_INT	The integer value of the offer.	Integer	8	Y
VALUE_DEC	The currency value for the actual offer.	Decimal	15,4	Y

[1] For Decimal, the requirement is a number of a certain defined length and with a certain number of decimal places. For example, (22,2) is a number that can be up to 22 digits long and that can have two digits after the decimal point.

Promotion Offer Merchandise Specification (BEE_PROMO_OFFER_MERCH_TBL)

Table 2–38 Promotion Offer Merchandise Standard Interface Specification [1]

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
PROMO_EXTERNAL_NAME	The ID for the promotion that is meaningful to the client.	String	120	N
PROMO_OFFER_EXTERNAL_ NAME	The ID for the promotion offer that is meaningful to the client.	String	120	N
MERCH_CLIENT_LOAD_ID	The client-specific category ID.	String	50	N
LEVEL_DESC	The client-specific merchandise hierarchy level description.	String	50	N
FULL_PRICE	The price of the item.	Decimal	15,4	Y
PROMO_PRICE	The promotion price of the item.	Decimal	15,4	Y
COST	The actual cost of the item.	Decimal	15,4	Y

[1] For Decimal, the requirement is a number of a certain defined length and with a certain number of decimal places. For example, (22,2) is a number that can be up to 22 digits long and that can have two digits after the decimal point.

Promotion Offer Store Specification (BEE_PROMO_STORE_TBL)

Table 2–39 Promotion Offer Store Standard Interface Specification

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
PROMO_EXTERNAL_NAME	The ID for the promotion that is meaningful to the client.	String	120	N
LOC_CLIENT_LOAD_ID	The client-specific store hierarchy level description.	String	50	N
LEVEL_DESC	The client-specific hierarchy level description.	String	50	N

Promotions Specification (BEE_PROMOTIONS_TBL)

Table 2–40 Promotions Standard Interface Specification[1]

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
NAME	A display name for the promotion.	String	40	N
INACTIVE	Activity flag. 0 = active. 1 = inactive.	Integer	1	Y
EXTERNAL_NAME	The ID for the promotion that is meaningful to the client. Unique across the promotion.	String	40	N
DESCRIPTION	An optional description of the promotion.	String	1000	Y
BEGIN_DATE	Start date of the promotion.	Date in format YYYY-MM-DD	10	N
END_DATE	End date of the promotion.	Date in format YYYY-MM-DD	10	N
TOTAL_COST	The total cost allocated to the promotion.	Decimal	15,4	Y
VEHICLE_EXTERNAL_NAME	The vehicle used when promoting items.	String	120	N
PAGES	The number of pages for the vehicle.	Integer	8	Y
CAMPAIGN_EXTERNAL_NAME	The name of the campaign used for the promotion.	String	120	N

[1] For Decimal, the requirement is a number of a certain defined length and with a certain number of decimal places. For example, (22,2) is a number that can be up to 22 digits long and that can have two digits after the decimal point.

Transaction Log Specification (MB_DETAIL)

Table 2–41 Transaction Log Standard Interface Specification[1]

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
TRANS_ID	Unique identifier for transaction.	Integer	9	N
STORE_ID	ID for location where transaction occurred.	String	25	N
TXN	Transaction date.	Date in format YYYY-MM-DD	10	N
QUANTITY	How many of given items were purchased in market basket.	Integer	9	N
PROD_ID	ID of sold product.	String	25	N
UNIT_COST	Per-unit cost of sold product.	Decimal	15,4	N
EXT_COST	Market basket cost of item.	Decimal	15,4	Y
UNIT_PRICE	Per-unit price of sold product.	Decimal	15,4	N
EXT_PRICE	Extended (total market basket) cost of item.	Decimal	15,4	N
EXT_DISCOUNT	Amount, for type of item in market basket, that price has been reduced if item is on promotion.	Decimal	15,4	Y
EXT_GM	Amount of market achieved for item.	Decimal	15,4	Y
DISCOUNT_CODE	Discount flag. 0 = none. 1 = on Ad (item was promoted).	Integer	1	N

[1] For Decimal, the requirement is a number of a certain defined length and with a certain number of decimal places. For example, (22,2) is a number that can be up to 22 digits long and that can have two digits after the decimal point.

User Defined Type Specification (BEE_USER_DEFINED_TYPE_TBL)

Table 2–42 User Defined Type Standard Interface Specification

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
TYPE_NAME	A display name for the type.	String	40	N
INACTIVE	Activity flag. 0 = active. 1 = inactive.	Integer	1	N
EXTERNAL_NAME	The ID for the type that is meaningful to the client. Unique across all types.	String	40	N
DESCRIPTION	An optional description of the offer.	String	1000	Y

User Defined Value Specification (BEE_USER_DEFINED_VALUE_TBL)

Table 2–43 User Defined Value Standard Interface Specification

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
VALUE_NAME	A display name for the user-defined value.	String	40	N
INACTIVE	Activity flag. 0 = active. 1 = inactive.	Integer	1	N
EXTERNAL_NAME	The ID for the type that is meaningful to the client. Unique across all types.	String	40	N
TYPE_EXTERNAL_NAME	A string name of the user-defined type.	String	40	N
DESCRIPTION	Optional description of user-defined type.	String	1000	Y
ORDER_ID	Position of the element in an ordered list.	Integer	8	Y
EXTERNAL_CODE	The element’s ID in the external system.	Integer	8	Y

Vehicle Specification (BEE_VEHICLE_TBL)

Table 2–44 Vehicle Standard Interface Specification

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
VEHICLE_NAME	A display name for the vehicle.	String	40	N
INACTIVE	Activity flag. 0 = active. 1 = inactive.	Integer	1	N
EXTERNAL_NAME	The ID for the vehicle that is meaningful to the client. Unique across all vehicles.	String	40	N
DESCRIPTION	An optional description of the vehicle.	String	1000	Y
BUSINESS_RULE_CLASS_NAME	Instance of what class to use in validation.	String	250	Y
MODEL_CODE	Bit identifier for vehicle. Must be power of 2 (e.g., 0, 1, 2, 4, 8...).	Integer	10	N

Vehicle Attributes Specification (BEE_VEHICLE_ATTR_TBL)

Table 2–45 Vehicle Attributes Standard Interface Specification

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
VEHICLE_ATTR_NAME	A display name for the vehicle attribute.	String	40	N
INACTIVE	Activity flag. 0 = active. 1 = inactive.	Integer	1	N
EXTERNAL_NAME	The ID for the vehicle attribute that is meaningful to the client. Unique across all vehicle attributes.	String	40	N
DESCRIPTION	An optional description of the vehicle attribute.	String	1000	Y
ATTRIBUTE_LEVEL	The level at which to show the attribute. 0 = vehicle. 1 = item.	Integer	1	Y
VEHICLE_EXTERNAL_NAME	ID for the parent vehicle that is meaningful to the client. Unique across all vehicles.	String	40	N
TYPE_EXTERNAL_NAME	Name of user defined type.	String	40	N
MODEL	Flag indicating if attribute should be sent to analysis engine. 0 = not send. 1 = send.	Integer	1	N
VISIBLE	Visibility flag. 0 = invisible. 1 = visible.	Integer	1	Y
ORDER_ID	Not used.	Integer	8	Y
FORMAT	Output format for vehicle attribute (e.g., to put Page label in front of number).	String	40	N

3

Standard Load

This chapter contains the following:

Introduction

This chapter describes the process to execute the standard load procedure, which transforms and loads retail data into the target database. It also includes standard load error messages and information about one-time data loads that are not part of the standard interface.

Standard Load Process

Promote provides two scripts that 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.

Table 3–1 Flat Files Names

Example File Name	File Content
apeprice.txt	APE Price Elasticity
apepromo.txt	APE Promo Elasticity
cal.txt	Calendar
images.txt	Images
lh.txt	Location Hierarchy
lh_cda.txt	Location Hierarchy Configurable Data Attributes
lhrename.txt	LH Rename
mh.txt	Merchandise Hierarchy

Table 3–1 (Cont.) Flat Files Names

Example File Name	File Content
mh_cda.txt	Merchandise Hierarchy Configurable Data Attributes
mhrename.txt	MH Rename
offer.txt	Offer
pralloc.txt	Promotion Allocation
prcampgn.txt	Promotion Campaign
proffer.txt	Promotion Offer
proffattr.txt	Promotion Offer Attributes
proffmerch.txt	Promotion Offer Merchandise
proffstore.txt	Promotion Offer Store
promo.txt	Promotions
translog.txt	Transaction Log
udetype.txt	UDE Types
udevalue.txt	UDE Values
veh.txt	Vehicles
vehattr.txt	Vehicle Attributes

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 Promote 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 3–2 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
-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
-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 3–3 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
-h	--help	displays help and exits

Load Procedures

Here is a description of each load procedure, which includes the source table and the target table.

Load APE Price Elasticity

Procedure: com.profitlogic.db.beech.LoadApePriceElasticity

Source Table: BEE_APE_PRICE_ELASTICITY_TBL

Target Table: PR_APE_PRICE_ELASTICITY_TBL

Description: This procedure loads the price elasticity data generated by the APE component.

Load APE Promotion Elasticity

Procedure: com.profitlogic.db.beech.LoadApePromoElasticity

Source Table: BEE_APE_PROMO_ELASTICITY_TBL

Target Table: PR_APE_PROMO_ELASTICITY_TBL

Description: This procedure loads the promotion elasticity data generated by the APE component.

Load Calendars

Procedure: com.profitlogic.db.birch.LoadCalendars

Source Table: ASH_CAL_TBL

Target Table: PERIODS_TBL

Description: This procedure updates the PERIODS_TBL, which is seeded by Promote during installation. The following columns in PERIODS_TBL are updated:

- FISCAL_YR
- FISCAL_MO
- FISCAL_WK
- FISCAL_QUARTER
- FISCAL_HALF
- CALENDAR_YR
- CALENDAR_MO

- CALENDAR_WK
- CALENDAR_QUARTER
- SEASON (the rows derived from ASH_CAL_TBL)

Load Images

Procedure: com.profitlogic.db.beech.LoadImageMaster

Source Table: BEE_IMAGE_TBL

Target Table: PR_IMAGE_TBL

Description: This procedure is responsible for loading promotion offer images into the database for a sample image feed interface.

Load Inventory

Procedure: com.profitlogic.db.beech.load_weekly_history_data.load_at_all_levels

Source Table: WK_HIST_SALES_INV

Target Table: ACT_HIST_TBL_LVL_X

Description: This procedure is responsible for the inventory load.

Load Location Hierarchy

Procedure: com.profitlogic.db.birch.LoadLocationHierarchy

Source Tables:

- ASH_LHL_TBL
- ASH_LH_TBL
- ASH_LH_CDA_TBL

Target Tables:

- LOCATION_HIERARCHY_TBL
- LOCATION_ATTR_TBL

Description: This procedure loads the entire location hierarchy, with the exception of the node (CHAIN) that is seeded by Promote during installation. It updates the location hierarchy based on the most recent information in ASH_LH_TBL and the levels specified in ASH_LHL_TBL. It completely re-loads LOCATION_ATTR_TBL with the most recent data from ASH_LH_CDA_TBL.

Load Location Table

Procedure: com.profitlogic.db.birch.LoadLHTbl

Source Table: LOCATION_HIERARCHY_TBL

Target Table: LOCATION_TBL

Description: This procedure completely re-loads LOCATION_TBL from LOCATION_HIERARCHY_TBL. LOCATION_TBL is a horizontally flattened view of the location hierarchy, used to improve the performance of other load procedures and Promote Planning.

Load LTClose Table

Procedure: com.profitlogic.db.birch.LoadLTCLOSE

Source Tables:

- LOCATION_TBL
- CLIENT_HIERARCHY_LEVELS_TBL

Target Table: LTCLOSE_TBL

Description: This procedure completely re-loads LTCLOSE_TBL from LOCATION_TBL using location hierarchy levels specified in CLIENT_HIERARCHY_LEVELS_TBL. TCLOSE_TBL is a vertically flattened view of the location hierarchy, containing each location node with all its parents. This table is used to improve the performance of other load procedures and Promote Planning.

Load LH Rename

Procedure: com.profitlogic.db.birch.LoadLHKeyRename

Source Table: ASH_LHRENAME_TBL

Target Table: LOCATION_HIERARCHY_TBL

Description: This procedure is responsible for moving locations within the location hierarchy. It updates CLIENT_LOAD_ID for a location node, based on the new LOCATION_KEY and LEVEL_DESC in ASH_LHRENAME_TBL.

Load Merchandise Hierarchy

Procedure: com.profitlogic.db.birch.LoadMerchandiseHierarchy

Source Tables:

- ASH_MHL_TBL
- ASH_MH_TBL
- ASH_MH_CDA_TBL

Target Tables:

- MERCHANDISE_HIERARCHY_TBL
- MERCH_ATTR_TBL
- PRODUCT_ITEMS_TBL

Description: This procedure loads the entire merchandise hierarchy, with the exception of the node (CHAIN) that is seeded by Promote during installation. It updates the merchandise hierarchy based on the most recent information in ASH_MH_TBL and the levels specified in ASH_MHL_TBL. It completely re-loads MERCH_ATTR_TBL with the most recent data from ASH_MH_CDA_TBL. It also updates PRODUCT_ITEMS_TBL according to the most recent merchandise hierarchy data.

Load Merchandise Table

Procedure: com.profitlogic.db.birch.LoadMHTbl

Source Table: MERCHANDISE_HIERARCHY_TBL

Target Table: MERCHANDISE_TBL

Description: This procedure completely re-loads MERCHANDISE_TBL from MERCHANDISE_HIERARCHY_TBL. MERCHANDISE_TBL is a horizontally flattened view of the merchandise hierarchy, used to improve the performance of other load procedures andPromote Planning.

Load TClose Table

Procedure: com.profitlogic.db.birch.LoadTCLOSE

Source Tables:

- MERCHANDISE_TBL
- CLIENT_HIERARCHY_LEVELS_TBL

Target Table: TCLOSE_TBL

Description: This procedure completely re-loads TCLOSE_TBL from MERCHANDISE_TBL using merchandise hierarchy levels specified in CLIENT_HIERARCHY_LEVELS_TBL. TCLOSE_TBL is a vertically flattened view of the merchandise hierarchy, containing each merchandise node with all its parents. This table is used to improve the performance of other load procedures and Promote Planning.

Load MH Rename

Procedure: com.profitlogic.db.birch.LoadMHKeyRename

Source Table: ASH_MHRENAME_TBL

Target Table: MERCHANDISE_HIERARCHY_TBL

Description: This procedure is responsible for moving merchandise within the merchandise hierarchy. It updates CLIENT_LOAD_ID for a merchandise node, based on the new MERCHANDISE_KEY and LEVEL_DESC in ASH_MHRENAME_TBL.

Load Offers

Procedure: com.profitlogic.db.beech.LoadOfferMaster

Source Table: BEE_OFFER_TBL

Target Table: PR_OFFER_TBL

Description: This procedure is responsible for loading offers.

Load Promotion Allocation

Procedure: com.profitlogic.db.beech.LoadPromoVehicleAlloc

Source Table: BEE_PROMO_ALLOC_TBL

Target Table:

- PR_PROMO_VEHICLE_ALLOC_TBL
- PR_PROMO_CAT_TBL

Description: This procedure is responsible for loading category allocations.

Load Promotion Campaign

Procedure: com.profitlogic.db.beech.LoadPromoCampaign

Source Table: BEE_PROMO_CAMPAIGN_TBL

Target Table: PR_CAL_ENT_TBL

Description: This procedure is responsible for loading campaigns.

Load Promotion Offer

Procedure: com.profitlogic.db.beech.LoadPromoOffer

Source Table: BEE_PROMO_OFFER_TBL

Target Table: PR_PROMO_OFFER_TBL

Description: This procedure is responsible for loading promotion offers.

Load Promotion Offer Attributes

Procedure: com.profitlogic.db.beech.LoadPromoVehiclePagePositionOfferAttribute

Source Table: BEE_PROMO_ORRER_ATTR_TBL

Target Table: PR_PROMO_VEH_PG_POS_OFF_ATR_TBL

Description: This procedure is responsible for loading promotion offer attributes.

Load Promotion Offer Merchandise

Procedure: com.profitlogic.db.beech.LoadPromoOfferMerchandise

Source Table: BEE_OFFER_MERCH_TBL

Target Table: PR_PROMO_OFFER_ITEM_TBL

Description: This procedure is responsible for loading offer items.

Load Promotion Offer Store

Procedure: com.profitlogic.db.beech.LoadPromoVehicleLocation

Source Table: BEE_PROMO_STORE_TBL

Target Table: PR_PROMO_VEH_LOC_TBL

Description: This procedure is responsible for loading promotion offer stores.

Load Promotions

Procedure: com.profitlogic.db.beech.LoadPromoMaster

Source Table: BEE_PROMOTIONS

Target Table: PR_PROMO

Description: This procedure is responsible for loading historical promotions.

Load Transaction Log

Procedure: com.profitlogic.db.beech.Load

Source Table:

Target Table: MB_DETAIL

Description:

Load UDE Types

Procedure: com.profitlogic.db.beech.LoadTypeMaster

Source Table: BEE_USER_DEFINED_TYPE_TBL

Target Table: PR_USER_DEFINED_TYPE_TBL

Description: This procedure is responsible for loading UDTs.

Load UDE Values

Procedure: com.profitlogic.db.beech.LoadValueMaster

Source Table: BEE_USER_DEFINED_VALUE_TBL

Target Table: PR_USER_DEFINED_VALUE_TBL

Description: This procedure is responsible for loading UDVs.

Load Vehicle

Procedure: com.profitlogic.db.beech.LoadVehicleMaster

Source Table: BEE_VEHICLE_TBL

Target Table: PR_VEHICLE_TBL

Description: This procedure is responsible for loading vehicles.

Load Vehicle Attributes

Procedure: com.profitlogic.db.beech.LoadVehicleAttributeMaster

Source Table: BEE_VEHICLE_ATTR_TBL

Target Table: PR_VEHICLE_ATTR_TBL

Description: This procedure is responsible for loading vehicle attributes.

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 22).

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

Table 3–4 Failed Records Table Names

Staging Table	Failed Record Table
ASH_CAL_TBL	ASH_CAL_TBL_BAD
BEECH_OFFER_TBL	BEECH_OFFER_TBL_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 3–5 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 Promote. In order to customize the threshold values, you must create a properties file and load it into Promote.

Error Handling Properties File

You can configure the threshold values for error handling in the properties file, **dbError.properties**. The values you set in this file override the corresponding Promote default values. The default value for the threshold of records failed is 100%. The default value for the total record threshold is 0%. Threshold values are expressed as a percentage. Note that the percentage symbol should not be included. Once you have created this file (which should be stored in **com/profitlogic/db/common/resources/dbError.properties** and called as a argument from there), you need to load it into the database schema using the procedure described on page 3-11.

Here is a sample **dbError.properties** file:

```
##########################################################################################
#This properties file contains all error customizations
#
#Note:all thresholds should be satisfied in order for the load procedure to succeed
#
##########################################################################################
#LoadPromotions error customizations
#
#Total error threshold is set to 0% of all records (default is 0%):
LoadPromotions.total.threshold=0
#
#Threshold of records failed with error 1205 should not exceed 100% (default is 100%):
LoadPromotions.1205.threshold=100
#
#Threshold of records failed with error 1207 should not exceed 100% (default is 100%):
LoadPromotions.1207.threshold=100
################################################################################
```

In the **dbError.properties** file, you can set the total error threshold as well as a separate threshold for specific verifications. When configuring the error threshold for specific verifications, you use the error message number, as shown in Table 3–8, " Standard Load Error Messages" to indicate which verification you are setting the error threshold for. The sum of all the individual thresholds cannot exceed the total threshold.

Loading the dbError.properties File

Once you have created the **dbError.properties** file, you can load it, as follows:

dbpropertiesinstaller.sh <config_root> conf/com/profitlogic/db/common/resources/dbError.properties, where config_root is the root directory of the Promote configuration files.

The format for the file <db_connections_properties> is as follows:

For Oracle:

```
db.type=oracle
db.driver=oracle.jdbc.OracleDriver
db.url=jdbc:oracle:thin:@<db_host>:<db_port>:<db_SID>
db.password=<db_password>
```

where

<db_username> is	the username for the database connection
<db_password> is	the password for the database connection
<db_host> is	the host name of the database server
<db_port> is	the port number of the database server
<db_SID> is	the SID or SERVICE_NAME value for the database from the tnsnames.ora file.

Custom Errors

As part of the **dbError.properties** file, you can create custom verifications. Custom error codes have a reserved range of 50001 to 50100. You need to provide the text of the error message and a query that defines the verification. The pre-load verification (error messages 50000 and 50001 in the following sample) is run during the pre-load verification step. The post-load verification (error message 50002 in the following sample) is run during the post-load verification step. (For a list of the steps in the load procedure, see See “Standard Load Steps” on page 23.

Once you have modified the **dbError.properties** file to include custom verifications, you must load it into the database schema using the above command.

Here is a sample:

```
######################################################################################
#Define custom PRE_LOAD verification errors with code 50000 and 50001
#(list of error codes separated by white spaces)
LoadPromotions.pre-load.custom-errors=50000 50001

#Error message:
LoadPromotions.pre-load.50000=Table ASH_CP_TBL is missing OPTIMIZATION levels
#Threshold (default is 100%):
#Note: the threshold affects only INSERT statements! If the statement is defined as a
#      SELECT, then the error will be triggered only if the query returns at least one row.
#      For any other type of statement amount of rows affected is not checked.
LoadPromotions.pre-load.50000.threshold=0
#INSERT statement should populate the “bad records” table with failed rows
#Note: in cases when the threshold is less than 100%, the INSERT statement should end
#       with a non-empty WHERE clause because the statement will be appended by an
#       additional condition.
LoadPromotions.pre-load.50000.query=                                  \
    SELECT 1 FROM %{YA_DUAL}%                                         \
     WHERE not exists (SELECT 1 FROM ash_cp_tbl                       \
                       WHERE intersect name = ‘OPTIMIZATION’)
```

```
#Error message:
LoadPromotions.pre-load.5000=No promotion is allowed after 01/01/2050
#Threshold (default is 100%):
#Note: the threshold affects only INSERT statements!
#      If the statement is defined as a SELECT, then the error will be
#      triggered only if the query returns at least one row.
#      For any other type of statement the number of rows is not checked.
LoadPromotions.pre-load.50001.threshold=0
#INSERT statement should populate the “bad records” table with failed rows
#Note: in cases when the threshold is less than 100%, the INSERT statement should end
#       with a non-empty WHERE clause because the statement will be appended by an
#      additional condition.
LoadPromotions.pre-load.50001.query=                                                      \
   INSERT INTO ash_promo_tbl_bad                                                          \
         (ERROR_ROWID, ERROR_CODE, ERROR DESC, ERROR_TIMESTAMP, merchandise_key,          \
          merchandise_level, location_key, location_level, promotion_key,                 \
         promo_start_date, promo_end_date, promo_price, promo_perc_off,                   \
          promo_desc, promo_type, prono_excl_fg, promo_number, attribute1,                \
          attribute2, attribute3, attribute4, attribute5)                                 \
   SELECT ROWID, 50001, ‘Promo after 01/01/2050’, %{YA_SYSDATE_AS_TIMESTAMP}%,            \
          merchandise_key,merchandise_level, location_key, location_level, promotion_key,\
         promo_start_date, promo_end_date, promo_price, promo_perc_off,                   \
          promo_desc, promo_type, prono_excl_fg, promo_number, attribute1,                \
          attribute2, attribute3, attribute4, attribute5)                                 \
     FROM ash_promo_tbl                                                                   \
    WHERE promo_end_date >= %{YA_TODATE/’2050-01-01’/’YYYY-MM-DD’}%

#################################################################################
# Define a custom POST_LOAD verification error with code 50002
# (list of error codes separated by spaces)
LoadPromotions.post-load.custom-errors=50002
LoadPromotions.post-load.50002=No promotion is allowed after 01/01/2050
#Note: If the statement is defined as a SELECT, then the error will be
#      triggered only if the query returns at least one row.
#      For any other type of statement the number of rows affected is not checked.
LoadPromotions.post-load.50002.query=                                               \
    SELECT 1 FROM %{YA_DUAL}%                                                       \
    WHERE exists (SELECT 1 FROM planned_promos_tbl                                  \
                   WHERE end_dt >= %{YA_TO_DATE/’2050-01-01’/’YYYY-MM-DD’}%)
```

Error Handling Report

The standard load validates the data prior to loading the data into the target tables.

A customizable view, pl_load_status_vw, provides a report on the status of data validations. This view has the following default attributes:

Table 3–6 pl_load_status_vw Default Attributes

Attribute	Description
LOAD_PROCEDURE	The specific load procedure used
SOURCE	The staging table
DATA_VALIDATION_STATUS	Success - The number of failed records is less than the threshold set or Failure - The number or failed records exceeds the threshold set
NUM_BAD_RECORDS	The number of failed records in the failed record table

Here is a sample validation report:

Table 3–7 Sample Standard Load Data Validation Report

LOAD_PROCEDURE	SOURCE	DATA_ VALIDATION_ STATUS	NUM_ BAD_ RECORDS
LoadCHLevels	ASH_MHL_TBL	Success	0
LoadCHLevels	ASH_LHL_TBL	Success	0
LoadLocationHierarchyTbl	ASH_LH_TBL	Success	0
LoadOffer	BEE_OFFER_TBL	Success	0
LoadCalendars	ASH_CAL_TBL	Success	0
LoadVehicle	BEE_VEHICLE_TBL	Success	0
LoadLocationHierarchy	ASH_LH_CDA_TBL	Success	0
LoadMerchandiseHierarchy	ASH_MH_CDA_TBL	Success	0
No transformation of data	ASH_CP_TBL	Success	0
LoadLHKeyRename	ASH_LHRENAME_TBL	Success	0
LoadMHKeyRename	ASH_MHRENAME_TBL	Success	0
LoadVehicleAttr	BEE_VEHICLE_ATTR_TBL	Success	0
LoadUDType	BEE_USER_DEFINED_TYPE_TBL	Success	0
LoadUDValue	BEE_USER_DEFINED_VALUE_TBL	Failure	50
Load Promotions	BEE_PROMOTIONS_TBL	Success	0

To generate an output file that can be emailed to interested users or integrated into production scripts, use the following script. The script writes to the standard output, which can be redirected to a file. Note that the optional WHERE clause, including the WHERE keyword itself, should be enclosed in quotes.

bash pl_load_status.sh *-r <configroot> -w <whereclause>*

where

-r DIR	--configroot=DIR	The configuration root directory
-w WHERE	--whereclause=WHERE	An optional clause used to filter specific information in the report
-h	--help	Displays help and exits

Standard Load Error Messages

The following are the error messages that may be generated during the standard load procedure.

Table 3–8 Standard Load Error Messages

Number	Error Message
System Errors	
0	The program has completed successfully.
10	An unspecified error has occurred.
20	An SQL exception has occurred.
30	A Java exception has occurred.
40	The exception limit has been exceeded.
50	The specified error threshold has been exceeded in this load procedure.
Common Errors	
100	At least one node in the hierarchy has more than one parent.
101	The number of levels in the levels table does not match the data from the source table.
102	The CHAIN level does not exist in the target table.
104	The levels table is empty.
105	The sequence for the CHAIN level should be defined as 1 in the levels table.
Load CH Levels Errors	
200	The cross-products information table (ASH_CP_TBL) does not have all the required records.
201	In the cross-products information table (ASH_CP_TBL), at least one INTERSECT_NAME has a value of NULL. An INTERSECT_NAME cannot have a value of NULL.
202	A duplicate INTERSECT_NAME has been found in the cross-products information table (ASH_CP_TBL).

Table 3–8 (Cont.) Standard Load Error Messages

Number	Error Message
203	Invalid INTERSECT_NAME has been found in the cross-products information table (ASH_CP_TBL) or not all necessary values (OPTIMIZATION, WORKSHEET, SALES, or CLUSTER) have been supplied.
204	The cross-products information table (ASH_CP_TBL) is empty.
205	In the cross-products information table (ASH_CP_TBL), at least one merchandise level has a value of NULL. A merchandise level cannot have a value of NULL.
206	In the cross-products information table (ASH_CP_TBL), at least one location level has a value of NULL. A location level cannot have a value of NULL.
Load Calendars Errors	
1000	In the calendar table (ASH_CAL_TBL), at least one fiscal year does not have between 52 and 53 weeks.
1001	In the calendar table (ASH_CAL_TBL), at least one fiscal year does not include twelve fiscal months.
1002	In the calendar table (ASH_CAL_TBL), at least one fiscal week has an End of Period (EOP) that is not Saturday.
1003	In the calendar table (ASH_CAL_TBL), at least one fiscal month is not in the range 1 - 12.
1004	In the calendar table (ASH_CAL_TBL), at least one fiscal week is not in the range 1 -53.
1005	In the calendar table (ASH_CAL_TBL), at least one fiscal year has a value of NULL. A fiscal year cannot have a value of NULL.
1006	In the calendar table (ASH_CAL_TBL), at least one fiscal month has a value of NULL. A fiscal month cannot have a value of NULL.
1007	In the calendar table (ASH_CAL_TBL), at least one fiscal week has a value of NULL. A fiscal week cannot have a value of NULL.
1008	In the calendar table (ASH_CAL_TBL), at least one fiscal season has a value of NULL. A fiscal season cannot have a value of NULL.
1009	In the calendar table (ASH_CAL_TBL), at least one End of Period (EOP) has a value of NULL. A End of Period (EOP) cannot have a value of NULL.
1010	In the calendar table (ASH_CAL_TBL), at least one fiscal quarter has a value of NULL. A fiscal quarter cannot have a value of NULL.
Load Location Hierarchy Errors	
1500	In the location hierarchy CDA staging table (ASH_LH_CDA_TBL), at least one location key has a value of NULL. A location key cannot have a value of NULL.
1501	In the location hierarchy CDA staging table (ASH_LH_CDA_TBL), at least one location level has a value of NULL. A location level cannot have a value of NULL.
1502	In the location hierarchy levels table (ASH_LHL_TBL), at least one location level has a value of NULL. A location level cannot have a value of NULL.
1503	In the location hierarchy levels table (ASH_LHL_TBL), at least one level sequence level has a value of NULL. A level sequence cannot have a value of NULL.

Table 3–8 (Cont.) Standard Load Error Messages

Number	Error Message
1504	In the location hierarchy levels table (ASH_LHL_TBL) the entries in LEVEL_SQC are not sequential.
1505	The location hierarchy levels table (ASH_LHL_TBL) should have sequence starting with 1.
1506	In the location hierarchy levels table (ASH_LHL_TBL), CHAIN is not assigned a sequence value (LEVEL_SQC) of 1.
1507	In the merchandise hierarchy table (ASH_MH_TBL), null values were detected in the hierarchy stage key columns.
Load Location Hierarchy Key Rename Errors	
1600	In the location hierarchy rename table (ASH_LHRENAME_TBL), at least one old location key has a value of NULL. A location key cannot have a value of NULL.
1601	In the location hierarchy rename table (ASH_LHRENAME_TBL), at least one new location key has a value of NULL. A location key cannot have a value of NULL.
1602	In the location hierarchy rename table (ASH_LHRENAME_TBL), at least one location level has a value of NULL. A location level cannot have a value of NULL.
1603	The old location key in the location hierarchy rename table (ASH_LHRENAME_TBL) contains duplicate values.
1604	The new location key in the location hierarchy rename table (ASH_LHRENAME_TBL) contains duplicate values.
1605	The new location key in the location hierarchy rename table (ASH_LHRENAME_TBL) is already present in the location hierarchy.
Load Merchandise Hierarchy Errors	
2001	NOT NULL has already been set for the merchandise hierarchy table (ASH_MH_TBL) stage key columns.
2002	In the merchandise hierarchy table (ASH_MH_TBL), an error dropping the unique index occurred.
2501	In the merchandise hierarchy table (ASH_MH_TBL), null values were detected in the hierarchy stage key columns.
2502	The merchandise hierarchy levels table (ASH_MHL_TBL) is empty.
2503	In the merchandise hierarchy levels table (ASH_MHL_TBL) the entries in LEVEL_SQC are not sequential.
2504	The merchandise hierarchy levels table (ASH_MHL_TBL) should contain a sequence starting with 1.
2505	In the merchandise hierarchy levels table (ASH_MHL_TBL), CHAIN is not assigned a sequence value (LEVEL_SQC) of 1.
2506	The merchandise hierarchy staging table contains duplicate values at the lowest key level.
2507	The merchandise hierarchy table (ASH_MH_TBL) contains a child node with more than one parent node.
2508	The merchandise hierarchy cda staging table (ASH_MH_CDA_TBL) contains at least one combination of MERCHANDISE_KEY and MERCHANDISE_LEVEL that is not unique.

Table 3–8 (Cont.) Standard Load Error Messages

Number	Error Message
2509	In the merchandise hierarchy CDA staging table (ASH_MH_CDA_TBL), at least one merchandise key has a value of NULL. A merchandise key cannot have a value of NULL.
2510	The merchandise hierarchy rename table (ASH_MHRENAME_TBL) contains duplicate values for OLD_MERCHANDISE_KEY.
2511	In the merchandise hierarchy levels table (ASH_MHL_TBL), at least one merchandise level has a value of NULL. A merchandise level cannot have a value of NULL.
2512	In the merchandise hierarchy levels table (ASH_MHL_TBL), at least one level sequence level has a value of NULL. a level sequence cannot have a value of NULL.
Load MH Key Rename Errors	
2600	In the merchandise hierarchy rename table (ASH_MHRENAME_TBL), at least one old merchandise key has a value of NULL. A merchandise key cannot have a value of NULL.
2601	In the merchandise hierarchy rename table (ASH_MHRENAME_TBL), at least one new merchandise key has a value of NULL. A merchandise key cannot have a value of NULL.
2602	In the merchandise hierarchy rename table (ASH_MHRENAME_TBL), at least one merchandise level has a value of NULL. A merchandise level cannot have a value of NULL.
2603	The old merchandise key in the merchandise hierarchy rename table (ASH_MHRENAME_TBL) contains duplicate values.
2604	The new merchandise key in the merchandise hierarchy rename table (ASH_MHRENAME_TBL) contains duplicate values.
2605	The new merchandise key in the merchandise hierarchy rename table (ASH_MHRENAME_TBL) is already present in the merchandise hierarchy.
Load MHTbl Errors	
6101	The MERCHANDISE_HIERARCHY_TBL table has no CHAIN record (where PARENT_MERCHANDISE_ID is NULL).
6102	The MERCHANDISE_HIERARCHY_TBL table has more than one record with PARENT_MERCHANDISE_ID = NULL (multiple CHAIN records).
Promote LoadTAE Errors	
8001	TAE output has invalid Merchandise CLIENT_LOAD_ID. It should match what is defined in MERCHANDISE_HIERARCHY_TBL.
8002	TAE output has invalid Location CLIENT_LOAD_ID. It should match what is defined in LOCATION_HIERARCHY_TBL.
8003	TAE output has an invalid Promotion. It should match what is defined in PR_PROMOTIONS
Promote LoadPromoMaster Errors	
8401	The Date defined for the promotion is NOT defined in the Fiscal Calendar.
Promote LoadTypeMaster Errors	
8501	The NAME of Attribute Type should NOT be NULL.

Table 3–8 (Cont.) Standard Load Error Messages

Number	Error Message
8502	The INACTIVE status of Attribute Type should NOT be NULL. It should be 0 or 1.
8503	The EXTERNAL NAME of Attribute Type should NOT be NULL.
8504	The EXTERNAL NAME of Attribute Type should be unique within a single load file.
Promote LoadValueMaster Errors	
8601	The NAME of Attribute Value should NOT be NULL.
8602	The INACTIVE status of Attribute Value should NOT be NULL. It should be 0 or 1.
8603	The EXTERNAL NAME of Attribute Value should NOT be NULL.
8604	The EXTERNAL NAME of Attribute Value should be unique within a single load file.
8605	The EXTERNAL CODE of Attribute Value should NOT be NULL.
8606	The Attribute Value TYPE is invalid.
Promote LoadOfferMaster Errors	
8701	The NAME of the Offer should NOT be NULL.
8702	The INACTIVE status of Offer should NOT be NULL. It should be 0 or 1.
8703	The EXTERNAL NAME of Offer should NOT be NULL.
8704	The Type for Offer, if specified, should be a valid User Defined Type.
8705	The MODEL_TYPE of Offer should NOT be NULL.
Promote LoadVehicleMaster Errors	
8801	The NAME of Vehicle should NOT be NULL.
8802	The INACTIVE status of Vehicle should NOT be NULL. It should be 0 or 1.
8803	The EXTERNAL NAME of Vehicle should NOT be NULL.
8804	The MODEL_TYPE of Vehicle should NOT be NULL.
Promote LoadVehicleAttributeMaster Errors	
8901	The NAME of Vehicle Attribute should NOT be NULL.
8902	The INACTIVE status of Vehicle Atribute should NOT be NULL. It should be 0 or 1.
8903	The EXTERNAL NAME of Vehicle Attribute should NOT be NULL.
8904	The Type for Vehicle Attribute, if specified, should be a valid User Defined Type.
8905	The Vehicle used in the Vehicle Attribute is invalid.
8906	The Attribute level is invalid. It should be 0 - promotions or 1 - item.
8907	The value for MODEL is invalid. It should be 1 - True or 0 - False.
8908	The value for INVISIBLE is invalid. It should be 1 - True or 0 - False.
Promote LoadPromoteVehicle Errors	
9001	The NAME of Promotion Vehicle should NOT be NULL.

Table 3–8 (Cont.) Standard Load Error Messages

Number	Error Message
9002	The INACTIVE status of Promotion Vehicle Atribute should NOT be NULL. It should be 0 or 1.
9003	The BEGIN DATE of Promotion Vehicle should NOT be NULL.
9004	The END DATE of Promotion Vehicle should NOT be NULL.
9005	The PROMOTION specified in Promotion Vehicle is invalid.
9006	VEHICLE specified in Promotion Vehicle is invalid
Promote LoadPromoteVehicleAttr Errors	
9101	The INACTIVE status of Promotion Vehicle Atribute should NOT be NULL. It should be 0 or 1.
9102	The PROMOTION specified in Promotion Vehicle Attribute is invalid.
9103	The VEHICLE specified in Promotion Vehicle Attribute is invalid.
9104	The VEHICLE ATTRIBUTE specified in Promotion Vehicle is invalid.
9105	Either the UDV_EXTERNAL_NAME or the VALUE_INT should be specified.
9106	The UDV_EXTERNAL_NAME used is not a valid User Defined Value.
9107	The Vehicle used is not defined for this Promotion.
9108	The Vehicle Attribute used is not defined for this Vehicle.
9109	Type of UDV and Vehicle Attribute do not match.
Promote LoadPromoteItem Errors	
9201	The INACTIVE status of the Promotion Item should NOT be NULL. It should be 0 or 1.
9202	The BEGIN DATE of the Promotion Vehicle should NOT be NULL.
9203	The END DATE of the Promotion Vehicle should NOT be NULL.
9204	The PROMOTION specified in Promotion Item is invalid.
9205	The OFFER specified in Promotion Item is invalid.
9206	The MERCH_CLIENT_LOAD_ID specified in Promotion Item is invalid.
9207	Either the UDV_EXTERNAL_NAME or VALUE_INT should be specified
9208	The UDV_EXTERNAL_NAME used is not a valid User Defined Value.
9209	Type of UDV and Offer do not match.
Promote LoadPromoteItemVehicle Errors	
9301	The INACTIVE status of Promotion Item Vehicle should NOT be NULL. It should be 0 or 1.
9302	The PROMOTION specified in Promotion Item Vehicle is invalid.
9303	The MERCH_CLIENT_LOAD_ID specified in Promotion Item Vehicle is invalid.
9304	The Promotion Item specified in Promotion Item Vehicle is invalid.
9305	The Vehicle specified in Promotion Item Vehicle is invalid.

Table 3–8 (Cont.) Standard Load Error Messages

Number	Error Message
9306	The Promotion Vehicle specified in Promotion Item Vehicle is invalid.
Promote LoadPromoteVehicleAttr Errors	
9401	The INACTIVE status of Promotion Item Vehicle Atribute should NOT be NULL. It should be 0 or 1.
9402	The PROMOTION specified in Promotion Item Vehicle Attribute is invalid.
9403	The VEHICLE specified in Promotion Item Vehicle Attribute is invalid.
9404	The VEHICLE ATTRIBUTE specified in Promotion Item Vehicle is invalid.
9405	Either the UDV_EXTERNAL_NAME or VALUE_INT should be specified.
9406	The UDV_EXTERNAL_NAME used is not a valid User Defined Value.
9407	The Vehicle used is not defined for this Promotion.
9408	The Vehicle Attribute used is not defined for this Vehicle.
9409	Type of UDV and Vehicle Attribute do not match.
Promote Interface Load Parse Errors	
9501	Invalid file format for Promote Interface.
9502	Invalid file format for Promote Item Interface.
Promote Vehicle Allocation Errors	
9601	The Promotion specified in Promo Vehicle Allocation File cannot be NULL.
9602	The Promotion specified in Promo Vehicle Allocation File is invalid.
9603	TheVehicle specified in Promo Vehicle Allocation File is invalid.
9604	The Promotion Vehicle specified in Promo Vehicle Allocation File is invalid.
9605	The Merchandise specified in Promo Vehicle Allocation File is invalid.
9606	The Merchandise specified in Promo Vehicle Allocation File cannot be NULL.
Promotions Errors	
12001	The NAME of Promotions should NOT be NULL.
12002	The EXTERNAL_NAME of Promotions should NOT be NULL.
12003	The BEGIN_DATE of Promotions should NOT be NULL.
12004	The END_DATE of Promotions should NOT be NULL.
Promo Store Errors	
12101	The PROMO_EXTERNAL_NAME should NOT be NULL.
12102	The PROMO_VEHICLE_EXTERNAL_NAME should NOT be NULL.
12103	The LOC_CLIENT_LOAD_ID should NOT be NULL.

Table 3–8 (Cont.) Standard Load Error Messages

Number	Error Message
12104	The PROMO_EXTERNAL_NAME is not valid.
12105	The LOC_CLIENT_LOAD_ID is not valid.
Promo Offer Errors	
12201	The NAME should NOT be NULL.
12202	The EXTERNAL_NAME should NOT be NULL.
12203	The BEGIN_DATE should NOT be NULL.
12204	The END_DATE should NOT be NULL.
12205	The PROMO_EXTERNAL_NAME should NOT be NULL.
12206	The PROMO_EXTERNAL_NAME is not valid.
12207	The OFFER_EXTERNAL_NAME should NOT be NULL.
12208	The OFFER_EXTERNAL_NAME is not valid.
12209	The UDVEXTERNAL_NAME is not valid.
Promo Offer Merchandise Errors	
12301	The PROMO_EXTERNAL_NAME should NOT be NULL.
12302	The OFFER_EXTERNAL_NAME should NOT be NULL.
12303	The MERCH_CLIENT_LOAD_ID and LEVEL_DESC should not be null.
12304	The MERCH_CLIENT_LOAD_ID and LEVEL_DESC combination is not valid.
Promo Campaign Errors	
12401	The NAME should not be NULL.
12402	The EXTERNAL_NAME should not be NULL.
12403	The BEGIN_DATE should NOT be NULL.
12404	The END_DATE should NOT be NULL.
Promo Offer Attribute Errors	
12501	The PROMO_EXTERNAL_NAME should NOT be NULL.
12502	The PROMO_OFFER_EXTERNAL_NAME should not be NULL.
12503	The VEH_ATTR_EXTERNAL_NAME should not be NULL.
12504	The UDV_EXTERNAL_NAME should not be NULL.
12505	The PROMO_EXTERNAL_NAME is not valid.
12506	The PROMO_OFFER_EXTERNAL_NAME is not valid.
12507	The VEH_ATTR_EXTERNAL_NAME is not valid.
12508	The UDV_EXTERNAL_NAME is not valid.

Standard Load Procedures Order

The standard load should execute in the following order:

1. The staging of:
   - ASH_CAL_TBL
   - ASH_CP_TBL
   - ASH_ITEMS_TBL
   - ASH_LHL_TBL
   - ASH_MHL_TBL
   - ASH_LH_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
   - BEE_PROMO_ALLOC
   - BEE_PROMO_CAMPAIGN
   - BEE_PROMO_OFFER_ATTR
   - BEE_PROMO_OFFER_MERCH
   - BEE_PROMO_OFFER
   - BEE_PROMO_STORE
   - BEE_PROMOTIONS
2. These load procedures:
   - com.profitlogic.db.birch.LoadCalendars
   - com.profitlogic.db.birch.LoadCHLevels
   - 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.beech.LoadPromoCampaign
- com.profitlogic.db.beech.LoadPromoMaster
- com.profitlogic.db.beech.LoadPromoVehicle
- com.profitlogic.db.beech.LoadPromoVehiclePage
- com.profitlogic.db.beech.LoadPromoVehicleAlloc
- com.profitlogic.db.beech.LoadPromoVehicleLocation
- com.profitlogic.db.beech.LoadPromoOffer
- com.profitlogic.db.beech.LoadPromoOfferMerchandise
- com.profitlogic.db.beech.LoadPromoVehiclePagePosOff
- com.profitlogic.db.beech.LoadPromoVehiclePagePositionOfferAttribute
- com.profitlogic.db.beech.MigrateMerchAttrs

3. Staging WK_HIST_SALES_INV
4. Loading load_weekly_history_data.load_at_all_levels
5. Staging MB_DETAIL

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.

Standard Interface Specifications for One-Time Data

The following three standard interface specifications are used for data that is loaded once at the beginning of a Promote deployment.

Cross Products Information Standard Interface (ASH_CP_TBL)

Items are globally defined to be at a specific level of the merchandise hierarchy and the location hierarchy through the cross products interface.

Technical Notes

The following list provides details to considering regarding the cross products information data.

- The INTERSECT_NAME is the name of the Key, which defines the purpose or feature for the data, and is either OPTIMIZATION, SALES, WORKSHEET, CLUSTER, or DEFAULT LEVEL.
- For each Key, identify the defining level of the merchandise hierarchy and location hierarchy.
- The cross products information is generally loaded only once.

Cross Products Information Specification

Table 3–9 Cross Products Information Standard Interface Specification

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
INTERSECT_NAME	The key name (OPTIMIZATION, SALES, WORKSHEET, CLUSTER, or DEFAULT LEVEL)	String	50	N
MERCHANDISE_LEVEL	The defining level within the hierarchy	String	50	N
LOCATION_LEVEL	The defining level within the hierarchy	String	50	N

Location Hierarchy Levels Standard Interface (ASH_LHL_TBL)

The location hierarchy levels interface is used to specify the names of a retailer's location levels and their order.

Technical Notes

The following list provides details to consider regarding the lh levels data.

- The Chain level should always be defined as 1.
- The sequence of level numbers must begin with 1 and increase in increments of 1, without any gaps in the sequence.
- The location hierarchy levels information is generally loaded only once.

LH Levels Specification

Table 3–10 Location Hierarchy Levels Standard Interface Specification

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
LOCATION_LEVEL	The name of the location level	String	50	N
LEVEL_SQC	The sequence number of the level	Integer	2	N

Merchandise Hierarchy Levels Standard Interface (ASH_MHL_TBL)

The merchandise hierarchy levels interface is used to specify the names of a retailer's merchandise levels and their order.

Technical Notes

The following list provides details to consider regarding the mh levels data.

- The Chain level should always be defined as 1.
- The sequence of level numbers must begin with 1 and increase in increments of 1, without any gaps in the sequence.
- The merchandise hierarchy levels information is generally loaded only once.

MH Levels Specification

Table 3–11 Merchandise Hierarchy Levels Standard Interface Specification

Field Name	Field Description	Data Type	Maximum Length	Nullable Y/N
MERCHANDISE_LEVEL	The name of the merchandise level	String	50	N
LEVEL_SQC	The sequence number of the merchandise level	Integer	2	N

4

Historical Analysis

This chapter contains the following:

Introduction

Historical analysis consists of baseline analysis followed by tae analysis. Baseline analysis defines the baseline behavior of a product absent promotion activity. Tae analysis defines the lift provided by a promotion. This chapter provides operational details about historical analysis.

Baseline Analysis

Baseline metrics are the measurements of the behavior of an item during the time that it is not being promoted. These metrics are used as the standard of comparison in order to determine the effect of the promotion on the item. The difference between the baseline behavior of an item and that item's behavior during a promotion is called the *lift*.

The following metrics are measured by the baseline utility:

- Rate of sale (ROS)
- Visit rate (VR)
- Gross sales
- Gross margin
- Baseline mode price

Baseline metrics (sums and averages) are measured per store per item per period (typically a week). The baseline application generates baseline measurements only for item/store pairs that have at least one day of non-promotion sales and no promotion sales for a given period.

Running Baseline

The baseline utility has the following options:

baseline.sh -startdate <*yyyy-mm-dd*> **-enddate** <*yyyy-mm-dd*> **-append**

The baseline utility has the option to specify the period length in days. The default value is 7. When the start and end dates are specified as command line options, the start date must be the first day of the first period and the end date must be the last day of the last period.

The append argument is optional. If this argument is included, the results of each successive run of the utility are appended to the dataset. If the output dataset exists and this option is not specified, then an error occurs. If the output dataset has not been created and this option is specified, there is no effect.

For information about the command options for kde.sh baseline, see Chapter 8, "Technical Reference".

Best Practices

For best analytical results, the following practices are recommended:

- The amount of transactional data processed by the baseline utility should be at least for 54 - 56 weeks and optimally for 116 consecutive weeks.
- Certain items may require less data than the minimum recommendation to provide reasonable results. For example, commodity-only items or holiday items may require only 13 weeks of transactional data.
- Large amounts of transactional data should be processed in separate batches by kde baseline. For example, if three years of data is available, process each year separately.
- After the initial execution of kde baseline, use the -append option with every subsequent execution of the utility in order to append the metrics produced to a single history data set.
- Thresholds for adequate baseline data can impact coverage vs. accuracy.

TAE Analysis

TAE analysis measures the effectiveness of promotions, based on the baseline data generated by kde baseline. The output of Tae is a Promotion Scorecard, containing metrics and statistics for Focus Items (items that were part of a promotion).

TAE calculates the following metrics for specific focus items and the overall ad performance:

- The average rate of sale, visit rate, and mode price for a focus item during one or more baseline periods.
- The total rate of sale, visit rate, and price for a focus item during a promotion. Incremental metrics can be derived by subtracting the baseline metrics from the ad-period metrics.

- The amount of drag or pull of non-ad items associated with or allocated to a focus item.
- The incremental and cross-sell effects of a focus item on other items.
- The total effect of the ad relative to the incremental ad-item and non-ad-item rate of sale, visit rate, revenue, and profit margin.
- Comparisons with similar ad events.

For information about the command options for kde.sh tae, see Chapter 8, "Technical Reference".

Running TAE

The tae utility has the following options:

Note: The -adeventid <ID> option is required. Either -algorithm <algorithm id> or -basewindow <start, end, pmin> is required. You can specify both, but only one is required.

tae.sh -adeventid <*ID*> {**-algorithm** <*algorithm id*>} {**-basewindow**}

where

- **adeventid** requires the ID of the ad event.
- **algorithm** can be:
  - **py** - a baseline window one year prior to the current date is used.

    The number of weeks in the window is specified in the kde.properties file in com.netperceptions.kde.prior.year.weeks. The default value is 3.

    The minimum number of periods in the window is specified in the kde.properties file in com.netperceptions.kde.prior.year.pmin. the default value is 2.
  - **baa** - baseline windows before and after the ad period are used. The size of each baseline window is configured separately. It is thus possible to set one of the weeks (before or after) to a value of 0 in order to provide only one window.

    The window settings are specified in the kde.properties file, as follows.

    * com.netperceptions.kde.baa.before.weeks - the number of weeks before the promotion to be used in the base window. The default value is 3.
    * com.netperceptions.kde.baa.before.pmin - the minimum number of periods before the promotion that can be used in the base window. The default value is 2.
    * com.netperceptions.kde.baa.after.weeks - the number of weeks after the promotion to be used in the base window. The default value is 3.
    * com.netperceptions.kde.baa.after.pmin - the minimum number of periods after the promotion that can be used in the base window. The default value is 2.
  - **cafe** - both py and baa are used.
- **basewindow**, which takes three options:
  - **start** - the date of the first day of the first period to be used.
  - **end** - the date of the last day of the last period to be used.

- **pmin** - the minimum number of periods in which the item/location combination must be found in the history data. The value for pmin must be between the number of periods specified in the window and 1.

Notes:

Specifying a promotion inside the window causes an error.

More than one -basewindow can be specified on the command line.

Some Examples

Here are two examples.

A promotion from Sunday, September 10, 2006 to Saturday, September 16, 2006 can be specified in two ways (assuming default settings for prior.year.weeks and prior.year.pmin):

tae.sh...**-algorithm**...

tae.sh...-basewindow 2005-09-04,2005-09-24,2...

A promotion from Sunday, September 10, 2006 to Saturday, September 16, 2006 can be specified in two ways (assuming default settings for baa.before.weeks, baa.before.pmin, baa.after.weeks, and baa.after.pmin):

tae.sh...**-algorithm** baa...

tae.sh...**-basewindow** 2006-08-20,2006-09-09,2 **-basewindow** 2006-09-17,2006-10-07,2

Substitution

The kde tae utility requires baseline data generated by the kde baseline utility. Baseline data may not be available because the item is new or because the item is frequently on promotion at the given location.

If a focus item does not have enough sales history to calculate the baseline, it is possible to use the behavior of a like item to calculate the baseline for the new item. It is also possible to define how much data is considered reliable for calculating the baseline. Adequate baseline parameters are based on the count of the number of uncontaminated baseline periods that occur in each baseline window.

If no baseline data is available, the kde tae utility generates a scorecard with zero baseline numbers and flags the bl_subst_code column.

To assign a like item to another item that does not have adequate sales data, create a mapping table that maps new items to old items that have similar sales behavior. TAE substitutes the baseline data from the old item. When baseline substitution occurs for a store/item combination, the corresponding TAE output contains a flag that indicates that the substitution occurs and a field specifies the substituted item.

Like Item Mapping

The Like Item mapping information is stored in a dataset file in text format. The data is pipe-delimited. A default dataset file, item.sub.ds, is created during installation.

The format for the text file is:

Item | Like Item | Weight

where

- Item is pi_id from merchandise_tbl
- Like Item is pi_id from merchandise_tbl

- Weight is not currently used, so a default value of 1.0 should be entered.

Choosing Baseline Windows

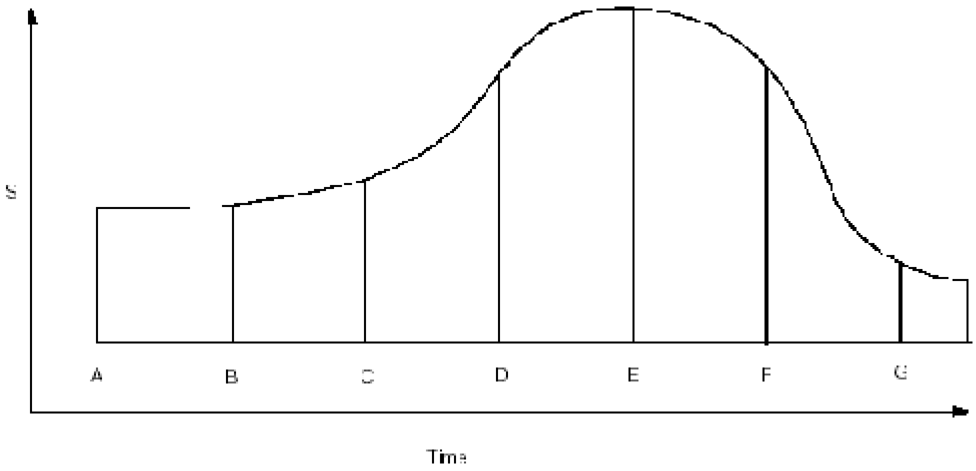
The baseline window used is selected because it is representative of what the sales would be in the absence of a promotion. Typically, the baseline window is selected from last year’s data. The periods selected are before and after the promotion.

Seasonality

Seasonality is the variation in the performance of an item based on the time of year. The baseline is more accurate if it incorporates seasonal effects. Baseline windows can be defined for specific time ranges in order to address seasonal variation. For example, one window can be defined for a period before the focus event, one window can be defined for a period after the focus event, and one window can be defined for a period similar to the one that occurred a year ago.

An Example

The following figure illustrates a seasonal sales curve. Baseline windows that are selected to retrieve seasonal baseline data can be applied to this curve.



Consider a TAE analysis of a focus event that occurs between point B and C on the curve and a focus event for a seasonal item that approaches its optimal sales performance for the year during that event. The use of a single baseline window (A-B) before the focus event and before the start of the season can result in an overstated lift. The season and the promotion both contribute to the lift. One way to mitigate this effect is to specify baseline windows on either side of the focus event. The rising sales of the season that occur during the baseline window and after the focus event (C-D) raise the baseline and lessen the lift.

A promotion of the same focus item during a later period (C-D) using the same or similar baseline windows (B-C or D-E) shows how multiple windows can mitigate the effects of the seasonal curve on the baseline. The higher seasonal curve causes the baseline of both windows to be greater than the out-of-season baseline. The D-E window is higher on the seasonal curve than during the focus period. However, the two windows are averaged and the result is a higher baseline and lower lift. Note that the use of windows that occur after the focus event affects when the analysis can be done.

The selection of the baseline window involves a compromise between accuracy and coverage. Items sell at different rates. Windows that are too narrow may not have adequate data for slow-moving items, which reduces coverage. Windows that are too large capture out-of-season data, which reduces accuracy.

TAE Results

The output of the TAE analysis is a Promotion Scorecard. The following metrics are included in the TAE Scorecard Report.

Ad Item Metrics are for ad items only.

- **ad_item_price** - the mode of price of the ad item
- **ad_item_rosale** - the sum of the item
- **ad_item_visit_rate** - the visit rate or the number of market baskets that the item was in
- **ad_item_sales** - the total sales amount of the ad- item
- **ad_item_gm** - the total gross margin amount of the ad-item
- **ttl_ad_days** - the number of days in the ad period that the item actually sold

Affinity Correlation
Note: These values will be zero if the ac_level is not specified in the MB_Detail data.

- **ad_item_ac_sales** - Affinity average sales
- **ad_item_ac_gm** - Affinity average gross margin
- **ad_item_pr_sales** - Revenue average sales
- **ad_item_pr_gm** - Revenue average gross margin
- **ad_nonad_sales** - Non-ad average sales
- **ad_nonad_gm** - Non-ad average gross margin

These items show the status of baseline data being used.

- **bl_subst_code** - the substitution code
- **bl_subst_item** - the item used if substitution is used

Where the values for bl_subst_code are:

- 0 - the focus item has enough baseline
- 1 - the focus does not have enough baseline, but was an item was substituted that does have enough baseline.
- 2 - the focus item does not have enough baseline and substitution is turned off.
- 3 - the focus item does not have enough baseline and a substitution item was not available.
- 4 - the focus item does not have enough baseline and the substitution item does not have enough baseline either.

If substitution is turned on, the TAE will classify the focus items into either class 0, 1, 4, or 5. Focus items that have enough baseline go into class 0. The others are joined with the substitution dataset. Any records without substitution matches go into class 3. Successful substitutions are then checked to ensure they have enough baseline. Those that do go into class 1. The others go into class 4.

If substitution is turned off, only class 0 and 2 are used. Those focus items with enough baseline go into class 0. Those without go into class 2.

These are the averages for the periods in the chosen baseline window for the item or substitute item. Theses items are all zero if the bl_subst_code is 2, 3 or 4 as explained below.

- **ttl_base_periods** - total number of periods in the baseline window
- **bl_item_rosale** - average rate of sale for the item in the baseline window
- **bl_item_sales** - average sales for the item in the baseline window.
- **bl_item_visit_rate** - average visit rate or market basket count for the item in the baseline window.
- **bl_item_gm** - average gross margin for the item in the baseline window
- **bl_item_price** - average price for the item in the baseline window.

These are the same as the ad-items explained above with exception that these were generated from the MB Detail data from the selected baseline window. These values are included when the option "blalloc" is specified. These values will be zero if the ac_level is not specified in the MB Detail data.

- **bl_item_ac_sales**
- **bl_item_ac_gm**
- **bl_item_pr_sales**
- **bl_item_pr_gm**
- **bl_nonad_sales**
- **bl_nonad_gm**

These are the quality metrics. These values will be zero unless the option *quality* is specified. They really seem to be market basket analysis instead of quality.

- **ad_mb_item_only** - the number of promotion baskets that contain ad item alone
- **ad_mb_item_ad** - the number of promotion baskets that contain ad item and some other ad items, but no nonad items
- **ad_mb_item_nonad** - the number of promotion baskets that contain ad item and at least one nonad item, but no ad items other than the ad item
- **ad_mb_item_adnonad** - the number of promotion baskets that contain ad item, other ad items, and at least one nonad item
- **ad_item_othad_ros** the count of the remaining item in the market basket

Best Practices

For the best analytical results, the following practices are recommended:

- The kde tae utility requires 13 - 26 weeks of baseline data in order to identify candidate baseline weeks:
  - Commodity merchandise requires at least 13 weeks of data
  - Slow-moving merchandise requires at least 26 weeks of data
  - Seasonal merchandise requires at least 19 weeks of data
- The value for the affinity index threshold (the -minlift option) should be 2 - 5. A higher value finds fewer affinities, but with a greater degree of confidence.

- In order to produce the detailed results required to generate meaningful Promo Scorecard reports, it may be necessary to run both kde baseline and kde tae more than once. In this case, both utilities should be run at the same product levels.
- Baseline must be built from seasonally-appropriate data. The KDE modules for measurement and prediction support the use of baseline windows to help mitigate the effect of seasonal sales. Historical, year-old data is more important for a predictive baseline than for a measurement baseline. Try experimenting with various baseline window widths in order to find a compromise between seasonal accuracy and coverage for slow-moving items.

ARM Analysis Process

Association Rule Mining (ARM) is the technique used by the analytics to compute a variety of association metrics. These metrics describe how promoting one product affects the sales of other products in the historical dataset.

1. To run ARM, use <install-dir>/modules/pce/bin/arm.sh. For example:

   arm.sh -minconfidence 0.05 **-runid** 1 **-maxsetsize** 2

2. Once the ARM data has been computed, it can be loaded into the database (RDM) for further analysis and reporting.

```
> arm_stage.sh
> cd
${installdir}/modules/Database/SEQUOIASchema/install/oracle/SEQUOIASchema/scrip
ts
> pl_refresh_ARM_sets.sh ${installdir}/config rdm_plexports.sh
```

5

Market-Basket Analysis

This chapter contains the following:

- "Introduction" on page 1
- "Configuration" on page 1
- "Analysis" on page 1

Introduction

Promote includes tools to use in order to both perform market-basket analysis of client data and populate the Retail Data Mart (RDM) with information about ad effectiveness. The data in the RDM is customized and aggregated so that it can be easily accessed for reporting. Note that the RDM cannot be set up until the POS data has been loaded into the Promote database.

Configuration

Prior to running the market-basket analysis, you must configure the system.
In the promoteRDM.properties file, which can be found in <*InstallationDirectory*>/modules/Database/SEQUOIASchema, assign values as follows:

```
CircularVehicleExternalName=<External ID of Vehicle>
CircularVehiclePagesAttributeExternalName=<Name of Property That Holds Page Count>
```

Analysis

Complete the following steps to run the market-basket analysis:

1. Run the script **pl_create_summary_tables.sh**. This script creates the database objects required to process and store the market-basket analysis at the Class, Department, and Division levels.

2. Run the script **pl_create_summaries.sh**. This script performs two functions.

   It reads the market-basket details from the POS data table and extracts the incremental data (data that has been inserted since the last date in the record). An incremental table is populated and the internal IDs are assigned for merchandise, location, and date.

   It then creates summary market-basket information at the Class, Department, and Division levels.

3. Run the script **pl_create_promote_ir_views.sh**. This script creates the views that provide access to the promotional summary information at different levels of the merchandise and location hierarchies. The views analyze data at the Class, Department, and Division levels of the merchandise hierarchy and at the Chain and Store levels of the location hierarchy.
4. Run the script **pl_create_RDM_summary.sh**. This script creates the tables and views necessary to analyze vehicle allocation information at the Class, Department, and Division levels of the merchandise hierarchy. Synonyms and views are created that are helpful for report writing.
5. Run the script **pl_load_promote_RDM.sh**. This script creates the promotion information needed by the RDM.

6

Predictive Modeling

This chapter contains the following:

Introduction

Analytical analysis analyzes trends and performs What-if scenarios. Like Historical Analysis, Analytical Analysis is performed by the engine.

Forecasting is a feature that allows a user to interactively run what-if scenarios, based on the data provided.

The forecasting configuration points are analytical modeling files, database views, and database tables.

The **model apply** application takes an input model and an input dataset and outputs the results of applying the model to the input data. This utility is effectively the "what-if" predictive tool.

Build Models

The Build Model steps includes the following.

Adjusting the Model

If needed, adjust the model, as follows:

- Configure Promote metadata
- Ensure that the data feeds have this data
- Update the model to add or adjust the attributes

The Promote Calc Engine models past promotion performance using a set of configurable attributes. To control the attributes used in the modeling process, edit the model_config.xml file located in <install-dir>/modules/pce/data/models/default. This is also the file to configure other options of the modeling process, such as the modeling algorithm, training time periods, minimum lift, and seed values.

admodeldataprep

The admodeldataprep analytic merges data from the model building source data sets into two data sets that are segmented by time. Typically, the earlier data is used for training and the later data is used for testing.

In addition to merging and time-segmenting the data, admodeldataprep also synthesizes two attributes from the input data: item lift mean and seasonality scale factor.

- The item lift mean is a product attribute that indicates, generally, how the item responded to promotion in the past.
- The seasonality scale factor is a measure of how the product sold at each store-week of the year.

The analytic also prunes and filters the data to avoid training on noise and statistically insignificant events.

admodelbuild

The admodelbuild analytic application builds models using a number of predictive and clustering algorithms. A common property of these algorithms is that they take an input stream of training data and produce a model that describes the data and that is suitable for input to the admodelapply application.

The admodelbuild application takes as input a mining schema, specified by the option -miningSchema, and produces a PMML (Predictive Modeling Make-up Language) model file.

adgenstats

The adgenstats utility uses the model test and training data to generate a chain-level stats file for use by modelapply, with a particular model, when aggregation is on. These chain-level statistics are not necessary if aggregation is off. The input to adgenstats is the training data from modelbuild. The output is a file that contains chain-level statistics. This file is used by modelapply to compute the statistics that are necessary to z-score the aggregates.

admodeldeploy

The modeldeploy application publishes the model information to the application database.

Building Predictive Baselines

The following utilities are used for building predictive baselines.

adblcompute

This utility uses the baseline data to generate projected (future) baseline information for all sold items. It also performs checks of the start and end dates of the baseline window.

adblprep

Once the predicted baseline data is built (adblcompute), adblprep utility renames some columns and adds customer-specific attributes, such as seasonality scale factor and liftmean data. The resulting baseline data segments are organized, by specific model, and stored in a directory, specified by "-bldir" attribute.

Using blcompute

Promote provides a means to define Like Items and Like Stores for new items and new stores that do not yet have sales data. The utility blcompute is used for Like Item substitution and Like Store substitution for prediction. Unlike tae substitution (for Like Items only), blcompute always performs substitutions when requested, so there is no feedback that substitution has occur for a given Focus Item or Focus Store.

The Like Store mapping information is stored in a dataset file in text format. The data is pipe-delimited. A default dataset file, store.sub.ds, is created during installation.

The format for the text file is:

Store | Like Store | Weight

where

- Store is pi_id from location_tbl.
- Like Store is pi_id from location_tbl.
- Weight is not currently used, so a default value of 1.0 should be entered.

The Like Item mapping is discussed in Chapter 4, "Historical Analysis", as part of the tae substitution discussion.

7

Affinity Modeling

This chapter contains the following:

Introduction

This chapter describes the requirements for the affinity-modeling feature, which allows the Promote product to produce predictions about the positive and negative sales impacts of planned promotions due to halo and cannibalization (substitution) effects.

The three types of affinity effects are:

- Complimentary halo effects of planned promotions
- Indirect (traffic) effects of planned promotions
- Cannibalization effects of planned promotions

Understanding the three types of affinity effects helps managers make better advertising decisions.

The Affinity Estimation Process

The Affinity Parameter Estimator (APE) is used by an analyst to discover, analyze, and refine the elasticity factors used during prediction. The APE includes a component that is responsible for estimating affinity effects. The APE is a separate application that is not included with the Promote CD.

The affinity process and the part the APE plays in that process is as follows:

1. The POS data is mined by the Promote Calc Engine (PCE) into aggregate form (store, item, week).
2. The aggregate data is saved to the database.
3. The analyst builds a configuration for the Promote Analytics module. This configuration details how to prepare the data for the APE. The analyst calls a process within the Promote Analytics module that performs the data preparation. The data is then saved to the database.

4. The analyst initializes the APE for the estimation of the affinity parameters.
5. The APE receives the previously prepared data from the application database and uses it to estimate the affinity parameters.
6. The APE provides the affinity parameters to the database.
7. The end user requests the predicted performance of an item for a planned promotion.
8. The Promotion Manager passes the request to the PCE.
9. Using the stored affinity parameters, the PCE computes the affinity information for the selected product.
10. The projected affinity effects are passed back to the Promotion Manager for display to the end user.

Choosing a Tool for APE Analysis

Any software package that supports basic statistical operations such as linear and non-linear modeling can be used as an APE. The instructions and examples in this guide assume that the "R" package (http://www.r-project.org/) is being used; however, any similar program such as S-Plus or SAS can also be used.

The tool used for APE analysis is not included with the product. However, the product CD does include some example scripts in the <*InstallationDirectory*>/modules/pce/ape directory. These example scripts are "R" compatible, but can be modified so that they can be used with other analytic packages.

APE Hierarchy

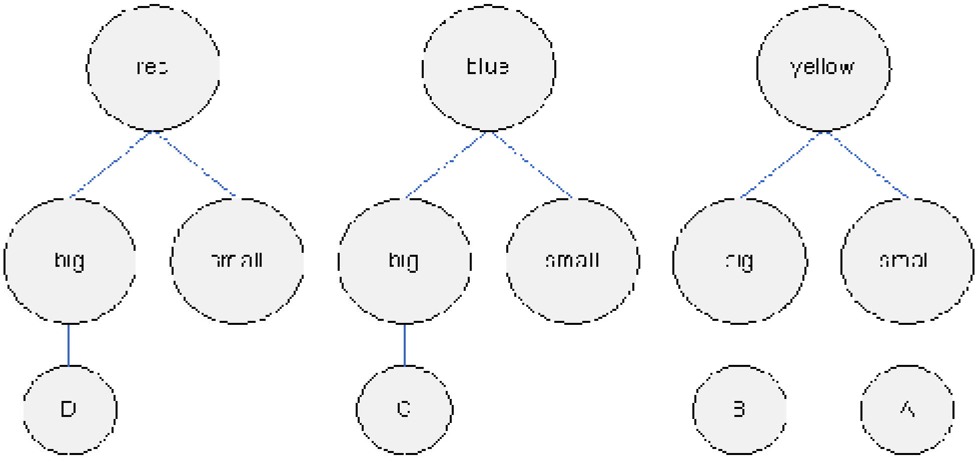
The APE uses a tree-like merchandise structure for performing predictions. In this way, it can identify affinity effects that exist at various levels above the item level. The APE uses its own independent hierarchy instead of the existing Promote hierarchical merchandise structure. This allows the user the flexibility to determine the best way to group items to suite their affinity effect prediction needs. While the existing Promote merchandise hierarchy has one root node, the APE hierarchy can have more than one, based on how the APE hierarchy has been configured. Affinity parameter estimation is only performed within an APE hierarchy tree, not across trees. Each item within the APE hierarchy belongs to one and only one hierarchy tree.

The APE hierarchy is built using merchandise attributes, such as size, color, or parent merchandise hierarchy node ID (from the existing hierarchy). To define a level in the APE hierarchy, such as the root level, a merchandise attribute or a set of attributes is specified. Under each node in the level above this level, a node is then created for every unique value of this attribute or for every unique combination of attribute values if more than one attribute is used. The set of attributes that can be used to define APE hierarchy levels is limited to the set that already exists within the client's existing merchandise data.

Once every level in the APE hierarchy is defined using merchandise attributes, the hierarchy can be built and items can then be assigned to their respective nodes. For example, if the first level in the APE hierarchy is defined by the attribute *color* and the set of unique values of this attribute are *blue*, *red*, and *yellow*, then three nodes are created at this level, one for each value of the color attribute. If the second level of the hierarchy is defined by the attribute *size* and the set of unique values for this attribute are *small* and *big*, then six nodes are created at this level, one set each of big and small nodes under the blue, red, and yellow nodes. Items whose color attribute have the value *yellow* and whose size attribute have the value *small* would be assigned to the first level node yellow and the second level node small under the yellow node. Note that, in this scenario, a small node exists under several first level nodes. Which of these small nodes an item belongs to depends on which first level nodes the item belongs to.

Item	Color	Size
A	yellow	small
B	yellow	big
C	blue	big
D	red	big

Nodes:



The end result of this mapping is a new set of database tables that defines the APE merchandise hierarchy, mapping the item (SKU-level) IDs from the pre-existing merchandise hierarchy into their respective APE hierarchy nodes.

Hierarchy Tree Depth

Each APE hierarchy tree has three levels, 0-2, where the 0th level is the root level. In the simplest scenario, an APE hierarchy is created by defining one set of level zero (root level) merchandise attributes, one set of level one merchandise attributes, and one set of level two merchandise attributes. However, this "one size fits all" hierarchy may not an acceptable way to segment a client's entire merchandise set. Take for example the case of a client who wishes to define two APE hierarchy trees, one for beverages and one for shoes. With beverages, the client might be most interested in seeing the interactions between flavors. Style might be a more important trait with shoes. In this situation it would be useful to be able to define first level nodes for the beverage tree by a flavor attribute and to define first level nodes for the shoe tree by a size attribute.

Anchor Nodes

The use of anchor nodes is one way for the user to apply the above type of discrimination between sets of APE hierarchy trees. The user simply associates a node from the system's existing merchandise hierarchy with a particular APE tree definition (which might then produce more than one tree). Only items that are children of the merchandise hierarchy node are allowed into this tree (or trees). This creates a type of filter on the APE hierarchy tree (or trees) that uses this anchor node. However, anchor nodes limit the user to filtering only by a particular node in the system's existing merchandise hierarchy and can only be used at the tree level.

The user must assign each APE hierarchy tree set definition an anchor node. Requiring that each tree set have an anchor node ensures that no item appears in more than one tree. Anchor nodes are created by defining keys, but rather are a required property of the tree set definition itself.

Merchandise Mapping xml Configuration File

An XML-based configuration file (found in <*InstallationDirectory*>/modules/pce/etc/APE_Hierarchy.xsd) is used to configure the APE hierarchy.

xml Tag Definitions

The following definitions are used in the xml configuration file.

·<APE_hierarchy> - The <APE_hierarchy> begins an individual APE hierarchy. The set can contain multiple hierarchy trees, depending on its configuration.

·<treeset> - A <treeset> tag defines the beginning of a particular hierarchy tree configuration

Property	Required	Description
name	No	The name for the <treeset>
mh_node_id	Yes	The anchor node ID, a merchandise hierarchy client load ID
mh_node_level_desc	Yes	The anchor node level description, a merchandise hierarchy client level description. Required because the merchandise hierarchy client load node IDs are not unique across levels.

·<level> - The <level> tag begins a level definition within a <treeset> or another <level>.   The level defined directly under a <treeset> is the 0th, or root, level.

Property	Required	Description
position	Yes	The integer representing the position of the level within the hierarchy. Root level is 0, first level is 1, etc.

·<key> - A <key> node provides filtering for a particular tree, or for a level within a tree, based on a merchandise attribute column and the value of that column.

Property	Required	Description
column	Yes	The merchandise attribute field name.
value	Yes	The value of the merchandise attribute field that <treeset> or <level> applies to.

Example xml file

```
<APE_hierarchy>
<!-- Typical tree set definition -->
  <treeset name="Shoe Tree" mh_node_id="12368"
  mh_node_level_desc="CLASS">
      <level>
            <attr column=CATGORY/>
            <level>
                  <attr column=BRAND/>
                  <level>
                        <attr column=SIZE/>
                  </level>
            </level>
      </level>
    </treeset>
```

The Results of Merchandise Mapping

The results of the merchandise mapping are a set of database tables that describe the APE hierarchy and map client item IDs to their respective APE hierarchy nodes.

APE Hierarchy Node Naming

Each APE hierarchy node should be given a unique name and description that has meaning to the analyst and the client. During the merchandise configuration process, node names are generated using a combination of the tree set name and the column names and values used to generate the node.

Error Checking

Items cannot appear in more than one APE hierarchy tree.

Performing Affinity Analysis

To perform affinity analysis, do the following:

1. Build an affinity tree definition in XML. See “Merchandise Mapping xml Configuration File” on page 4 for details.
2. Load the affinity tree. To do this, load the xml configuration file using the commands below. The first command, **apbuilder_run.sh**, loads the xml configuration file into the database. The second command, **pl_create_ape_hierarchy.sh**, creates the hierarchy based on the configuration information provided.

   ```
   > apebuilder.sh -config ape_configuration.xml
   ```

```
> cd
${installdir}/modules/Database/WALNUTSchema/install/oracle/WALNUTSchema/scripts
> pl_create_ape_hierarchy.sh ${installdir}/config promote_plexports.sh
```

3. Once the APE configuration is built and processed, summary sales data must be built which matches the loaded APE configuration tree. This prepares the data for analysis.

   ```
   > cd
   ${installdir}/modules/Database/WALNUTSchema/install/oracle/WALNUTSchema/scripts
   > pl_create_ape_summary.sh ${installdir}/config promote_plexports.sh
   ```

4. Use the APE to perform the affinity parameter estimation. It is assumed you have Perl installed on your system.

   To run the APE, do the following:

   1. Download latest software called "R", which is freely available, from www.r-project.org and follow the installation instructions for your platform.
   2. Install the ROracle package required by APE:

      > R CMD INSTALL ROracle
   3. The Promote distribution includes the R package aper-1.0.tar.gz, the script mainAPE.R, and the configuration file options.R. These files can be found in <*InstallationDirectory*>/modules/pce/ape.

      To install the APE:

      >R CMD INSTALL aper
   4. The file options.R contains configurable APE settings. You must edit the file to define DB login parameters:

      DB.HANDLE='USER/PASSWORD@PROMOTE_INSTANCE'

      The other options can be left at their default settings. The comments in options.R describe the default settings.
   5. To run APE:

      >R CMD BATCH mainAPE.R ape.log

5. Once APE analysis is complete, the results need to be loaded back into the application database. There should be two files, one named bee_ape_price_elasticity.txt and the other named bee_ape_promo_elasticity.txt.

   ```
   > cd
   ${installdir}/modules/Database/BEECHSchema/install/oracle/BEECHSchema/scripts
   > pl_stage_promote.sh ${installdir}/config promote_plexports.sh
   ${installdir}/modules/Database/BEECHSchema/install/oracle/BEECHSchema/controlfi
   les ${deploy}/bee_ape_price_elasticity.txt ${deploy}/bee_ape_promo_
   elasticity.txt
   > pl_load_ape.sh ${installdir}/config promote_plexports.sh
   ```

8

Technical Reference

This chapter contains the following:

Introduction

This chapter contains useful reference material, including command options.

kde.properties

The following is a subset of the properties included in kde.properties. These properties are configured during the installation procedure.

Table 8–1 kde.properties

Property	Description
com.netperceptions.kde.temDirPath=/ldisk/s000/qapromo/builds/12.0.1/2/modules/pce/temp	The temp space allocated to the PCE. It should be cose to the size of the dataset storage.
com.netperceptions.kde.numberOfCPUs=2	The number of CPUs that the host machine has the PCE should optimize to. Dual core and hyper-treading CPUs should count as 2 each.
com.netperceptions.kde.rmi.server.port=1239	The port for the PCE Prediction Enginie to run. The default port is 11269.

Table 8–1 (Cont.) kde.properties

Property	Description
com.netperceptions.kde.rmi.server.ServerDebug=false	Specifies whether the Prediction Engine runs in debug mode. In debug mode, the temporary files for a request are not deleted afterwards.
com.netperceptions.kde.rmi.server.ModelApplyDebug=false	Specifies whether AdModelApply is executed in debug mode by the Prediction Engine.
com.netperceptions.kde.rmi.server.PromoDataValidation=false	Specifies whether the Prediction Engine should do additional validation of the data received.

kde-local.vars

The /etc/kde-local.vars file contains environment variables used by promote shell scripts. It includes the following settings:

- DEBUG_MODE - determines if the steps in model_build.sh are printed to the console. If this is set to anything, then it is active.
- DWH_HOST - the host name of the database server used by the system.
- DWH_INSTANCE - the instance name of the database server used by the system.
- DWH_ALIAS - the alias of the database server used by the system.
- DWH_USERNAME - the username for the database used by the system.
- DWH_PASSWORD - the password for the database used by the system.
- DWH_TYPE - the database type used by the system. Must be set to *oracle*.
- DWH_PORT - the port number of the database server used by the system.
- TEMP_DIR - the temporary directory sued by model_build.sh.
- LOCATION_SUMMARY_LEVEL - the level in the location hierarchy at which analysis is done by the system. This setting is client-specific.
- MERCHANDISE_SUMMARY_LEVEL - the level in the merchandise hierarchy at which analysis is done by the system. This setting is client-specific.

If it is necessary to change any of the settings in this file, it is recommended that you configure kde-local.vars, shown in the following code example.

```
#!/bin/bash

#uncomment DEBUG_MODE to get more messages
DEBUG_MODE=true

#
# Database settings
#
DWH_Host=dev-db-s02
DWH_INSTANCE=pmdev10
DWH_ALIAS=pmdev10
DWH_USERNAME=ssanchez
DWH_PASSWORD=ssanchez
DWH_TYPE=oracle
DWH_PORT=1521
```

```
TEMP_DIR=/ldisk/s000/qapromo/builds/12.0.1/2/modules/pce/temp

# SUMMARY LEVELS
LOCATION_SUMMARY_LEVEL=7
MERCHANDISE_SUMMARY_LEVEL=4
```

kde.sh baseline

The baseline.sh script calls kde.sh baseline.

Here is an example of kde.sh baseline:

```
kde.sh baseline
    -indb dbtype=oracle,host=dbhost,instance=retaildb,user=myuser,
        password=password,table=MD_DETAIL
    -out type=binary,name=/bigdisk/dataset/baseline.out.ds
    -indbMerchMap dbtype=oracle,host=dbhost,instance=retaildb,user=myuser,
        password=password,table=PR_MERCHANDISE_MAP_VIEW
    - indbLocMap dbtype=oracle,host=dbhost,instance=retaildb,user=myuser,
         password=password,table=PR_LOCATION_MAP_VIEW
    -outMerchOutCast type=binary,name=/bigdisk/dataset/merch.outcast.ds
    -outLocOutCast type=binary,name=/bigdisk/dataset/locn.outcast.ds
    -genxml baseline.xml
    -backtrace
    -startdate 2002-12-29
    -enddate2003-03-29
    -append
```

The following table lists the complete set of options for the command.

Table 8–2 kde.sh baseline Command Options

Option	Option Description	Required/ Optional?
-out <out DS attributes>	Baseline output dataset. Takes attributes.	Required
-startdate *yyyy-MM-dd*	Set start of date range for baseline.	Required
-ac_level *column-name*	Name of column to use for ac_level.	Optional
-ad_ind *column-name*	Name of column to use for ad_ind.	Optional
-[no] append	Append output to dataset.	Optional
-[no] backtrace	Display detailed stack backtrace.	Optional
-[no] debug	Turn on debugging.	Optional
-enddate *yyyy-MM-dd*	Set end of date range for baseline.	Optional
-ext_margin_amt *column-name*	name of column to use for ext_margin_amt.	Optional
-ext_retail_amt *column-name*	Name of column to use for ext_retail_amt.	Optional
-genxml *file-name*	Name of file to write pre-processed generated xml to.	Optional
-[no] help	Display message.	Optional
-in <in DS attributes>	Detail input dataset. Takes attributes.	Optional
-inLocMap <in DS attributes>	Location Map input dataset. Takes attributes.	Optional
-inMerchMap <in DS attributes>	Merchandise Map input dataset. Takes attributes.	Optional
-indb <in JDBC connection attributes>	detail input JDBC connection. Takes attributes.	Optional

Table 8–2 (Cont.) kde.sh baseline Command Options

Option	Option Description	Required/ Optional?
-indbLocMap <in JDBC connection attributes>	Location Map input JDBC connection. Takes attributes.	Optional
-indbMerchMap <in JDBC connection attributes>	Merchandise Map input JDBC connection. Takes attributes.	Optional
-loc_client_load_id *column-name*	Name of column to use for loc_client_id.	Optional
-margin	Enable margin metric calculations.	Optional
-merch_client_load_id *column-name*	Name of column to use for merch_client_load_id.	Optional
-nelements *cnt*	Number of elements to read (-1 to all).	Optional
-options *file-name*	Name of property file containing values for command line options.	Optional
-outLocOutCast <out DS Attributes>	Location unmapped records output dataset. Takes attributes.	Optional
-outMerchOutCast <out DS Attributes>	Merchandise unmapped records output dataset. Takes attributes.	Optional
-outdbLocOutCast <out JDBC connection attributes>	Location unmapped records output JDBC connection. Takes attributes.	Optional
-outdbMerchOutCast <out JDBC connection attributes>	Merchandise unmapped records output JDBC connection. Takes attributes.	Optional
-runid *identifier*	Run identifier.	Optional
-[no] ssd	Input dataset contains sku/store/day aggregated data.	Optional
-txn_date *column-name*	Name of column to use for txn_date.	Optional
-txn_id *column-name*	Name of column to use for txn_id.	Optional
-txn_total *column-date*	Name of column to use for txn_total.	Optional
-units_sold *column-name*	Name of column to use for units_sold.	Optional

Option Attributes

Some of the options in Table 8–2, " kde.sh baseline Command Options" take attributes.

The options **-out, -outLocOutCast,** and **-outMerchOutCast** take the following attributes:

- **type** - the type of output set. Values are *text*, *fixed-text*, and *binary*.
- **schema** - the pathname of the schema file.
- **compression** - the compression method used. Values are *none* and *gzip*.
- **name** - the pathname of the dataset. This is required.
- **timezone** - the timezone used.
- **encoding** - the encoding used.
- **delimiter** - the delimiter used.
- **locale** - the locale used.
- **mode** - the mode used. Values are *append* and *overwrite*.

The options **-in, -inLocMap,** and **inMerchMap** take the following attributes:

- type - the type of output set. Values are *text*, *fixed-text*, and *binary*.
- **schema** - the pathname of the schema file.
- **compression** - the compression method used. Values are none and gzip.
- **name** - the pathname of the dataset. This is required.
- **nElements** - the number of elements to read. The default value is *all*.
- **logConvErrors** - indicates if the conversion error should be logged.
- **maxRejects** - the maximum number of parse errors allowed before the application stops.
- **rejectFile** - the pathname of the reject file.

The options **-outdbLocMap** and **-outdbMerchMap** take the following attributes:

- **dbtype** - the type of database to access.
- **host** - the host name of the database to access.
- **port** - the network port number of the database to access.
- **instance** - the instance name of the database to access.
- **user** - the database user name.
- **password** - the database user password.
- **table** - the name of the table to access.

The options **-indb, -indbLocMap**, and **-indbMerchMap** take the following attributes:

- **dbtype** - the type of database to access.
- **host** - the host name of the database to access.
- **port** - the network port number of the database to access.
- **instance** - the instance name of the database to access.
- **user** - the database user name.
- **password** - the database user password.
- **table** - the name of the table to access.
- **query** - the query to run instead of accessing the table.

kde.sh tae

The tae.sh script calls kde.sh tae.

Here is an example of kde.sh tae:

```
kde.sh tae
    -indbdetail dbtype=oracle,host=dbhost,instance=retaildb,user=myuser,
        password=password,table=MD_DETAIL
    -inhistory type=binary,name=/bigdisk/dataset/baseline.out.ds
    -out type=binary,name=/bigdisk/dataset/tae.out.ds
    -indbAdEvent dbtype=oracle,host=dbhost,instance=retaildb,user=myuser,
        password=password,table=IR_PR_PROMOTIONS_VW
    -indbMerchMap dbtype=oracle,host=dbhost,instance=retaildb,user=myuser,
        password=password,table=PR_MERCHANDISE_MAP_VIEW
    -indbLocMap
```

```
dbtype=oracle,host=dbhost,instance=retaildb,user=myuser,password=password,
        table=PR_LOCATION_MAP_VIEW
    -insub type=text-delimited,name=/bigdisk/dataset/item.tae.sub.ds
    -outMerchOutCast type=binary,name=/bigdisk/dataset/merch.outcast.ds
    -outLocOutCast type=binary,name=/bigdisk/dataset/locn.outcast.ds
    -tmpdir /bigdisk/dataset/temp
    -blalloc
    -affinity
    -quality
    -append
    -genxml tae.xml
    -backtrace
    -adeventid 77890
    -algorithm cafe
    -inFocusItems
dbtype=oracle,host=dbhost,instance=retaildb,user=myuser,password=password,
         table=PR_FOCUS_ITEMS
```

The following table lists the complete set of options for the command.

Table 8–3 kde.sh tae Command Options

Option	Option Description	Required/ Optional?
-adeventid	Ad event ID.	Required
-inhistory <in DS attributes>	Historical baseline input dataset.	Required
-ac_id ac *analysis level*	Hierarchy ID to use for affinity correlation analysis.	Optional
-ac_level *column-name*	Name of column to use for ac_level.	Optional
-ad_ind *column-name*	Name of column to use for ad_ind.	Optional
-[no] affinity	Perform affinity.	Optional
-algorithm *algorithm*	The algorithm to use to create the window(s).	Optional
-[no] append	Append output to dataset.	Optional
-[no] backtrace	Display detailed stack backtrace.	Optional
-basewindow *yyyy-MM-dd, yyyy-MM-dd, pmin*	The set of baseline windows. Can be repeated.	Optional
-[no] blalloc	Perform baseline allocation.	Optional
-[no] debug	Turn on debugging.	Optional
-ext_margin_amt *column-name*	Name of column to use for ext_margin_amt.	Optional
-ext_retail_amt *column-name*	Name of column to use for ext_retail_amt.	Optional
-genxml *file-name*	Name of file to write pre-processed generated xml to.	Optional
-[no] groupsort	Use group sort.	Optional
-[no] help	Display message.	Optional
-inAdEvent <in DS attributes>	Ad event input dataset. Takes attributes.	Optional
inFocusItems <in DS attributes>	Focus items input dataset. Takes attributes.	Optional
-inLocMap <in DS attributes>	Location Map input dataset. Takes attributes.	Optional
-inMerchMap <in DS attributes>	Merchandise Map input dataset. Takes attributes.	Optional

Table 8–3 (Cont.) kde.sh tae Command Options

Option	Option Description	Required/ Optional?
-indbAdEvent <in JDBC connection attributes>	Ad event input JDBC connection. Takes attributes.	Optional
indbFocusItems <in JDBC connection attributes>	Focus items input JDBC connection. Takes attributes.	Optional
-indbLocMap <in JDBC connection attributes>	Location Map input JDBC connection. Takes attributes.	Optional
-indbMerchMap <in JDBC connection attributes>	Merchandise Map input JDBC connection. Takes attributes.	Optional
-indbdetail <in JDBC connection attributes>	Detail input JDBC connection. Takes attributes.	Optional
-indetail <in DS attributes>	Detail input dataset. Takes attributes.	Optional
-insub <in DS attributes>	Substitution input dataset. Takes attributes.	Optional
-item_id *analysis level*	Hierarchy ID to use for analysis.	Optional
-loc_client_load_id *column-name*	Name of column to use for loc_client_id.	Optional
-[no] memoryreport	Print memory usage between steps.	Optional
-merch_client_load_id *column-name*	Name of column to use for merch_client_load_ id.	Optional
-minconf *conf*	Minimum confidence (0.0).	Optional
-minfreq *freq*	Minimum frequency (0.0).	Optional
-minlift *lift*	Minimum lift (5.0).	Optional
-minwindows *cnt*	Minimum number of windows (1).	Optional
-nelements *cnt*	Number of elements to read (-1 to all).	Optional
-options *file-name*	Name of property file containing values for command line options.	Optional
-out <out DS attributes>	Metric output dataset. Takes attributes.	Optional
-outLocOutCast <out DS Attributes>	Location unmapped records output dataset. Takes attributes.	Optional
-outMerchOutCast <out DS Attributes>	Merchandise unmapped records output dataset. Takes attributes.	Optional
-outaffinity <out DS attributes>	Affinity output dataset. Takes attributes.	Optional
-outdb <out JDBC connection attributes>	Metric output JDBC connection. Takes attributes.	Optional
-outdbLocOutCast <out JDBC connection attributes>	Location unmapped records output JDBC connection. Takes attributes.	Optional
-outdbMerchOutCast <out JDBC connection attributes>	Merchandise unmapped records output JDBC connection. Takes attributes.	Optional
-[no] quality	Perform quality metrics.	Optional
-runid *identifier*	Run identifier.	Optional
-tmpdir *dir*	Directory for temporary files.	Optional
-txn_date *column-name*	Name of column to use for txn_date.	Optional
-txn_id *column-name*	Name of column to use for txn_id.	Optional
-units_sold *column-name*	Name of column to use for units_sold.	Optional

Option Attributes

Some of the options in Table 8–3, " kde.sh tae Command Options" take attributes.

The options **-inhistory, -inAdEvent, -infocusItems, -inLocMap, -inMerchMap, -indetail,** and -**insub** take the following attributes:

- **type** - the type of input set. Values are *text*, *fixed-text*, and *binary*.
- **schema** - the pathname of the schema file.
- **compression** - the compression method used. Values are *none* and *qzip*.
- **name** - the pathname of the dataset. This is required.
- **nElements** - the number of elements to read. The default is *all*.
- **logConvErrors** - indicates if the conversion error should be logged.
- **maxRejects** - the maximum number of parse errors allowed before the application stops.
- **rejectFile** - the pathname of the reject file.

The options **-indbAdEvent, -indbFocusItems, -indbLocMap, -indbMerchMap,** and **-indbdetails** take the following attributes:

- **dbtype** - the type of database to access.
- **host** - the host name of the database to access.
- **port** - the network port number of the database to access.
- **instance** - the instance name of the database to access.
- **user** - the database user name.
- **password** - the database user password.
- **table** - the name of the table to access.
- **query** - the query to run instead of accessing the table.

The options **-out, -outLocOutCast, -outMerchOutCast,** and **-outaffinity** take the following attributes:

- **type** - the type of output set. Values are *text*, *fixed-text*, and *binary*.
- **schema** - the pathname of the schema file.
- **compression** - the compression method used. Values are *none* and *gzip*.
- **name** - the pathname of the dataset. This is required.
- **timezone** - the timezone used.
- **encoding** - the encoding used.
- **delimiter** - the delimiter used.
- **locale** - the locale used.
- **mode** - the mode used. Values are *append* and *overwrite*.

The options **-outdb, -outdbLocMap** and **-outdbMerchMap** take the following attributes:

- **dbtype** - the type of database to access.
- **host** - the host name of the database to access.
- **port** - the network port number of the database to access.

- **instance** - the instance name of the database to access.
- **user** - the database user name.
- **password** - the database user password.
- **table** - the name of the table to access.

Outcast Files

The KDE applications read and write data to a dataset. A dataset is a set of directories containing binary files in a proprietary format. The Location and Merchandise Outcast are stored as datasets. The content of the outcast file is the transaction record itself, which has an unmatched id.

In order to look at the data, you must export the dataset to a text file, using the following command:

kde.sh export -in merch.outcast.ds -out merch.outcast.txt

This writes all of the data to merch.outcast.txt and writes a description of the data to the schema file, merch.outcast.txt.schema.

The merch.outcast.txt looks something like:

```
10000603|2006-09-20|5773|T0000048205|1|24.5|12.25
10000510|2006-09-20|5773|T0000100045|5|222.5|111.25
10000454|2006-09-20|5169|T0000099958|1|24.5|12.25
10000454|2006-09-20|5169|T0000028447|1|2.5|1.25
10000454|2006-09-20|5169|T0000028121|1|2.5|1.25
10000454|2006-09-20|5169|T0000080813|1|24.5|12.25
10000454|2006-09-20|5169|T0000027734|1|2.5|1.25
10000542|2006-09-20|6493|T0000961305|1|24.5|12.25

….
```

Here is an example of the schema file for Merchandise Outcast:

```
<?xml version="1.0"?>
<analytic name="generated_schema">
    <dataset name="implicit"
             timezone=""
             type="text-delimited"
             encoding="UTF-8"
             delimiter="|"
        >
        <field flow="txn_id" type="integer" />
        <field flow="txn_date" type="date" />
        <field flow="loc_client_load_id" type="string" />
        <field flow="merch_client_load_id" type="string" />
        <field flow="units_sold" type="integer" />
        <field flow="ext_retail_amt" type="double" />
        <field flow="ext_margin_amt" type="double" />
        <field flow="ad_ind" type="integer" />
    </dataset>
</analytic>
```

Each field in the schema file describes a column in the .txt file. The columns in the .txt file are separate by a"|". In the above example, the first column is txn_id, the second is txn_date, and so on.

The Location Outcast dataset is almost identical to the Merchandise Outcast dataset, except that one column, the merch_client_load_id, is replaced with our id, pi_id.

Here is an example of the schema file for Location Outcast:

```
<?xml version="1.0"?>
<analytic name="generated_schema">
    <dataset name="implicit"
             timezone=""
             type="text-delimited"
             encoding="UTF-8"
             delimiter="|"
        >
        <field flow="txn_id" type="integer" />
        <field flow="txn_date" type="date" />
        <field flow="loc_client_load_id" type="string" />
        <field flow="pi_id" type="long" />
        <field flow="units_sold" type="integer" />
        <field flow="ext_retail_amt" type="double" />
        <field flow="ext_margin_amt" type="double" />
        <field flow="ad_ind" type="integer" />
    </dataset>
</analytic>
```