

Oracle® Retail Strategic Store Solutions Implementation Guide

Oracle Retail Strategic Store Solutions to Merchandising
Operations Management Integration

Release 12.0.3IN

November 2008

Copyright © 2007, 2008 Oracle. All rights reserved.

Primary Author: Uma Shankar

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

If the Programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software--Restricted Rights (June 1987). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

Oracle, JD Edwards, PeopleSoft, and Siebel are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

Value-Added Reseller (VAR) Language

Oracle Retail VAR Applications

The following restrictions and provisions only apply to the programs referred to in this section and licensed to you. You acknowledge that the programs may contain third party software (VAR applications) licensed to Oracle. Depending upon your product and its version number, the VAR applications may include:

(i) the software component known as **ACUMATE** developed and licensed by Lucent Technologies Inc. of Murray Hill, New Jersey, to Oracle and imbedded in the Oracle Retail Predictive Application Server - Enterprise Engine, Oracle Retail Category Management, Oracle Retail Item Planning, Oracle Retail Merchandise Financial Planning, Oracle Retail Advanced Inventory Planning and Oracle Retail Demand Forecasting applications.

(ii) the **MicroStrategy** Components developed and licensed by MicroStrategy Services Corporation (MicroStrategy) of McLean, Virginia to Oracle and imbedded in the MicroStrategy for Oracle Retail Data Warehouse and MicroStrategy for Oracle Retail Planning & Optimization applications.

(iii) the **SeeBeyond** component developed and licensed by Sun Microsystems, Inc. (Sun) of Santa Clara, California, to Oracle and imbedded in the Oracle Retail Integration Bus application.

(iv) the **Wavelink** component developed and licensed by Wavelink Corporation (Wavelink) of Kirkland, Washington, to Oracle and imbedded in Oracle Retail Store Inventory Management.

(v) the software component known as **Crystal Enterprise Professional and/or Crystal Reports Professional** licensed by Business Objects Software Limited ("Business Objects") and imbedded in Oracle Retail Store Inventory Management.

(vi) the software component known as **Access Via™** licensed by Access Via of Seattle, Washington, and imbedded in Oracle Retail Signs and Oracle Retail Labels and Tags.

(vii) the software component known as **Adobe Flex™** licensed by Adobe Systems Incorporated of San Jose, California, and imbedded in Oracle Retail Promotion Planning & Optimization application.

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

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

Contents

Preface	xiii
Audience.....	xiii
Related Documents	xiii
Customer Support	xiii
Review Patch Documentation	xiii
Oracle Retail Documentation on the Oracle Technology Network	xiii
Conventions	xiv
 1 Integration Overview	
Data Import from Oracle Retail Merchandising System and Oracle Retail Price Management	1-1
Generic Data Import Flow	1-3
Feed Methods.....	1-4
Data Import Dependencies	1-5
Oracle Retail Price Management to Oracle Retail Strategic Store Solutions Integration Overview	1-5
Oracle Retail Merchandising System to Oracle Retail Strategic Store Solutions Integration Overview	1-8
Tax Import - Oracle Retail Merchandising System to Oracle Retail Strategic Store Solutions	1-9
Process Flow	1-9
Oracle Retail Strategic Store Solutions to Oracle Retail Sales Audit Overview	1-11
Oracle Retail Strategic Store Solutions	1-11
Oracle Retail Strategic Store Solutions RTLog Files.....	1-11
Transport Middleware	1-12
Oracle Retail Sales Audit.....	1-12
Preconditions	1-13
System Flow Description	1-14
Existing Functionality Gaps	1-15
Oracle Retail Price Management.....	1-15
Oracle Retail Merchandising System	1-17
Data Import Field Width Maximums.....	1-19

2 Integration Architecture

Strategic Store Solutions to Oracle Retail Sales Audit Integration Architecture	2-1
RTLog Batch Generator	2-2
Sleep Interval	2-2
Maximum Transactions	2-2
Oracle Retail Sales Audit	2-2
Data Import.....	2-2
Error Handling	2-3
Import Status Logging.....	2-4
The Logic	2-4
Reprocessing a Bundle	2-5
Exception Flow	2-5
Logging	2-7
RTLog Mapping and Translation	2-8

3 Implementation Configuration

Data Import Spring Configurations	3-1
spring.properties	3-3
dimplogger.properties.....	3-4
Archive File Format.....	3-4
Oracle Retail Merchandising System Configuration	3-8
Oracle Retail Price Management Configuration	3-9
Data Requirements – Oracle Retail Strategic Store Solutions to Oracle Retail Sales Audit..	3-10
Tax Import Configuration	3-11
Batch Location	3-11
Execution process	3-11
Steps	3-12
Data Requirements for India L10N Stores	3-13

4 Capacity Planning

5 Customization Notes

Data Import Extension Points and Development	5-1
Import Adapter and Translator.....	5-3
SAXParserGenerator.....	5-3
Manually Editing Generated Code	5-3
Metadata	5-5
ImportControllerIfc.....	5-6
Strategic Store Solutions to Oracle Retail Sales Audit Extension Points and Development ...	5-6
Adding Data Elements to the RTLog Format	5-6
Creating a New Fixed Length Export Record Format	5-7
Exporting a Non-Fixed-Length Record Format.....	5-8
Object Factories.....	5-8
StoreServerConduit.xml.....	5-9
DomainObjectFactory.....	5-10
ExtractorObjectFactory.....	5-10

EntityMappingObjectFactory	5-10
RTLogMappingConfig.xml	5-11
RecordFormatObjectFactory	5-11
Configuration.....	5-12
The Store Server Conduit File	5-12
The Export Format Configuration file	5-12
The Entity Reader Configuration File	5-13
The Mapping Configuration File	5-13
Development and Testing Tools	5-14
Classes	5-14
Executables in the bin Directory	5-15
Extending the RTLog Encryption model	5-16

6 Known Issues and Troubleshooting

DepartmentDefaultTaxGroup.....	6-1
Character Restrictions for UOMs	6-1
POSlog	6-1
Preload Section of ItemImport	6-1
UTF-8.....	6-1
Transaction Level Items.....	6-1
Need To Escape Special Characters In XML File.....	6-2
Geocode Tag Missing For Store	6-2
Missing Encryption Key For Saencrypt.pc.....	6-2
Clearance Pricing.....	6-2
Oracle Retail Price Management Price Promotion endDateTime in Pricing Import XSD.....	6-2
Data Import Failure.....	6-2
Integration with Oracle Retail Sales Audit.....	6-2
Total ID in the RTLog	6-3
Debit Card Problem with Oracle Retail Sales Audit	6-3
Data Import Field Width Maximums	6-3
Price Change Applied Before Start Date	6-3
The Sub-Transaction Type in the RTLog Entry for Closing A Register is Not Correct	6-3
When a Gift Certificate is Issued, the Item Type and Voucher Number Are Not Correct in the RTLog.....	6-3
DiscountReferenceNumber Field Format in the RTLog is Incorrect	6-4
If the Refund Amount for a Deleted Layaway is Zero, A TTEND Record is Not Captured in the RTLog.....	6-4
For a Layaway Complete Transaction, the Voucher Number is Not Included in the TTEND Record	6-4
UnitRetail Amount Does Not Match the OriginalUnitRetail Amount in the TITEM Record..	6-4
ItemType and Voucher_no Fields are Not Correct in the TITEM Record in the RTLog	6-4
The Reason Code is Missing in the THEAD Record in the RTLog.....	6-4
The Vouch_no Field is Empty in the THEAD Record in the RTLog	6-5
Item Import Fails Because an Item Already Exists	6-5
The Cost Field for an Item is Updated from the Import Instead of Using 0.00.....	6-5
Empty Item Classes Lists for DIMP.....	6-5
Discountable Attribute from Oracle Retail Merchandising System.....	6-5

RegistryEligible Field	6-5
Authorized for Sale	6-5
CatchWeight Item in RTLog	6-5
Customer-Specific Pricing in Pricing Data Import.....	6-6
Item-Level Coupon Number is Not Captured in the RTLog	6-6
ReferenceNumber1 Field Does Not Contain Number of RTLogs Sent to Oracle Retail Sales Audit	6-6
Temporary Price Change Loaded Using DIMP Not Going Into Effect.....	6-6
Unexpected Exception Occurs When Trying to Create a Store Group	6-6
Tax Import Troubleshooting	6-6
Employee Information	6-7

A Appendix: Discount Rules – Any or All

B Appendix: XSD Files and Data Element Definition Tables

Employee Import	B-1
Item Import.....	B-7
Merchandise Hierarchy Import	B-31
Pricing Import	B-40
Store Hierarchy Import	B-66
Tax Flat File Specification	B-76

List of Figures

1-1	Integration Overview Including Strategic Store Solutions and Merchandising Operations Management Products	1-2
1-2	Strategic Store Solutions to Oracle Retail Price Management Integration	1-7
1-3	Strategic Store Solutions and Oracle Retail Merchandising System Integration.....	1-9
1-4	Tax Import.....	1-10
1-5	High-Level Model for Oracle Retail Strategic Store Solutions-Oracle Retail Sales Audit Integration	1-13
2-1	Data Import Tables Logical Data Model	2-6
3-1	Adding Files To a Jar	3-6
3-2	Adding Files To A WinZip Archive	3-7
3-3	Tax Assignment successfully completed.....	3-13
6-1	Tax Import Troubleshooting	6-7

List of Tables

1-1	Functionality Gaps for Promotion Data Import.....	1-15
1-2	Functionality Gaps for Price Change Data Import	1-16
1-3	Functionality Gaps for Discount Rule Data Import.....	1-16
1-4	Functionality Gaps for Item Data Import.....	1-17
1-5	Functionality Gaps for Merchandise Hierarchy Data Import	1-18
1-6	Functionality Gaps for Store Hierarchy Data Import.....	1-18
1-7	Affected XML Elements	1-19
2-1	TransactionType (TRAT)	2-8
2-2	ReasonCode (REAC).....	2-9
2-3	OverrideReasonCodes (ORRC).....	2-10
2-4	ReturnReasonCodes (SARR)	2-10
2-5	SADT.....	2-11
2-6	TaxCode (TAXC).....	2-11
2-7	TenderTypes (TENT).....	2-11
2-8	TenderType ID (POS_TENDER_TYPE_HEAD).....	2-12
2-9	CCEM.....	2-13
2-10	Unit of Measure.....	2-13
2-11	Total ID for TOTAL type transactions	2-13
2-12	PRMT	2-14
3-1	Spring Bean IDs Used For Each Of The Pluggable Components.....	3-1
3-2	Additional Spring Bean IDs Used For Each Of The Pluggable Components	3-2
3-3	Oracle Retail Merchandising System Default Values in the Back Office Item Maintenance Screen	3-8
3-4	Oracle Retail Price Management Default Values	3-10
4-1	File Sizes	4-1
4-2	Bundle Size.....	4-2
4-3	Hard Drive Capacity	4-2
4-4	Item Import Data Volumes.....	4-2
5-1	Store Server Conduit File.....	5-9
5-2	EntityMappingObjectFactory Classes.....	5-10
5-3	RecordFormatObjectFactory Classes	5-11
5-4	Store Server Conduit File.....	5-12
5-5	Exportfile Utility Classes	5-15
5-6	bin Directory BAT Files.....	5-15
B-1	Employee Import XSD Element Mapping Table.....	B-1
B-2	Item Import XSD PreloadData Element Mapping Table	B-7
B-3	Item Import XSD Item Element Mapping Table	B-10
B-4	Item Import XSD Oracle Retail Merchandising System Export Files Mapping Table...	B-17
B-5	Merchandise Hierarchy Import XSD PreloadData Element Mapping Table.....	B-31
B-6	Merchandise Hierarchy Import XSD Element Mapping Table.....	B-33
B-7	Merchandise Hierarchy Import XSD Oracle Retail Merchandising System Export Files Mapping Table	B-34
B-8	Pricing Import XSD PriceChange Element Mapping Table	B-40
B-9	Pricing Import XSD Price Promotion Element Mapping Table	B-42
B-10	Pricing Import XSD Discount Rule Element Mapping Table.....	B-46
B-11	Pricing Import XSD Oracle Retail Merchandising System Export Files Mapping Table	B-52
B-12	Store Hierarchy Import XSD Preload Data Mapping Table	B-66
B-13	Store Hierarchy Import XSD Element Mapping Table.....	B-69
B-14	Tax Flat File Specification	B-76

List of Examples

1-1	Sample excerpted from StoreServerConduit.xml.....	1-11
1-2	Sample excerpted from StoreServerConduit.xml.....	1-14
2-1	Sample JMX Configuration.....	2-7
2-2	Message Bean Definition.....	2-7
5-1	SAXParserGenerator utility command prompt.....	5-3
5-2	EmployeeAccessHandler Process DTO Before Children.....	5-4
5-3	EmployeeImportHandler Process DTO During Start.....	5-4
6-1	Employee File XML Schema Definition.....	6-8
B-1	EmployeeImport.xsd	B-4
B-2	ItemImport.xsd.....	B-22
B-3	MerchandiseHierarchyImport.xsd	B-36
B-4	PricingImport.xsd	B-60
B-5	StoreHierarchyImport.xsd	B-71

Preface

Audience

The Implementation Guide is intended for the Oracle Retail Point-of-Service integrators and implementation staff, as well as the retailer's IT personnel.

Related Documents

For more information, see the following documents:

- Oracle Retail Back Office documentation set
- Oracle Retail Central Office documentation set
- Oracle Retail Point-of-Service documentation set

Customer Support

- <https://metalink.oracle.com>

When contacting Customer Support, please provide:

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

Review Patch Documentation

For a base release ("0" release, such as 12.0), Oracle Retail strongly recommends that you read all patch documentation before you begin installation procedures. Patch documentation can contain critical information related to the base release, based on new information and code changes that have been made since the base release.

Oracle Retail Documentation on the Oracle Technology Network

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

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

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

Conventions

The following text conventions are used in this document:

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

Integration Overview

Data Import from Oracle Retail Merchandising System and Oracle Retail Price Management

Seed data such as item, price and tax must be updated on an ongoing basis in the Store database (SDB) as well as Operational Data Store (ODS) to enable daily store operations. Typically the system of truth for such data is an enterprise system, such as Oracle Retail Merchandising System, Oracle Retail Price Management or a third-party product. The frequency and size of the data feeds varies from customer to customer. Imports are scheduled to be picked up by stores on a nightly basis. This interval is adjustable. See "[spring.properties](#)".

Note: DIMP is not the system of record for data correctness. All data coming into the data import module is assumed to be correct. This applies at two levels:

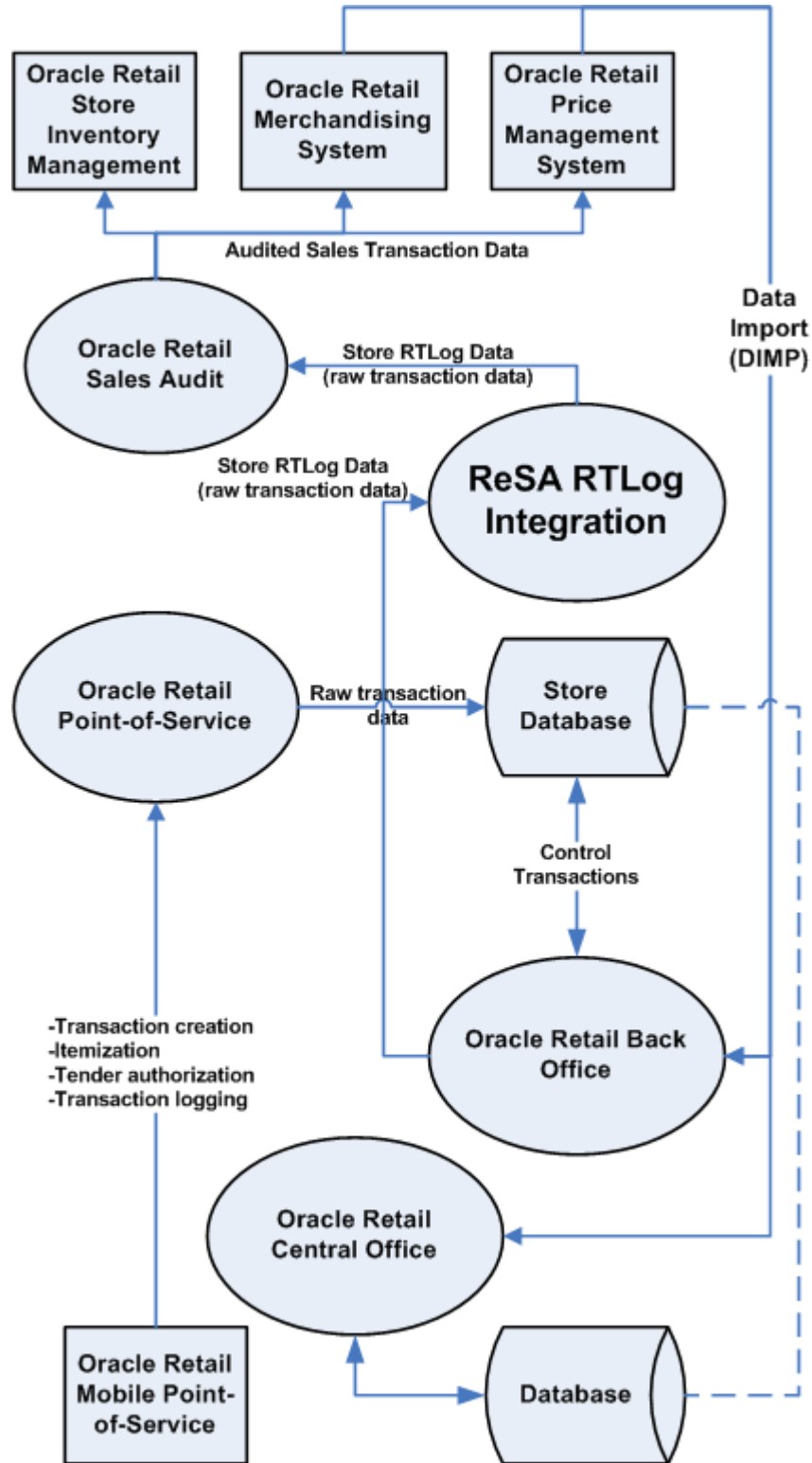
- First, the data must conform to the published XSDs. See [Appendix B, "Appendix: XSD Files and Data Element Definition Tables"](#).
- In addition, the database does not enforce referential integrity on the imported data, so the external system is responsible for not sending data that would create orphaned records in the database.

For example, there is no foreign key constraint enforced between the employee and store entities. A Kill And Fill import of the store hierarchy can result in a new set of stores that does not include a store for some existing employees. The external system that creates this import data must ensure that this type of situation does not occur.

Note: The base DIMP application supports parsing XML files only.

The following is an overview diagram of an integration of Strategic Store Solutions and Merchandising Operations Management products, including a Data Import logical flow:

Figure 1-1 Integration Overview Including Strategic Store Solutions and Merchandising Operations Management Products



Generic Data Import Flow

The following describes the flow of a generic data import:

1. The flow begins with the Quartz Scheduler configured in Spring invoking the ImportIOAdapter of the DIMP Controller module.

An import can also be processed by Central Office. However, this information is for-your-information only. To get new data to a store, the data must be imported by Back Office.
2. The DIMP Controller picks up the import bundle, which is a compressed archive, and invokes the DIMP Translator.
3. The XML files are processed as input streams in order by DIMP translators: one for each import type.
4. The implementation of the ImportTranslatorIfc (as configured by Spring) retrieves an instance of an ImportControllerIfc from Spring and creates a new ImportBatch.
5. The translator begins to parse its document and calls initializeImport onto the controller.
6. The translator sets the batch size based upon its configuration.
7. The translator then loops through the elements in the document, creating a Data Transfer Object (DTO) for each complex element. The entity DTOs are processed one at a time in the order they are placed into the ImportBatch, with all Delete DTOs processing first, all Add DTOs second, then all Update DTOs last.
8. The controller retrieves an instance of the specified Data Access Object (DAO) from Spring based upon the key passed to it and calls initializeImport() on the DAO.
9. The translator then loops through the elements in the document, creating a Data Transfer Object (DTO) as each complex element. The entity DTOs are processed one at a time by placing them into the batch.
10. Each batch is processed as a transaction. Any records in the batch with data errors roll back that transaction. The import proceeds with the next batch.

The default batch size is 1000. See [spring.properties](#) in Chapter 3 for more information.
11. The translator gives the ImportController a signal to process the batch after adding each DTO by calling processBatch().
12. If the batch size has been reached, the controller sends the batch to the DAO to be persisted.
13. The ImportDAOIfc loops through each DTO and delegates its data operation to a subordinate DAO.
14. Once the document parsing is complete, the translator notifies the controller, which processes the batch if there are any DTOs left over.

15. Finally, the controller calls `completeImport()` on the DAO, giving it the opportunity to copy data from temporary to production tables and drop temporary tables in case of a Kill And Fill, or release JDBC resources, and so forth.

Note: If you choose to retain any existing Oracle Retail Back Office or Oracle Retail Point-of-Service item-related functionality that creates or changes data types that are imported from Oracle Retail Merchandising System or any third party merchandising systems, you are responsible for handling and addressing any data overwrites performed by the import process.

Feed Methods

There are three feed methods:

Kill And Fill

Temporary tables are created at the beginning of a file's processing. Batches are written to the temporary tables. If the entire file is processed without error (all batches), the temporary table data replaces the production data and the temporary tables are dropped. If an error occurs, it is logged and the entire file import is aborted.

Note: During the data import of any `PricingImport` that has had its `FillType` set to **Kill And Fill**, all tables that contain `AdvancedPricingRules`, `PricePromotions`, and `PriceChanges` are cleared and refilled with the new data that is imported only. The `PriceLookup` mechanism uses the `PriceChange` tables to calculate the current price of an item. If all the prices are not supplied for existing items during a `PricingImport` Kill And Fill, then the items without prices have values of **zero**.

Full Incremental

Full Incremental is a fill type that performs adds, updates, or deletes expecting that all data attributes for a particular record are included in the file. Any missing attributes are provided default values.

Note: All columns for a row must be present in the import data.

For Full Incremental imports, each import XML data element must include all values. If some values are omitted from the import file, then the Data Import still updates the records in question, but uses default values for the omitted elements or attributes. Usually the default value chosen is **null**, **zero** or **false** unless otherwise specified in the XSD.

Delta Incremental

Delta Incremental is a fill type that produces dynamic update statements that allow for only those data attributes which are included in the file to be updated, leaving existing data attributes intact.

Note: Only those fields being updated are required in the import data.

Data Import Dependencies

The following dependencies dictate the order of the Data Import process:

- Store Hierarchy
- Merchandise Hierarchy
- Item
- Taxation

Note: For India Localization (herein after referred to as L10N), the DIMP has been enhanced to import the additional data related to MRP and Tax information for Items. Tax Assignment Data is not uploaded via DIMP module, it is uploaded via a batch file Calculate_TaxFactor.bat after DIMP has been run.

The implementation team must run the batch file to upload appropriate tax information after every Kill And Fill operation. This ensures that tax information applied to the store database is retained the next time a Kill And Fill operation is conducted.

- Employee depends on Store Hierarchy/Stores
- Merchandise Hierarchy depends on nothing
- Tax depends on Item
- Item depends on Merchandise Hierarchy
- Pricing depends on Item

Oracle Retail Price Management to Oracle Retail Strategic Store Solutions Integration Overview

Oracle Retail Price Management is a strategy-based pricing solution that suggests and assists with pricing decisions, yielding a more predictable and profitable outcome. Oracle Retail Price Management evaluates prices within a broad business context with real-time access to the following:

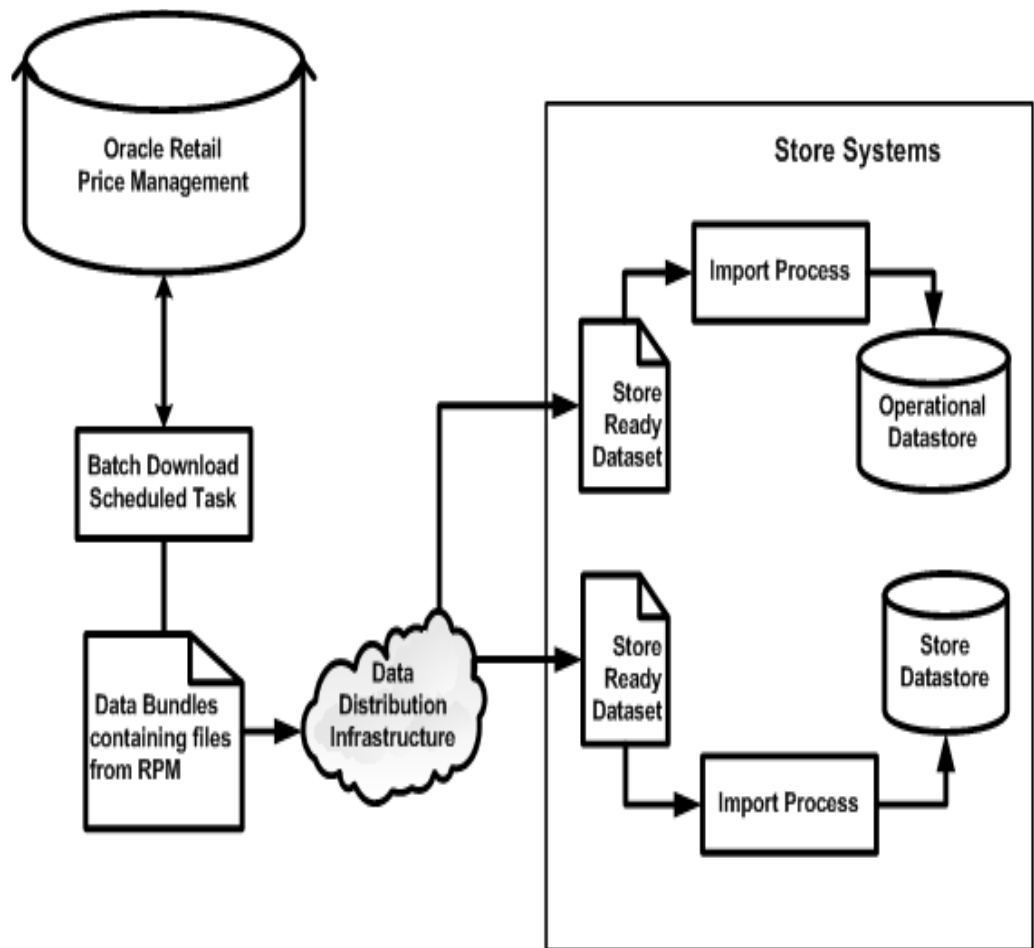
- Competitive and market data
- Projected sales impact
- Margin
- Pricing-based costs
- Current and projected inventory positions
- Markdown budgets

Oracle Retail Price Management provides a well-defined and efficient price change process that allows for aggregated permanent and clearance price change execution. Oracle Retail Price Management enables retailers to automate and streamline pricing strategies across the organization. Oracle Retail Price Management provides decision support to all pricing-focused business information to validate and approve pricing and markdown suggestions.

Note: This integration is one-way only. Oracle Retail Strategic Store Solutions changes are not communicated back up to Oracle Retail Price Management.

The following figure shows a high level overview of the integration.

Figure 1–2 Strategic Store Solutions to Oracle Retail Price Management Integration



Oracle Retail Merchandising System to Oracle Retail Strategic Store Solutions Integration Overview

Oracle Retail Merchandising System provides for core merchandising activities, including inventory replenishment, purchasing, and vendor management, in a global environment, across multiple retail channels. The solution incorporates three functional areas:

- Business foundation management
- Merchandise management
- Merchandise financial tracking

These functional areas enable retailers to streamline their business systems and unify business practices across their organization.

Oracle Retail Merchandising System is the main application for item, item location, merchandise hierarchy, stores, tax, and store (organizational) hierarchy data. This data is necessary for store operations and must be updated in the stores on an ongoing basis. Further, this data, particularly item data, can range in size from small incremental updates to large batch loads. The frequency and size of data feeds varies widely from customer to customer. Tax data import has been explained in the following section.

Note: This integration is one-way only. Oracle Retail Strategic Store Solutions changes are not communicated back up to Oracle Retail Merchandising System.

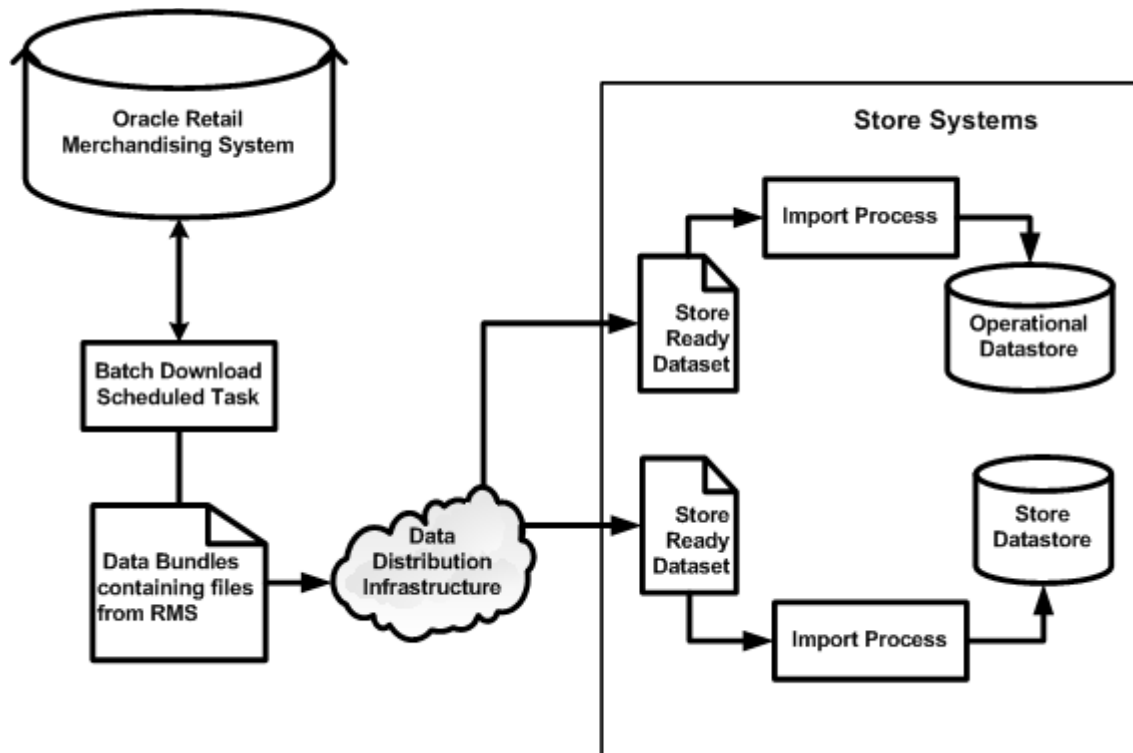
Note: There are some conditions required on data in order to filter out the Oracle Retail Merchandising System data being extracted to the XML files. This is required mainly because Oracle Retail Point-of-Service has these limitations on data types. Some of these conditions are:

- Store ID length is less than or equal to 5.
- Chain value length is less than or equal to 4.
- Item ID length is less than or equal to 14.
- UOM length is less than or equal to 2.
- Diff_1 (ColorCode) length is less than or equal to 20.
- Diff_2 (SizeCode) length is less than or equal to 10.
- Unit retail is less than or equal to 999999.99

For more information, see *Oracle Retail Strategic Store Solutions Relational Integrity Diagrams*.

The following figure shows a high level overview of the integration.

Figure 1–3 Strategic Store Solutions and Oracle Retail Merchandising System Integration



Tax Import - Oracle Retail Merchandising System to Oracle Retail Strategic Store Solutions

In India L10N all the tax data setup is done in RMS which is the master for tax information. The tax information is downloaded to ORPOS as a flat file. The information in the tax flat file would primarily contain tax category, vat code, vat rate, order in which tax would be applied (that is, application order of tax) and on what component to be applied on (that is, selling Retail or Tax). The Selling Retail downloaded from RMS is tax inclusive. Hence the taxable amount (tax exclusive selling retail) and the tax amount (total tax including tax breakup) need to be calculated. Taxes will be applied on either the selling retail/tax value or percentage of the selling retail/tax value only. A new batch Calculate_TaxFactor.bat is developed for India L10N which will calculate the tax factors and taxable factors for each tax category for a given store ID and populates the tax assignment (TX_ASGMT) table. Tax category for the items are pre-populated as part of item data import.

Note: The tax factors are calculated only for those tax assignments which are applied on Selling Retail.

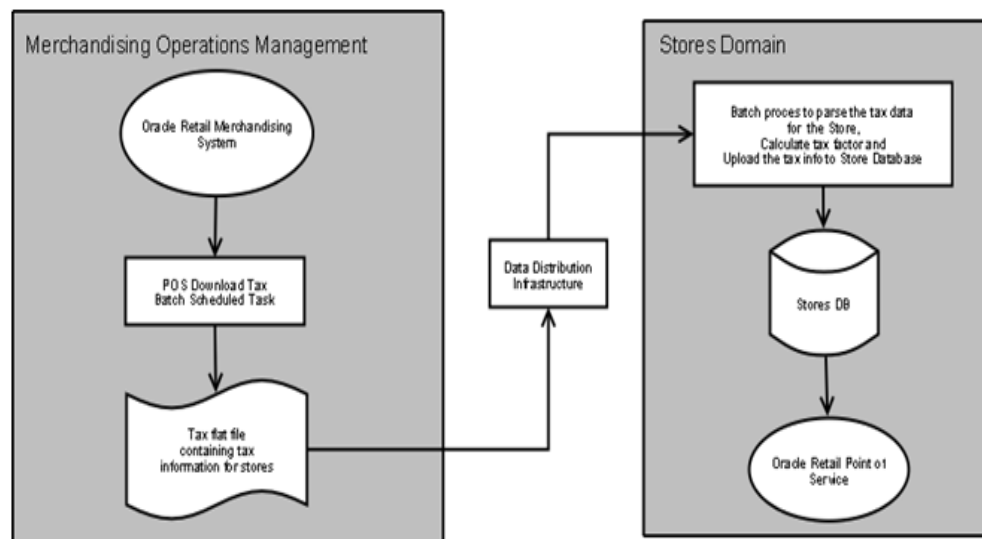
Process Flow

- All the tax assignments for an item's tax category are setup in ORMS (Oracle Retail Merchandising System) for India L10N.

- A batch POSDNLD_TAX program is used to download all the tax assignment information from Oracle Retail Merchandising System to Stores in the form of a flat file. For more information on the batch program, refer to *Oracle Retail Merchandising System Operations Guide - Batch Overviews and Designs - Volume 1 Release 12.0.5IN*.
- A DDI middleware will be used to transfer the tax flat file from RMS to POS.
- A new batch, Calculate_TaxFactor.bat, is developed for India L10N which will calculate the tax for each tax category for a given store ID.
- The Calculate_Taxfactor.bat batch performs the following tasks.
 - Reads the flat file which contains tax assignment provided by RMS for StoreId/TaxCategory level.
 - Loads all the lines of data for the given store ID (In the data file, the store ID is stored in the position of 16th column to 25th column of line starting with value FDETL).
 - The tax data is uploaded using SQL loader into TX_ASGMT table with 0 as default value for column TX_FCT and TXBL_FCT.
 - The PL/SQL tax calculation engine is initialized with the assumption that ID_CTY_TX column of AS_ITM table is populated for all the items.
 - For each tax category, the PL/SQL will calculate the value of tax factor and taxable factor. The factors mentioned above are calculated for single rupee. These values are then stored in TX_ASGMT table in TX_FCT and TXBL_FCT columns.
- During the runtime, POS multiplies the tax factor with the tax inclusive selling retail and calculates the tax amount for an item. The already computed tax factor simplifies and optimizes the tax calculation during an item scan at the POS terminal.

Note: This middleware has to be implemented by the retailer. This is not provided as part of the product.

Figure 1–4 Tax Import



Oracle Retail Strategic Store Solutions to Oracle Retail Sales Audit Overview

The integration of the Oracle Retail Strategic Store Solutions products with the Oracle Retail Sales Audit (ReSA) application involves the following components:

Oracle Retail Strategic Store Solutions

The Oracle Retail Strategic Store Solutions logical component is comprised of Oracle Retail Point-of-Service, Back Office, and Central Office. RTLog data is created from Point-of-Service.

Oracle Retail Strategic Store Solutions RTLog Files

The RTLog file is the communication mechanism for providing data from the Oracle Retail Strategic Store Solutions to Oracle Retail Sales Audit. The RTLog is a transaction log file that is formatted specifically for Oracle Retail Sales Audit. Raw transaction data in the RTLog file is meant to update other Merchandise Operations Management applications, and is populated from Oracle Retail Strategic Store Solutions. The file is written to the physical file system by Oracle Retail Strategic Store Solutions for consumption by the transportation middleware.

Oracle Retail Strategic Store Solutions is responsible for writing the RTLog files to a configurable physical directory on the Store Server.

Note: RTLog files are encrypted. See *Oracle Retail Merchandising System Operations Guide - Batch Overviews and Designs - Volume 1 Release 12.0.5IN*.

The `propname="outputAdapterClassName"` class in the `StoreServerConduit.xml` file controls the writing of the RTLog to a file. This particular implementation encrypts the file with a hard coded key. To use a Key Store, a new implementation of the class must be written to interface with the chosen Key Store. The Key Store must be provided by the implementation team. Additionally, a mechanism for sharing the key used to encrypt the RTLog file must be added, such as embedding a public key in the file itself. See [Extending the RTLog Encryption model](#) in chapter 5 for more information.

Example 1–1 Sample excerpted from `StoreServerConduit.xml`

```
<TECHNICIAN name="RTLogExportDaemonTechnician"
class="RTLogExportDaemonTechnician"
package="com.extendyourstore.domain.manager.rtlog"
export="Y">
  <PROPERTY propname="daemonClassName"
    propvalue="com.extendyourstore.domain.manager.
rtlog.RTLogExportDaemonThread"/>
  <PROPERTY propname="daemonName"
    propvalue="RTLogExportDaemon"/>
  <PROPERTY propname="sleepInterval"
    propvalue="600"/>
  <PROPERTY propname="exportDirectoryName"
    propvalue="POSLog"/>
  <PROPERTY propname="databaseAdapterClassName"
    propvalue="com.extendyourstore.domain.manager.
rtlog.RTLogDatabaseAdapter"/>
```

```
<PROPERTY propname="encryptionAdapterClassName"
propvalue="com.extendyourstore.domain.manager.
rtlog.RTLogEncryptionAdapter" />
<PROPERTY propname="outputAdapterClassName"
propvalue="oracle.retail.stores.exportfile.rtlog.
RTLogEncryptingOutputAdapter" />
```

Transport Middleware

The transport middleware is a component that is responsible for polling the RTLog file produced by the Oracle Retail Strategic Store Solutions. This component has the following responsibilities:

- Polling the physical file system at a specified directory.
- Writing the RTLog file to a location that Oracle Retail Sales Audit expects.
- Cleaning and archiving the RTLog file once Oracle Retail Sales Audit has consumed the RTLog file.
- Error notification if the RTLog file is not able to be extracted successfully from a physical directory.

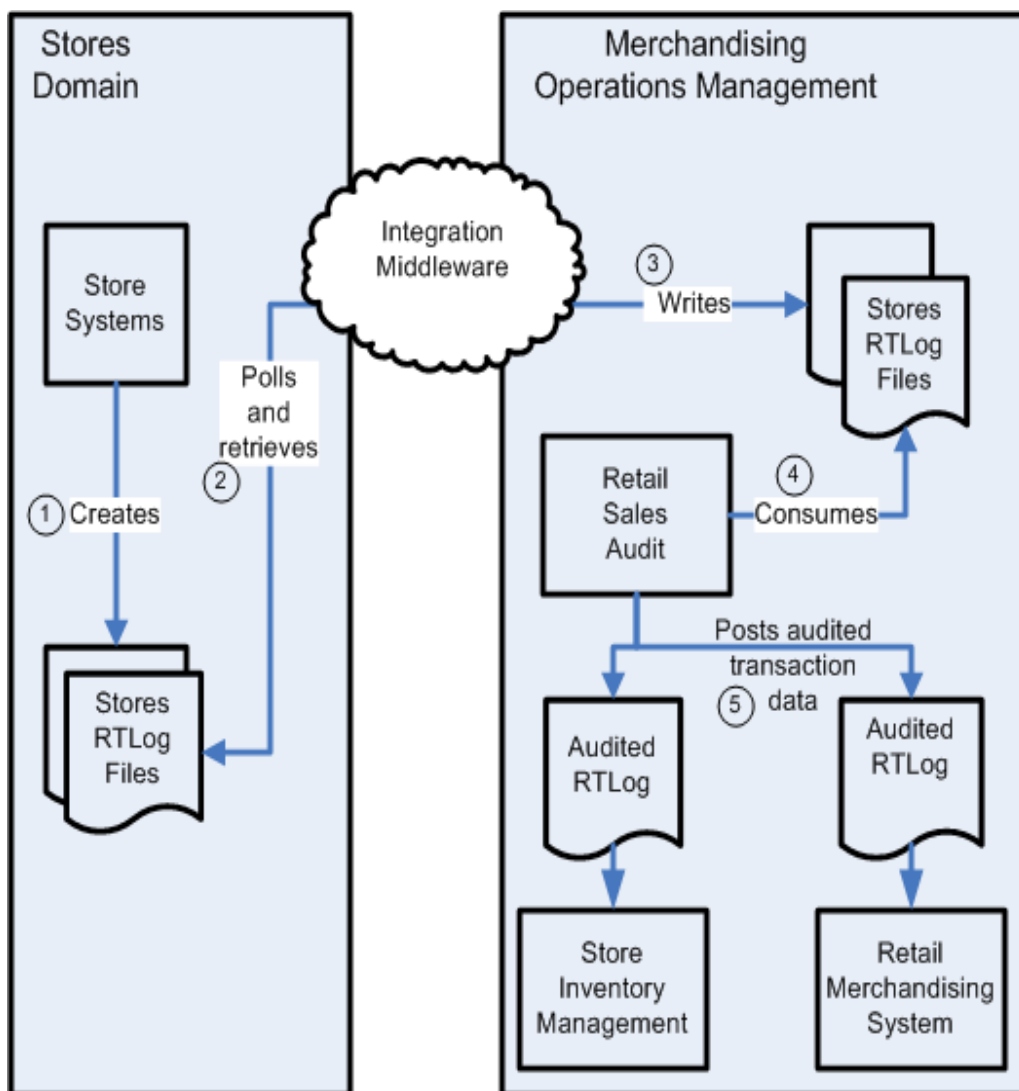
Note: Transport middleware is not provided by Oracle Retail. It is the responsibility of the implementation team to provide the integration middleware of their choice.

Oracle Retail Sales Audit

Oracle Retail Sales Audit is the gateway for transaction data updates to merchandising and inventory systems. The Oracle Retail Sales Audit consumes the RTLog file written to a specific directory by the integration middleware. Oracle Retail Sales Audit also sends audited data files to other Merchandise Operations Management applications for consumption.

The following figure depicts the two domains that are involved when integrating transaction data within the Oracle Retail suite.

Figure 1–5 High-Level Model for Oracle Retail Strategic Store Solutions-Oracle Retail Sales Audit Integration



Preconditions

The following preconditions must be observed for the system flow to function correctly:

1. Transport middleware requires read and write access to the physical file system to which Oracle Retail Strategic Store Solutions writes the RTLog file.
2. Transport middleware requires read and write access to the physical file system from which Oracle Retail Sales Audit reads the RTLog files.
3. Oracle Retail Strategic Store Solutions requires access to a physical file system to produce the RTLog file.

System Flow Description

The Point-of-Service client application generates transaction data and sends the transaction object structure to the Point-of-Service store server. The Point-of-Service store server populates the JDBC statement type and commits the transaction data to the store database. The time increment at which data is sent to Oracle Retail Sales Audit is dictated by the retailer by editing the `propname="sleepInterval"` property in the `StoreServerConduit.xml` file:

Example 1–2 Sample excerpted from StoreServerConduit.xml

```
<TECHNICIAN name="RTLogExportDaemonTechnician"
            class="RTLogExportDaemonTechnician"
            package="com.extendyourstore.domain.manager.rtlog"
            export="Y">
  <PROPERTY propname="daemonClassName"

propvalue="com.extendyourstore.domain.manager.rtlog.RTLogExportDaemonThread" />
  <PROPERTY propname="daemonName"
            propvalue="RTLogExportDaemon" />
  <PROPERTY propname="sleepInterval"
            propvalue="600" />
  <PROPERTY propname="exportDirectoryName"
            propvalue="POSLog" />
```

See [Table 5–1, "Store Server Conduit File"](#) in chapter 5 for more information.

The overall flow shown in [Figure 1–5](#) is summarized in the following sequence:

1. Oracle Retail Strategic Store Solutions creates and encrypts RTLog files.
If the RTLog is not successfully created due to unsupported mappings, the transaction identifier and exceptional condition is logged in detail on the Point-of-Service store server.
2. Transport middleware scans directory that Oracle Retail Strategic Store Solutions writes the RTLog file to and reads in unprocessed RTLog files.
3. Transport middleware moves the RTLog file from the physical directory written to by Oracle Retail Strategic Store Solutions to a physical directory on an enterprise server defined by Oracle Retail Sales Audit.
4. Oracle Retail Sales Audit consumes the RTLog file written to a pre-defined directory by the transport middleware, decrypts, and executes data cleansing operations to produce audited transaction data. See [Oracle Retail Strategic Store Solutions RTLog Files](#), this chapter.
5. Oracle Retail Sales Audit outputs audited RTLog-formatted transaction batch files and places the files into directories accessible by Merchandise Operations Management applications.

Existing Functionality Gaps

For this release, there are certain functionality gaps that exist that are not remedied at this time. This section describes the functional gaps, and the suggested resolution.

Oracle Retail Price Management

[Table 1–1](#) is a list of functionality gaps that exist for the Promotion data import.

Table 1–1 Functionality Gaps for Promotion Data Import

Identified Functionality Gap	Suggested Resolution
Oracle Retail Price Management does not download a start time.	Assume a start time of 00:00:00.
Oracle Retail Price Management does not download an end time.	Assume an end time of 23:59:59.
Oracle Retail Price Management supports a larger field (Change Value - Number) than does Strategic Store Solutions. This field is the amount, either monetary or percent, to be used to change or replace the current selling price for a sale unit of an item. Could result in loss of data in case of a very large discount amount	Gap to remain unchanged for this release.
In Oracle Retail Price Management, all applicable price promotions are applied. In Point-of-Service, if price promotion and discount rule apply to the same item, then the best deal is applied. If price change and discount rule or price promo apply to the same item, then both price change and promo or discount rule are applied.	Oracle Retail Price Management turns off overlapping promotions. This ensures that only one promotion is applied to an item or location at a time.
The Item Number field is larger in Oracle Retail Price Management than Strategic Store Solutions.	Strategic Store Solutions logs an error if the database field is exceeded.
Field for Promotion Price attribute is larger in Oracle Retail Price Management. Multiple promotions can be applied, and the selling price represents the results of each promotion applied in the "Apply Order." One record is downloaded for each promotion applied, and each has the same selling price. The stores system only applies the best deal, and it does so at the time the transaction is rung up. In addition to the multiple promotions, Oracle Retail Price Management can also apply "price guides", which might specify the price ends in .99, for example. These price guides are not included in the download file. The selling price is ignored by Point-of-Service. This results in a possible problem if Point-of-Service does not calculate the same price that Oracle Retail Price Management sends as selling price. This discrepancy can result from rounding, price guides, and so forth.	Oracle Retail Price Management turns off overlapping promotions. This ensures that only one promotion is applied to an item or location at a time.

Table 1–2 is a list of functionality gaps that exist for the Price Change data import.

Table 1–2 Functionality Gaps for Price Change Data Import

Identified Functionality Gap	Suggested Resolution
Oracle Retail Price Management supports a longer field (Selling Retail) and more precision.	Gap to remain unchanged for this release.
Oracle Retail Price Management Item field is longer.	Item ID length remains the same in Strategic Store Solutions and Oracle Retail Price Management. If the item ID is too long in the download file, the record is logged and discarded.
Oracle Retail Price Management does not support description field in download file.	Optional Description field is not populated.

Table 1–3 is a list of functionality gaps that exist for the Discount Rule data import.

Table 1–3 Functionality Gaps for Discount Rule Data Import

Identified Functionality Gap	Suggested Resolution
Oracle Retail Price Management Item field length is longer.	Item ID length remains the same in Strategic Store Solutions and Oracle Retail Price Management. If the item ID is too long in the download file, the record is logged and discarded.
Oracle Retail Price Management field (Threshold Value) is longer and supports more precision.	Field length remains the same in Oracle Retail Price Management and Strategic Store Solutions. If the threshold is a decimal value, it is logged and discarded.
Oracle Retail Price Management Item field length is longer.	Item ID length remains the same in Strategic Store Solutions and Oracle Retail Price Management.
Oracle Retail Price Management supports larger values and more precision than stores. Meaning of value (%, \$, or new price) is defined by Change Type.	Field length remains the same in Oracle Retail Price Management and Strategic Store Solutions.
Oracle Retail Price Management does not download a start time.	Assume a start time of 00:00:00.
Oracle Retail Price Management does not download an end time.	Assume an end time of 23:59:59.
Oracle Retail Price Management does not support threshold or limit.	Assume no threshold
Oracle Retail Price Management does not support the Number Of Times Per Transaction (NbrTimesPerTrans) field.	Assume -1, which means no limit to the number of times the promotion can be applied to a transaction. The NbrTimesPerTrans attribute is in the PricingImport.xsd file.
Oracle Retail Price Management does not support the Accounting Method field.	Assume the discount.
Oracle Retail Price Management does not directly support the Allow Source to Repeat field.	Allow source to repeat.
Oracle Retail Price Management does not directly support the Deal Distribution field.	Assume target only.
Target Quantity field is not supported in Oracle Retail Price Management.	Assume target quantity of 1.

Oracle Retail Merchandising System

Table 1–4 is a list of functionality gaps that exist for the Item import.

Table 1–4 Functionality Gaps for Item Data Import

Strategic Store Solutions Attribute	Identified Functionality Gap	Suggested Resolution
Cost	Cost data is not included in the Point-of-Service download file, but Oracle Retail Merchandising System has this data. However, Point-of-Service does not access item cost data from manufacturer.	Gap to remain unchanged for this release.
Sign/Label	This is not maintained by Oracle Retail Merchandising System.	Gap to remain unchanged for this release.
Manufacturer	Not included in the Point-of-Service download, but Oracle Retail Merchandising System has this data.	This value is null.
Planogram	Not maintained by Oracle Retail Merchandising System. Oracle Retail Merchandising System has a generic attribute that could be used for this purpose.	Gap to remain unchanged for this release.
Serialized	Not maintained by Oracle Retail Merchandising System. Point-of-Service uses this to prompt for serial number during order pickup.	Default to false for Oracle Retail Merchandising System imports.
Restocking Fee	Not maintained by Oracle Retail Merchandising System. Point-of-Service uses this to prompt for a restocking fee during returns.	Default to false for Oracle Retail Merchandising System imports.
Activation Required	Not maintained by Oracle Retail Merchandising System.	No attribute in Oracle Retail Merchandising System. Not used by Point-of-Service.
Registry Eligible	Not maintained by Oracle Retail Merchandising System.	No attribute in Oracle Retail Merchandising System. Not used by Point-of-Service.
Special Order Eligible	Prevents certain items from being placed on a special order. Not maintained by Oracle Retail Merchandising System.	Default to false for Oracle Retail Merchandising System imports.
Employee Discount Eligible	Identifies an item as eligible for an employee discount. Not maintained by Oracle Retail Merchandising System.	Default to true for Oracle Retail Merchandising System imports.
Damage Discount Eligible	Identifies an item as eligible for damage discount. Not maintained by Oracle Retail Merchandising System.	Default to true for Oracle Retail Merchandising System imports.

Table 1–4 Functionality Gaps for Item Data Import

Strategic Store Solutions Attribute	Identified Functionality Gap	Suggested Resolution
Size Entry Required	Not maintained by Oracle Retail Merchandising System. Point-of-Service uses this attribute during a sale or return to prompt for item size.	Default to false for Oracle Retail Merchandising System imports.
Itemizing	Strategic Store Solutions assumes item data is interpreted as local time. File creation has the local Oracle Retail Merchandising System time, but no timezone info.	Assume all Timestamps are relative to GMT.
Localization	Oracle Retail Merchandising System data file does not contain localized data for a store.	Accepts one localized text from Oracle Retail Merchandising System and use as all three: stores, user, customer.

Table 1–5 is a list of functionality gaps that exist for the Merchandise Hierarchy import.

Table 1–5 Functionality Gaps for Merchandise Hierarchy Data Import

Strategic Store Solutions Attribute	Identified Functionality Gap	Suggested Resolution
Merchant ID	Oracle Retail Merchandising System does not specify a merchant ID with any of the merchandise classification records sent with the Merchandise Hierarchy download.	Gap to remain unchanged for this release.

Table 1–6 is a list of functionality gaps that exist for the Store Hierarchy import.

Table 1–6 Functionality Gaps for Store Hierarchy Data Import

Strategic Store Solutions Attributes	Identified Functionality Gap	Suggested Resolution
Store Class	Strategic Store Solutions does not accept class.	Gap to remain unchanged for this release.
Store Class Description	Strategic Store Solutions does not accept class description.	Gap to remain unchanged for this release.
Store Format	Strategic Store Solutions does not accept format as part of the data import.	Gap to remain unchanged for this release.
Format Name	Store does not accept format name as part of the data import.	Gap to remain unchanged for this release.

Data Import Field Width Maximums

Some fields can potentially overflow at the database level because the fields are not specifically limited in length by the Data Import XSDs. The following table lists the XML elements that are affected.

Table 1–7 Affected XML Elements

Import	Elements	Maximum Column Size
Employee Import	Employee > EmployeeFullName	VARCHAR(150)
	Employee > EmployeeLastName	VARCHAR(50)
	Employee > EmployeeFirstName	VARCHAR(50)
	Employee > EmployeeMiddleName	VARCHAR(50)
Item Import	Item > RetailStoreItem > POSIdentity @SupplierID	VARCHAR(20)
	PreloadData > Color@Code	VARCHAR(20)
	Item@Color	VARCHAR(20)
	PreloadData > Size@Code	VARCHAR(10)
	Item@Size	VARCHAR(10)
Merchandise Hierarchy Import	PreloadData > MerchandiseGroup > Description	VARCHAR(250)
	PreloadData > POSDepartment > POSDepartmentID	VARCHAR(14)
	PreloadData > POSDepartment > ParentPOSDepartmentID	VARCHAR(14)
	HierarchyList > Hierarchy@Name	VARCHAR(14)
	HierarchyList > Hierarchy > LevelList > Level@Name	VARCHAR(120)
	HierarchyList > Hierarchy > NodeList > Node@ParentNodeID	VARCHAR(14)
	HierarchyList > Hierarchy > NodeList > Node@ID	VARCHAR(14)

Table 1–7 (Cont.) Affected XML Elements

Import	Elements	Maximum Column Size
Pricing Import	PricingImport > PriceChange @ID	VARCHAR(20)
	PricingImport > PriceChange > Item @ID	VARCHAR(14)
	PricingImport > PriceChange > Item @TemplateType	VARCHAR(8)
	PricingImport > PriceChange @TemplateType	VARCHAR(8)
	PricingImport > PricePromotion @ID	VARCHAR(20)
	PricingImport > PricePromotion @TemplateType	VARCHAR(8)
	PricingImport > PricePromotion @TemplateType	VARCHAR(8)
	DiscountRule > Sources > Source @ID	VARCHAR(14)
	DiscountRule > Targets > Target @ID	VARCHAR(14)
	DiscountRule > Sources > Source @ID	VARCHAR(14)
	DiscountRule > Sources > Source @ID	VARCHAR(10)

Table 1–7 (Cont.) Affected XML Elements

Import	Elements	Maximum Column Size
Store Hierarchy Import	PreloadData > StoreRegion > RegionID	VARCHAR(14)
	PreloadData > StoreRegion > RegionName	VARCHAR(120)
	PreloadData > StoreDistrict > DistrictID	VARCHAR(14)
	PreloadData > StoreDistrict > RegionID	VARCHAR(14)
	PreloadData > RetailStore > GeoCode	VARCHAR(10)
	PreloadData > StoreDistrict > DistrictName	VARCHAR(120)
	PreloadData > RetailStore > LocationName	VARCHAR(150)
	PreloadData > RetailStore > DistrictID	VARCHAR(14)
	PreloadData > RetailStore > RegionID	VARCHAR(14)
	PreloadData > RetailStore > GeoCode	VARCHAR(10)
	PreloadData > RetailStore > Address > AddressLine1	VARCHAR(30)
	PreloadData > RetailStore > Address > AddressLine2	VARCHAR(30)
	PreloadData > RetailStore > Address > AddressLine3	VARCHAR(30)
	PreloadData > RetailStore > Address > City	VARCHAR(30)
	PreloadData > RetailStore > Address > State	VARCHAR(30)
	PreloadData > RetailStore > Address > PostalCode	VARCHAR(30)
	PreloadData > RetailStore > Address > Territory	VARCHAR(30)
	PreloadData > RetailStore > Address > Country	VARCHAR(30)
	PreloadData > RetailStore > Address > TelephoneCountryCode	VARCHAR(30)
	PreloadData > RetailStore > Address > TelephoneAreaCode	VARCHAR(30)
	PreloadData > RetailStore > Address > TelephoneLocalNumber	VARCHAR(30)
	HierarchyList > Hierarchy@Name	VARCHAR(120)
	HierarchyList > Hierarchy > LevelList > Level@Name	VARCHAR(120)
	HierarchyList > Hierarchy > NodeList > Node@Name	VARCHAR(120)
	HierarchyList > Hierarchy > NodeList > Node@Descriptor	VARCHAR(250)

Table 1–7 (Cont.) Affected XML Elements

Import	Elements	Maximum Column Size
Tax Import	GEOCode > GeoCodeID	VARCHAR(10)
	GEOCode > TaxJurisdictionName	VARCHAR(50)
	GEOTaxJurisdiction > GeoCodeID	VARCHAR(10)
	TaxAuthority > TaxAuthorityName	VARCHAR(40)
	TaxAuthority > GeoCodeID	VARCHAR(10)
	TaxableGroup > TaxGroupName	VARCHAR(120)
	TaxableGroup > TaxGroupDescription	VARCHAR(250)
	TaxAuthority > AddressLine	VARCHAR(30)
	TaxAuthority > City	VARCHAR(30)
	TaxAuthority > State	VARCHAR(30)
	TaxAuthority > PostalCode	VARCHAR(30)
	TaxAuthority > CountryCode	VARCHAR(30)
	TaxGroupRule > TaxTypeName	VARCHAR(30)
	TaxGroupRule > TaxRuleName	VARCHAR(40)
	TaxGroupRule > TaxRuleDescription	VARCHAR(250)

Integration Architecture

Strategic Store Solutions to Oracle Retail Sales Audit Integration Architecture

The Point-of-Service terminal is the platform that the Point-of-Service client application resides on. The cashier and the store manager interact with the Point-of-Service client application, which generates transaction data. The Point-of-Service client application sends a serialized object structure representing the sales transaction to the Point-of-Service store server residing on the In-Store-Processor (ISP). The ISP is responsible for logging the raw transaction data to the store database.

The major components of the Strategic Store Solutions to Oracle Retail Sales Audit integration are:

- **RTLog Export Daemon Technician**

Processes configuration settings from the Store Sever Conduit XML file; settings include sleep interval, maximum number of transactions per batch, export directory name, object factory class names, and export configuration files names.

Starts the RTLog Export Daemon Thread.

- **RTLog Export Daemon Thread**

Starts the export process on a periodic basis based on the configured sleep interval. Calls the RTLog Batch Generator.

- **RTLog Batch Generator**

Creates a list of transactions ready for export and calls the Export File Generator.

- **Export File Generator**

Reads the transactions in the list and formats the export data based on the export configuration files.

In this integration, the Point-of-Service store server also maps the transaction object structure to RTLog format and places the RTLog-formatted transaction into a file. The individual components that comprise the RTLog generation are described in the following subsections.

RTLog Batch Generator

The RTLog Batch Generator is a Java class that reads transactions from the store database and creates a physical RTLog file. The file format follows the standards outlined in *Oracle Retail Merchandising System Operations Guide - Batch Overviews and Designs - Volume 1 Release 12.0.5IN*.

The RTLog Batch Generator consumes a configuration file that has the settings outlined in the following sections.

Sleep Interval

The RTLog batch generator runs in a daemon mode, which periodically outputs RTLog files created by pulling transactions from the database. In this configuration, Oracle Retail Sales Audit processes one or more RTLog files from any given store.

The default sleep interval value is 600 seconds. This value can be changed in the `StoreServerConduit.xml` file. See [Table 5-4, "Store Server Conduit File"](#) for more information

Maximum Transactions

The Maximum Transactions setting puts a cap on the number of RTLog transactions read from the store database during a processing cycle. If the number of transaction available is less than the maximum transactions setting, the RTLog Batch Generator reads the number of transactions available.

If Maximum Transactions is set to **-1**, then there is no limit to the number of RTLog transactions.

Oracle Retail Sales Audit

The Oracle Retail Sales Audit system is responsible for sales audit functionality at the store and at the corporate level. Store operations make use of Oracle Retail Sales Audit's functionality to determine over/short situations in stores, and make the necessary adjustments to raw transaction data in order to ensure integrity of data being sent to backend merchandising and store inventory systems.

Oracle Retail Sales Audit consumes unaudited transaction data in RTLog batch format. It then subjects the transaction data to numerous checks, and indicates exceptional conditions leading to out-of-balance situations. The Oracle Retail Sales Audit system outputs cleansed or audited RTLog data to be consumed by Oracle Retail Merchandising System (RMS), Oracle Retail Price Management (RPM), and Oracle Retail Store Inventory Management (SIM).

Data Import

The Data Import (DIMP) utility is the application that enables the import of data from both Oracle Retail Merchandising System and Oracle Retail Price Management to Point-of-Service.

Note: When discussing Data Import, functionality applies to both Oracle Retail Merchandising System and Oracle Retail Price Management.

The Data Import (DIMP) subsystem and components are designed to enable external systems to send large volumes of data to the Oracle Retail Strategic Store Solutions applications. The primary intent of this functionality is to allow for initial data seeding or routine data loading (and optional purging) to occur for such types of data as:

- Merchandise Hierarchy
- Store Hierarchy
- Employee
- Item
- Pricing

Note: For ItemImport.xml, refer to the xsd. Some attributes are labeled **required**. All attributes listed as required in the xsd must be included in the ItemImport.xml file. See "[Archive File Format](#)" in Chapter 3 for more information about ItemImport.xml.

Note: When an item is imported without a POSDepartmentID, that particular item not associated with a POSDepartment. When the item is viewed in Back Office, the POSDepartment list defaults its selection to the first department in the list.

Note: Employee information is not provided by Oracle Retail Merchandising System or Oracle Retail Price Management. The Employee information comes from third-party systems. Tax information for India L10N comes from by Oracle Retail Merchandising System and is uploaded to Stores via a batch file.

For more information, see "Employee Information" in Chapter 6.

Error Handling

Strategic Store Solutions applications are not the system of record for data correctness. Error handling is limited to logging errors during the import and performing a retry in certain cases. Because the data imports can be interdependent, a failure in one file import results in an abort of the import of the rest of the files in the order.

There were no changes made to the base data model to support the new data import subsystem. However, a few tables have been added to take care of data import error handling and to support any recovery or retry mechanism that might be put in place in the future (that may be custom developed).

For the current implementation, all Kill And Fill imports are applied into temporary tables. Once the import of the complete bundle is successful, the data is written onto the main tables. If any data operation fails, the entire file import is aborted. A FAILURE status message is logged for each of those files.

Incremental file imports continue even if a data operation fails. In that case, the error is logged and it is the customer's responsibility to decide how to handle the failed operations.

The act of aborting the import is configurable and can be changed based on implementation requirements. The class `ImportErrorHandler` mapped to the

Spring key persistence_ImportErrorHandler in the Spring context file PersistenceContext.xml can be configured to any other custom implementation of an ImportErrorHandler.

Import Status Logging

- In case of failure in opening the bundle or reading a file in the bundle, the status in the tables is MA_STS_BNDL_IMP - FAILED.

No other status is logged in any other table.

- In case of failure in parsing a file, the statuses are:
 - MA_STS_BNDL_IMP - PROCESSED
 - MA_STS_FL_IMP - FAILED for that file and all other files that are dependent on that file.
 - MA_FL_IMP_FLRS - Failure exception details of the file.
- In case of failure while persisting a batch:
 - If Kill and Fill then:

MA_STS_BNDL_IMP - PROCESSED

MA_STS_FL_IMP - FAILED for that file and all other files that are dependent on that file.

MA_FL_IMP_FLRS - Failure exception details of the file that has failed.

- If Full Incremental or Delta Incremental then:

MA_STS_BNDL_IMP - PROCESSED

MA_STS_FL_IMP - PARTIALLY PROCESSED for that file only.

MA_FL_IMP_FLRS - Failure exception details of the files that have failed.

The Logic

MA_STS_BNDL_IMP

This is the Bundle Import Status, which has the processing status at the bundle level. In a case where an input/output error occurs, such as unable to open the bundle or read a file from the bundle, the status is logged as FAILED. In all other cases where there is no input/output error, the status is PROCESSED. This is because a bundle can contain more than one file, and it is, from a performance standpoint, degenerative to keep track of how many files there are in the bundle and how many of them have succeeded and how many have failed. Therefore, unless an input/output error is encountered, the status PROCESSED is logged into the table.

MA_STS_FL_IMP

File Import Status maintains the processing status of each file in a bundle. The status FAILED for a file indicates that there is a parsing exception, or there is a failure while persisting a Kill And Fill file (as complete processing is aborted in case of Kill And Fill). If a failure is logged in this table for a file, then all other files in the bundle that are dependent on the failed file also have a FAILED status.

The status PARTIALLY PROCESSED for a Full Incremental or Delta Incremental import indicates there is a failure in persisting a batch. This status is irrespective of the number of records in the file. In an incremental type of import, a batch of records with no exceptions is persisted to the database and committed. Therefore, to note a FAILED status we must know how many records there are in the file, how many batches do these records form and the processing status of each of the batch. Performance wise this is not advisable.

Also, if a bundle is re-processed, a FAILED status on an incremental file causes the file to be processed again, generating more exceptions.

MA_FL_IMP_FLRS

Any failures encountered are logged in this table.

Reprocessing a Bundle

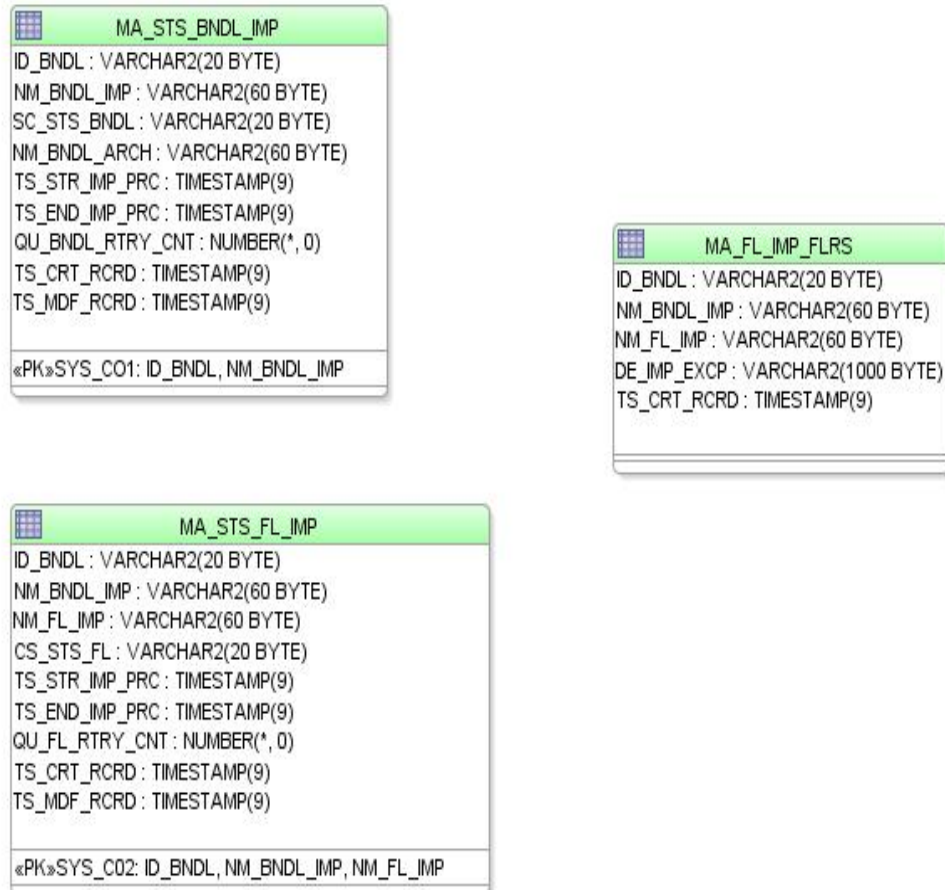
This facility is provided to reprocess any file that failed, that is, has a FAILED status in MA_STS_FL_IMP. No change is needed in the bundle to process a file again. If the same bundle is reprocessed, all the files with a status FAILED in MA_STS_FL_IMP are reprocessed. Therefore, if an incremental file has already crossed the point of parsing, (an exception while persisting) then the status for that file must never be logged as FAILED, as some of the batches might have been persisted and reprocessing the file generates more errors.

Exception Flow

- If there is a failure in any insert operation for a file of the Kill And Fill variety, the exception is logged and the complete file is aborted. Import of any subsequent file in the sequence that depends upon the failed/aborted file is also aborted. This is done to ensure that partial data inserts from the file are not performed, compromising the integrity of the data in the database. Import of files that do not depend on this particular file is not impacted.
- If an operation (insert, update, delete) fails during the processing of an incremental file, delta or full, the current batch is aborted and subsequent batches are processed. The errors are logged for the failed batch and processing continues, starting with the next batch of the data in the file.

The following figure shows the logical data model for the tables being used in error handling in Data import.

Figure 2–1 Data Import Tables Logical Data Model



The archive file status is logged as CONSISTENT or INCONSISTENT in the table ImportBundleStatus, with the BundleID of the archive.

If an exception is encountered during the import of a file, the record where the problem is encountered is logged in the table ImportRecordStatus.

The exception is then sent up to the Data Import Controller where a FAILED status is logged on to the table ImportFileStatus. If the import has been successful for a file, a status of SUCCESS is inserted in the table.

Instrumentation for application monitoring can be provided by exposing beans to JMX through Spring, which orchestrates the process of creating JMX management interfaces for beans, and removes the need to compile them to the JMX API.

The following example must be configured in the Spring PersistenceContext.xml file.

Example 2-1 Sample JMX Configuration

```
<bean id="mbeanServer"
class="org.springframework.jmx.support.MBeanServerFactoryBean"/>

<bean id="exporter" class="org.springframework.jmx.export.MBeanExporter">
  <property name="beans">
    <map>
      <entry key="bean:name=EmployeeImportDAOKey"
value-ref="EmployeeImportDAO"/>
    </map>
  </property>
  <property name="server" ref="mbeanServer"/>
</bean>

<bean id="EmployeeImportDAO"
class="com._
360commerce.commerceservices.employee.importdata.dao.EmployeeImportDAO"/>
```

Logging

At various points in the import process, exceptions such as SQLException and SAXException might be generated. They are generally rethrown as ImportExceptions and passed up the chain to the DIMP Controller, as well as logged for error tracking and resolution.

DIMP introduces a new Spring-based logging object to provide message consistency and allow retailer customization of messages. The underlying logging uses Apache Commons logging as the interface, and Log4j for the logging implementation. A MessageLogger is retrieved from the Spring service context. The logger gets message templates from a property file. Customers can define the layout of these messages to suit their needs, using the following format, where {x} is a placeholder for input data from the calling program:

Message from {0} with {1} information.

The Spring bean ID used for the pluggable message logger component is shown in [Table 3-1, "Spring Bean IDs Used For Each Of The Pluggable Components"](#). The mapping is shown below.

Example 2-2 Message Bean Definition

```
<bean id="service_MessageBuilder" class="com._
360commerce.commerceservices.importdata.MessageBuilder" singleton="true"
lazy-init="true">
  <property name="prefix"><value>${dimp.prefix}</value></property>
  <property name="texts">
    <list>
      <value>${dimp.text1}</value>
      <value>${dimp.text2}</value>
      <value>${dimp.text3}</value>
    </list>
  </property>
</bean>
```

RTLog Mapping and Translation

The following tables identify the changes to the RTLog codemapping that enables Oracle Retail Sales Audit to consume RTLogs generated by Strategic Store Solutions applications. This information is reference material as needed.

For more RTLog information, see the *Oracle® Retail Merchandising System Operations Guide - Batch Overviews and Designs - Volume 1 Release 12.0.5N*.

Note: New Oracle Retail Sales Audit code is highlighted with *italics*.
New Point-of-Service code is highlighted in **bold**.

Table 2–1 TransactionType (TRAT)

TransactionType	TRAT (Static)	TRAS (Static) Sub-Transaction Type
(1) Sale	SALE	<i>SALE</i>
(2) Return	RETURN	<i>RETURN</i>
(3) Void	PVOID	<i>VOID</i>
(4) NoSale	NOSALE	<i>NOSALE</i>
(1) Sale where an even exchange is made	EEXCH	<i>EXCH</i>
(6) OpenStore	OPEN	<i>OSTORE</i>
(7) CloseStore	DCLOSE	<i>DSTORE</i>
(8) OpenRegister	OPEN	<i>OREG</i>
(9) CloseRegister	CLOSE	<i>CRGRC</i>
(10) OpenTill	OPEN	<i>OTILL</i>
(11) CloseTill	CLOSE	<i>CTILL</i>
	TOTAL	<i>CTILLT</i>
(12) LoanTill	LOAN	<i>LOTILL</i>
(13) PickupTill	PULL	<i>PUTILL</i>
(14) SuspendTill	NOSALE	<i>STILL</i>
(15) ResumeTill	NOSALE	<i>RTILL</i>
(16) PayinTill	PAIDIN	<i>PITILL</i>
(17) PayoutTill	PAIDOU	<i>POTILL</i>
(18) HousePayment	PAIDIN	<i>HOUSE</i>
(19) LayawayInitiate	PAIDIN	<i>LAYINT</i>
(20) LayawayComplete	SALE	<i>LAYCMP</i>
(21) LayawayPayment	PAIDIN	<i>LAYPAY</i>
(22) LayawayDelete	PAIDOU	<i>LAYDEL</i>
(23) OrderInitiate	PAIDIN	<i>ORDINT</i>
(24) OrderComplete	SALE	<i>ORDCMP</i>
(25) OrderCancel	PAIDOU	<i>ORDCAN</i>
(26) OrderPartial	SALE	<i>ORDPAR</i>

Table 2–1 TransactionType (TRAT)

TransactionType	TRAT (Static)	TRAS (Static) Sub-Transaction Type
(27) BankDepositStore	NOSALE	<i>BANK</i>
(35) Instant Credit Enrollment	NOSALE	<i>INSCRE</i>
(36) Redeem	RETURN	<i>REDEEM</i>
(37) Enter Training Mode	NOSALE	<i>NTRAIN</i>
(38) Exit Training Mode	NOSALE	<i>XTRAIN</i>
(40) Payroll Payout	PAIDOU	<i>PAYOUT</i>
(41) Enter Transaction Reentry	NOSALE	<i>NTRENT</i>
(42) Exit Transaction Reentry	NOSALE	<i>XTRENT</i>
Any transaction where Transaction.TransactionStatusCode = (3) Canceled	VOID	<i>CANCEL</i>
Any transaction where Transaction.TrainingMode= 'ON	NOSALE	<i>TRAIN</i>
Any transaction where Transaction.TransactionStatusCode = (4) Suspend Transaction	NOSALE	<i>SUSPND</i>

Note: (4) Suspend Transactions get sent in the RTLog. Subsequent resume or cancel actions simply change the transaction status code to (6) Resume Transaction or (7) Canceled Suspended Transaction. A new transaction is not created, hence no subsequent RTLog record, except if the Suspended Transaction is Resumed then SOLD, upon which a SALE transaction is created.

Table 2–2 ReasonCode (REAC)

Reason entered by cashier for some transaction types. Required for Paid In and Paid out transaction types, but can also be used for voids, returns, and so forth.	REAC	Description
Till.TillPayInReasonCodes (53) BadCheckPayment	NSF	NSF Check Payment
Till.TillPayInReasonCodes(54) VendingMachineRevenue	<i>TPIVMR</i>	<i>TillPayIn VendingMachineRevenue</i>
Till.TillPayInReasonCodes(55) Miscellaneous	<i>TPIMSC</i>	<i>TillPayIn Miscellaneous</i>
Till.TillPayrollPayOutReasonCodes (1) PayrollAdvance	PAYRL	Payroll Payout
Till.TillPayrollPayOutReasonCodes (2) FinalPay	PAYRL	Payroll Payout
Till.TillPayOutReasonCodes (56) Postage	<i>TPOP</i>	<i>TillPayOut Postage</i>
Till.TillPayOutReasonCodes (57) Supplies	<i>TPOS</i>	<i>TillPayOut Supplies</i>
Till.TillPayOutReasonCodes (58) Entertainment	<i>TPOE</i>	<i>TillPayOut Entertainment</i>
Sale.NoSaleReasonCodes(1) CustomerChange	NSCC	<i>NoSale CustomerChange</i>

Table 2–2 ReasonCode (REAC)

Reason entered by cashier for some transaction types. Required for Paid In and Paid out transaction types, but can also be used for voids, returns, and so forth.		
	REAC	Description
Sale.NoSaleReasonCodes(2) ChangeForRegister	NSCFR	NoSale ChangeForRegister
Sale.PostVoidReasonCodes(1) IncorrectPrice	PVIP	PostVoid IncorrectPrice
Sale.PostVoidReasonCodes(2) DiscountIncorrect	PVDI	PostVoid DiscountIncorrect
Sale.PostVoidReasonCodes(3) CustomerChangedMind	PVCCM	PostVoid CustomerChangedMind
Sale.PostVoidReasonCodes(4) AssociateError	PVAE	PostVoid AssociateError
Sale.PostVoidReasonCodes(5) OtherFormPayment	PVOFP	PostVoid OtherFormPayment
Sale.PostVoidReasonCodes(6) Other	PVO	PostVoid Other
Where transaction type = (18) House Payment	HOUSE	House Payment
Where transaction type = (19) Layaway Initiate	LAYINT	Layaway Initiate
Where transaction type = (21) Layaway Payment	LAYPAY	Layaway Payment
Where transaction type = (23) Order Initiate	ORDINT	Order Initiate
Where transaction type = (22) Layaway Delete	LAYDEL	Layaway Delete
Where transaction type = (25) Order Cancel	ORDCAN	Order Cancel

Table 2–3 OverrideReasonCodes (ORRC)

Reason an item price is overridden at the Point-of-Service.		
	ORRC	Dynamic
(3) Defective	D	Damaged Goods
(5) SignageError	S	Incorrect Signage
(2) CompetitionPrice	CP	CompetitionPrice
(1) AdPrice	AP	AdPrice
(4) ManagersSpecial	MS	ManagersSpecial

Table 2–4 ReturnReasonCodes (SARR)

The reason an item is returned.		
	SARR	Dynamic
(33) Defective	01	Damaged
(33) Defective	02	Defective

Table 2–4 ReturnReasonCodes (SARR)

The reason an item is returned.	SARR	Dynamic
(11) WrongColor	06	Color Not As Shown
(45) CustomerChangedMind	19	CustomerChangedMind
(55) PriceAdjustment	20	PriceAdjustment

Table 2–5 SADT

The type of discount within a promotion.	SADT	Dynamic
(2402,2006,2303,2105) Saturday Morning Special	SATSPL	Saturday Morning Special
(2410,2014,2311,2113) Senior Citizen	SENCIT	Senior Citizen
(2428,2022,2329,2121) Competition Special	CMPSPL	Competition Special
(2436,2030,2337,2139) Store Coupon	SCOUP	Store Coupon
Employee Discount (Generated when the Employee discount button is pushed in POS. The EmployeeID receiving the discount is recorded in the SaleReturnPriceModifier table.)	EMPDSC	Employee Discount

Table 2–6 TaxCode (TAXC)

Tax code to represent whether it is an item-level tax or transaction-level tax.	TAXC	Dynamic
TGTAX	TGTAX	Aggregate total of tax breakup including VAT.
IGTAX	IGTAX	Item level tax breakup including VAT.

Table 2–7 TenderTypes (TENT)

High-level grouping of tender types.	TENT	Static
CASH Cash	CASH	Cash
CRDT Credit Card	CCARD	Credit Card
CHCK Check	CHECK	Personal Check
ECHK E-Check	CHECK	Personal Check
TRAV Travelers Check	CHECK	Personal Check
MBCK Mail Bank Check	CHECK	Personal Check
QPON Manufacturers Coupon	COUPON	Coupon
DBIT Debit Card	DCARD	Debit Card
MNYO Money Order	MORDER	Money Order
GCRD Gift Card	VOUCH	Voucher (gift cert. or credit)
GICT Gift Certificate	VOUCH	Voucher (gift cert. or credit)
STCR Store Credit	VOUCH	Voucher (gift cert. or credit)
MACT Mall Certificate	VOUCH	Voucher (gift cert. or credit)
PRCH Purchase Order	VOUCH	Voucher (gift cert. or credit)
VOUCH Voucher	VOUCH	Voucher (gift cert. or credit)

Table 2–8 TenderType ID (POS_TENDER_TYPE_HEAD)

Tender Type ID. Low level grouping of tender types.	POS_TENDER_TYPE_HEAD	Notes
CASH Cash	1000 CASH Cash - primary currency	
CHCK Check	2000 CHECK Personal Check	
TRAV Travelers Check	2020 CHECK Traveler Check	
QPON Manufacturers Coupon	5000 COUPON Manufacturers Coupons	
DBIT Debit Card	8000 DCARD Debit Card	
MNYO Money Order	6000 MORDER Money Orders	
GICT Gift Certificate	4030 VOUCH Gift Certificate	
GCRD Gift Card	4040 VOUCH Gift Card	
STCR Store Credit	4050 VOUCH Store Credit	
MACT Mall Certificate	4060 VOUCH Mall Certificate	
PRCH Purchase Order	4070 VOUCH Purchase Order	
VOUCH Voucher	4080 VOUCH PrePaid	Use for Orders and Layaways
ECHK E-Check	2030 CHECK E-Check	
MBCK Mail Bank Check	2040 CHECK Mail Bank Check	
Visa	3000 CCARD Visa	
MasterCard	3010 CCARD Mastercard	
AmEx	3020 CCARD American Express	
Discover	3030 CCARD Discover	
DinersClub	3040 CCARD Diners Club - N. America	
HouseCard	3120 CCARD House Card	
JCB	3130 CCARD JCB	
CASH Cash Alternate Currency	1010 CASH Cash Alternate Currency	
CHCK Check Alternate Currency	2050 CHECK Personal Check Alternate Currency	
TRAV Travelers Check Alternate Currency	2060 CHECK Travelers Check Alternate Currency	
STCR Store Credit Alternate Currency	4090 VOUCH Store Credit Alternate Currency	
GICT Gift Certificate	4100 VOUCH Gift Certificate Alternate Currency	

Table 2–9 CCEM

Credit card input type	CCEM	Dynamic
Manual	T	Terminal Used
MSR	MSR	Magnetic Strip Read

Table 2–10 Unit of Measure

Unit of Measure	
'LF' 'linear feet'	'LF' 'linear feet'
'LM' 'linear meters'	'LM' 'linear meters'
'PN' 'pounds net'	'LBS' 'POUNDS'
'KG' 'kilograms'	'KG' 'KILOGRAM'
'UN' 'units'	'EA' 'EACH'

Table 2–11 Total ID for TOTAL type transactions

Total ID (Reference Number 1) for TOTAL type transactions.	
1000 CASH Cash - primary currency	CASH
2000 CHECK Personal Check	CHCK
2020 CHECK Traveler Check	TRAVCHK
5000 COUPON Manufacturers Coupons	QPON
8000 DCARD Debit Card	DEBITCARD
6000 MORDER Money Orders	MNYORDER
4030 VOUCH Gift Certificate	GIFTCERT
4040 VOUCH Gift Card	GIFTCARD
4050 VOUCH Store Credit	STCREDIT
4060 VOUCH Mall Certificate	MALLCERT
4070 VOUCH Purchase Order	PRCHORDER
2030 CHECK E-Check	ECHECK
2040 CHECK Mail Bank Check	MBCHECK
3000 CCARD Visa	CCARDVisa
3010 CCARD Mastercard	CCARDMCard
3020 CCARD American Express	CCARDAmEx
3030 CCARD Discover	CCARDDisc
3040 CCARD Diners Club - N. America	CCARDDiner
3120 CCARD House Card	CCARDHCard
3130 CCARD JCB	CCARDJCB
1010 CASH Cash Alternate Currency	CASHAC
2050 CHECK Personal Check Alternate Currency	PCHECKAC
2060 CHECK Alternate Currency	TCHECKAC
4090 VOUCH Store Credit Alternate Currency	STCRDTAC
4100 VOUCH Gift Certificate Alternate Currency	GIFTCERTAC

Table 2–12 PRMT

PRMT		
The RMS promotion type.		Static
Computed	1004	In Store Discount
Computed	1005	Employee Discount
Computed	9999	Promotion

Implementation Configuration

Data Import Spring Configurations

The system has been designed to support a pluggable model. The DIMP Controller, ImportTranslator, ImportController, ImportDAO, MessageLogger and scheduler are all designed to be configurable at deployment time. This is accomplished through the use of Spring as a deployment framework. Each of these classes is only accessed through their interface. Therefore, any new implementations only need to support the interfaces to be used by the subsystem. Introducing an alternate implementation is done through updates to the Spring configuration file. No additional code changes are necessary.

Table 3–1 includes the set of Spring bean IDs used for each of the pluggable components.

Note: 1 to $2^{64} - 1$ is the logical range of the batchSize, though database performance may require the upper limit to be much smaller than that. Only the implementation team will be able to determine what the actual upper limit should be based upon database performance.

Table 3–1 Spring Bean IDs Used For Each Of The Pluggable Components

Spring bean ID	Provided implementation	Default Configuration
service_MerchandiseHierarchyImportTranslator	com._360commerce.commerceservices.item.hierarchy.importdata.MerchandiseHierarchyImportTranslator	batchSize=1000
service_StoreHierarchyImportTranslator	com._360commerce.commerceservices.store.hierarchy.importdata.StoreHierarchyImportTranslator	batchSize=1000
service_TaxImportTranslator	com._360commerce.commerceservices.tax.importdata.TaxImportTranslator	batchSize=1000
service_EmployeeImportTranslator	com._360commerce.commerceservices.employee.importdata.EmployeeImportTranslator	batchSize=1000

Table 3–1 Spring Bean IDs Used For Each Of The Pluggable Components

Spring bean ID	Provided implementation	Default Configuration
service_MessageLogger	com._ 360commerce.commerceservices. common.importdata.MessageLogger	messages=messageSource
service_ImportJobTrigger	org.springframework.scheduling. quartz.SimpleTriggerBean	JobDetail=service_ ImportOrchestratorStartDelay=10000 RepeatInterval=10000
service_ImportOrchestrator	org.springframework.scheduling. quartz.JobDetailBean	com._ 360commerce.commerceservices.import data.ImportOrchestratorJob
DIMP_Scheduler	org.springframework.scheduling. quartz.SchedulerFactoryBean	triggers=service_ ImportJobTriggerAutoStartup=true ApplicationContextSchedulerContext Key=applicationContextWaitForJobsTo CompleteOnShutdown=true

These setting can be found in the `ServiceContext.xml` file packaged in the `config.jar` under the `/config/context` package.

The `web.xml` in `WEB-INF` directory has the following configuration under the `web-app` section.

```
<context-param>
<param-name>contextConfigLocation</param-name>
<param-value>/WEB-INF/schedulingContext-quartz.xml</param-value>
</context-param>
```

The following servlet should also be configured to start up automatically. The servlet loads the context configuration files. Because the `schedulingContext-quartz.xml` file is configured in the context, this file is loaded by the servlet. `SchedulerFactoryBean` is configured to start on load; hence it is invoked and starts the scheduler timer.

```
<servlet>
<servlet-name>context</servlet-name>
<servlet-class>org.springframework.web.context.ContextLoaderServlet</servlet-class>
>
<load-on-startup>1</load-on-startup>
</servlet>
```

[Table 3–2](#) includes additional sets of Spring bean IDs used for each of the pluggable components.

Table 3–2 Additional Spring Bean IDs Used For Each Of The Pluggable Components

Spring bean ID	Provided implementation	Additional configuration
persistence_ImportController	com._ 360commerce.commerceservices.impo rtdata.ImportController	batchSize=1000
persistence_MerchandiseHierarchyImport DAOTarget	com._ 360commerce.commerceservices.item. hierarchy.importdata.dao. MerchandiseHierarchyImportDAO	dataSource=persistence_dataSource

Table 3–2 Additional Spring Bean IDs Used For Each Of The Pluggable Components

Spring bean ID	Provided implementation	Additional configuration
persistence_ StoreHierarchyImportDAOTarget	com._ 360commerce.commerceservices.store. hierarchy.importdata.dao.StoreHierar chyImportDAO	dataSource=persistence_dataSource
persistence_TaxImportDAOTarget	com._ 360commerce.commerceservices.tax. importdata.dao.TaxImportDAO	dataSource=persistence_dataSource
persistence_ EmployeeImportDAOTarget	com._ 360commerce.commerceservices. employee.importdata.dao.Employee ImportDAO	dataSource=persistence_dataSource

These settings can be found in the PersistenceContext.xml file packaged in the config.jar under the /config/context package.

By default, the ImportController's batch size is set to 1000 and all the translators are also using the same. However, each individual translator can be configured separately to optimize the import per the size of the data operation. Spring sets the batch size value onto the translator when instantiated. It is the responsibility of the translator to call setBatchSize(int) with that value onto the ImportController.

Notice that the ID of the DAO beans end with Target. This is because the ID that is actually used by the application returns a Proxy Bean configured to intercept method calls to the DAO and associate transactions with them. Upon ImportExceptions thrown by those methods, the transaction is rolled back.

Several configuration files exist containing settings specific to DIMP. Properties are read when the server starts, so any changes require a server restart before they take effect.

spring.properties

You can find the spring.properties in the following location:

```
<INSTALL_DIR>\profiles\AppSrv01\properties
```

The following is an example spring.properties file:

```
#####
## Global settings (applicable to OC4J and WAS) ##
#####

# directory in which incoming data import bundles arrive
importdata.file.path=C:/temp/dataimport/incoming

# directory in which dimp bundles are archived after processing
importdata.archive.path=C:/temp/dataimport/archive

# true/false whether data import scheduler should scan importdata.file.path
execute.import=false

# the delay in milliseconds to start the import and price change triggers (900000
= 15 minutes)
import.scheduler.startdelay=60000

# the delay between import and price change scheduler executions (86400000 = 24
hours)
```

```
import.scheduler.repeatinterval=86400000
pricechange.scheduler.repeatinterval=3600000
```

```
# name of the DIMP logger config file
logger.filename=dimplogger
```

`importdata.file.path` and `importdata.archive.path` are file-system dependent. Windows systems would use paths such as:

```
C:/temp/dataimport/incoming
```

Linux systems would use paths such as:

```
/tmp/dataimport/incoming
```

`execute.import` determines whether or not data imports execute in the environment. Its default is `false`.

`import.scheduler.startdelay` is a value, in milliseconds, that determines the interval from when the Quartz scheduler starts and the import process is executed for the first time. For example, a value of 60000 means that the scheduler is delayed for one minute.

`import.scheduler.repeatinterval` is a value, in milliseconds, that determines how often the import process is run.

`pricechange.scheduler.startdelay` and `pricechange.scheduler.repeatinterval` are used by Back Office only. They are the Quartz scheduler settings for the process that applies imported price changes when they are about to go into effect or expire.

`logger.filename` points to another properties file containing the string values that can be customized for DIMP messages.

dimplogger.properties

This is the file referred to by the value, `logger.filename`, in `spring.properties`. It contains text values that can be customized to make DIMP messages easily distinguishable in the `oracleretail` log file.

Every DIMP message appears with the `dimp.` prefix. `dimp.text1`, `dimp.text2` and `dimp.text3` are used depending on how much information is supplied by the underlying system.

Archive File Format

The Archive File is of the following format:

```
META-INF
  MANIFEST.MF
ItemImport-12345-20032-007.xml
PriceImport-12345-20032-007.xml
StoreHierarchy.xml
...
```

The suggested file naming convention for the archive is as follows:

```
[arbitrary_portion]-[store_id]-[YYYYMMDD]-[NNN].jar
```

Where [arbitrary_portion] can be used by the implementation team for any value, and [NNN] is the batch ID in the range of 0 through $2^{32}-1$, or 2,147,483,647 (because of the limitations of the XSD int datatype). This is a sequential number that is used to determine the processing order for archives, if more than one exists on the server at a time. The date is only available for visual reference. If the file name is not formatted as above, the values in the manifest are used instead. However, if both the archive file name and the file names within the manifest contain a batch ID, the value in the archive file name takes precedence.

There is no restriction on the file names and they can be in any format. But the exact file names have to be listed in the MANIFEST.MF.

The format of the MANIFEST.MF is as follows

Manifest-Version: 1.0

```
# This manifest describes the contents of an archive referred to as a
# bundle. The following two values list the ID of the batch that
# produced this bundle and the ID of the destination store to receive
# it. The BatchID should be numeric less than  $2^{32}-1$ .
```

BatchID: <N>

StoreID: <NNNNN>

```
# The following section lists the files contained in this bundle archive.
# Each key should begin with "FileN" without quotes and N being a number.
# The value of the key consists of a bundle entry file name followed
# by hard brackets containing a list of files that should be processed
# before it.
#
# e.g. File1: ItemImport.xml[TaxImport.xml,StoreHierarchyImport.xml]
#
# The order of the files or their dependency list is not important.
```

File1: <filename1>[<optional dependencies>]

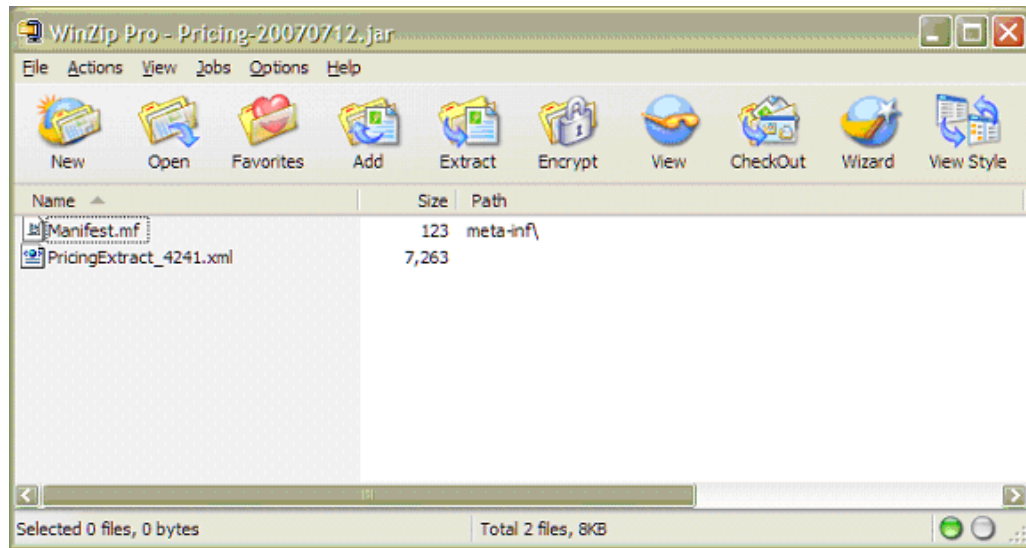
...

FileN: <filenameN>[<optional dependencies>]

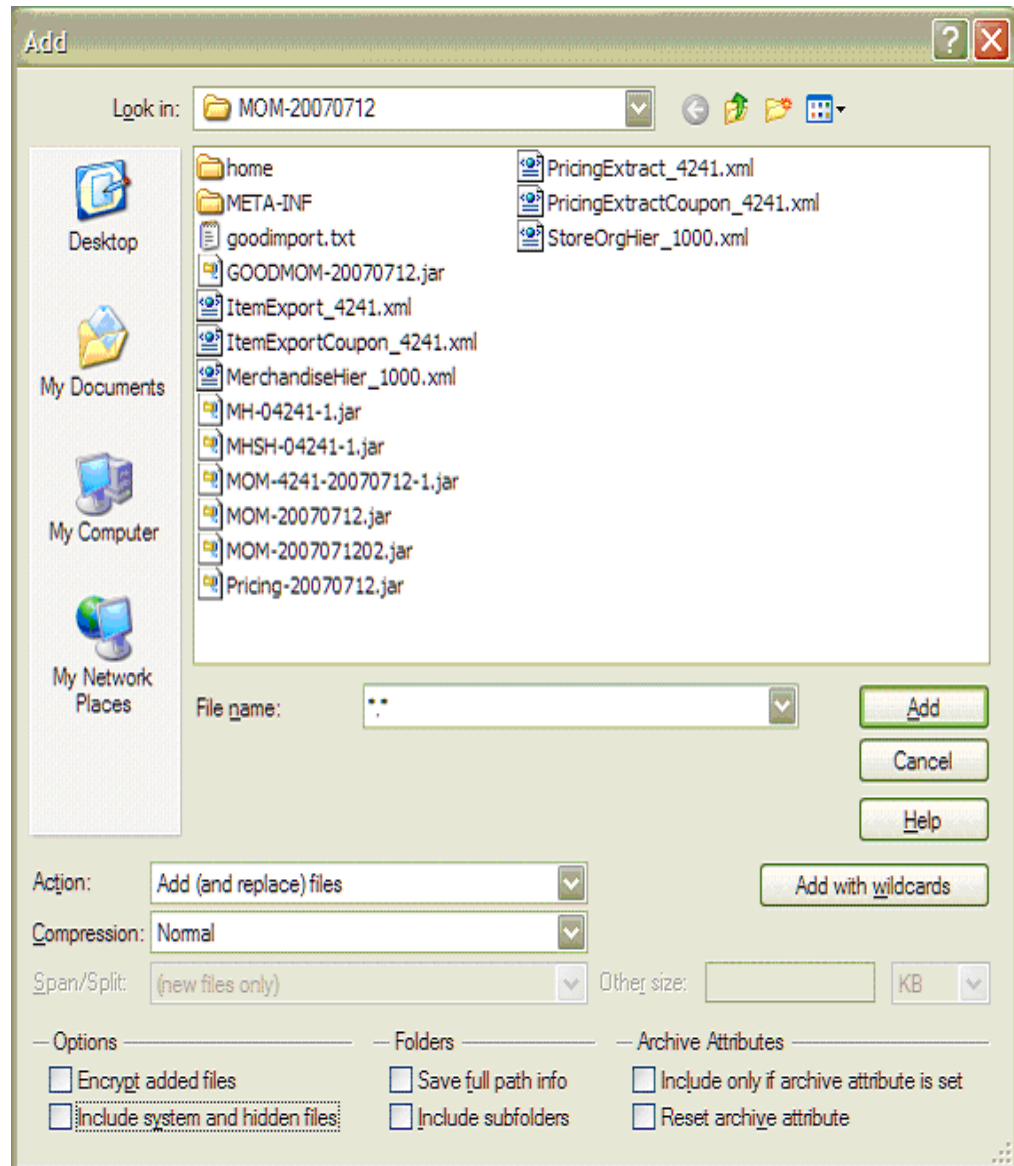
With the exception of manifest.mf, path names should not be used when creating the manifest. In the figure below, note that the path column is empty except for meta-inf, the path for manifest.mf.

Note: Although WinZip cannot be used to create a bundle, it can be used to inspect the bundle, as well as add, delete, or modify the XML contents. Use the following Jar command line utility to create a bundle:

```
C:\temp\dataimport\archive>%JAVA_HOME%\bin\jar -cvfm test_
coupon3.jar manifest_details.txt
PricingImportSample_addCouponDiscount.xml ItemImportSample_
addCoupon.xml
```

Figure 3–1 Adding Files To a Jar

In the following screen shot of the dialog box for adding files to a WinZip archive, note that the **Save full path info** option at the bottom is unchecked.

Figure 3–2 Adding Files To A WinZip Archive

The following is an example of a manifest file:

```
Manifest-Version: 1.0

# This manifest describes the contents of an archive referred to as a
# bundle. The following two values list the ID of the batch that
# produced this bundle and the ID of the destination store to receive
# it. The BatchID should be numeric less than 2^32-1.

BatchID: 1
StoreID: 04241
File1: ItemImportSample_addCoupon.xml[]
File2: PricingImportSample_addCouponDiscount.xml[ItemImportSample_addCoupon.xml]

# The following section lists the files contained in this bundle archive.
# Each key should begin with "FileN" without quotes and N being a number.
# The value of the key consists of a bundle entry file name followed
# by hard brackets containing a list of files that should be processed
# before it.
#
# e.g. File1: ItemImport.xml[TaxImport.xml,StoreHierarchyImport.xml]
#
# The order of the files or their dependency list is not important.

File1: TaxImport.xml[]
File2: MerchandiseHierarchyImport.xml[]
File3:
ItemImport.xml[TaxImport.xml,MerchandiseHierarchyImport.xml,StoreHierarchyImport.x
ml]
File4: ItemImport2.xml[ItemImport.xml]
File5: PriceImport.xml[ItemImport2.xml]
File6: StoreHierarchyImport.xml[]
File7: EmployeeImport.xml[StoreHierarchyImport.xml]
```

Oracle Retail Merchandising System Configuration

If the retailer is integrating with Oracle Retail Merchandising System, it is assumed that the retailer is setting up items within Oracle Retail Merchandising System, and not using this feature in Back Office. If the retailer chooses to add or edit an item within Back Office, then that item data might be overridden by the next download from Oracle Retail Merchandising System.

Some data fields are defaulted to the values shown in [Table 3–3](#).

Table 3–3 Oracle Retail Merchandising System Default Values in the Back Office Item Maintenance Screen

Back Office Data Field	Default Value when integrating with Oracle Retail Merchandising System or Limitation
Cost	0
Class	Items belong to one class only
Manufacturer	Null
Planogram	Null
Labels/Tags Template Type	Default
Serialized	FALSE
Restocking Fee	FALSE
Activation Required	No

Table 3–3 Oracle Retail Merchandising System Default Values in the Back Office Item Maintenance Screen

Back Office Data Field	Default Value when integrating with Oracle Retail Merchandising System or Limitation
Registry Eligible	No
Special Order Eligible	No
Employee Discount Eligible	Yes
Damage Discount Eligible	Yes
Size Entry Required	No
Authorized for Sale	Active
Item Department	The first department in the drop down list. If no Item Department is specified, then the value is defaulted to the first value in the drop down list.

Service items (non-merchandise items that are non-inventory) need to be loaded separate from the download process.

In Oracle Retail Merchandising System, differentiators **1** through **4** are sent as values and are mapped to STYLE, COLOR, SIZE and NULL in Point-of-Service.

Oracle Retail Price Management Configuration

If the retailer is integrating with Oracle Retail Price Management, it is assumed that the retailer is setting up items within Oracle Retail Price Management, and not using this feature in Back Office. If the retailer chooses to add or edit an item within Back Office, that item data might be overridden by the next download from Oracle Retail Price Management.

Note: You must edit the Data Definition Language (DDL) before building the store's database when integrating with Oracle Retail Price Management.

In the files `CreateTableTemporaryPriceChangeItem.sql` and `CreateTablePriceDerivationRule.sql` there are the following two lines:

```
-- Uncomment and use this index for Oracle Retail Price Management
(RPM) integrations
-- CREATE UNIQUE INDEX ITM_TMP_PRM_IDX ON MA_ITM_TMP_PRC_CHN (ID_
PRM, ID_PRM_CMP, ID_PRM_CMP_DTL);
```

Remove the dashes that start the second line so that when the database is built, these three columns (that contain Oracle Retail Price Management IDs) create a unique index.

In the first phase of the integration, some data fields are defaulted to the values shown in [Table 3–4](#).

Table 3–4 Oracle Retail Price Management Default Values

Back Office Screen	Back Office Data Field	Default Value when integrating with Oracle Retail Price Management or Limitation
Discount Rule	Start Time	0:00
Discount Rule	End Time	23:59:59
Discount Rule	Source	Promotions defined at Item Level only
Discount Rule	Target	Promotions defined at Item Level only
Discount Rule	Source Threshold	None
Discount Rule	Source Limit	None
Discount Rule	Target Threshold	None
Discount Rule	Target Limit	None
Discount Rule	Number of Times Per Transaction	-1
Discount Rule	Accounting Method	Discount
Discount Rule	Allow Source to Repeat	Yes
Discount Rule	Deal Distribution	Target
Discount Rule	Target Quantity	1
Price Maintenance	Start Time	0:00
Price Maintenance	End Time	23:59:59
Price Maintenance	Status	Back Office gets status from effective date and record descriptors
Price Maintenance	Template Type	Default

Data Requirements – Oracle Retail Strategic Store Solutions to Oracle Retail Sales Audit

For Point-of-Service there are two types of data that must be provided for the Point-of-Service to start, and before the task of ringing up items begins:

- Base Data – the basic configuration data. This includes:
 - currency
 - password policy
 - user role
 - work group information

This information is provided with the application.

- Store and Application Data (Seed Data) – this is data that differentiates one store from another. This information includes:
 - item
 - employee
 - pricing rules

Examples of Base Data insert statements can be found in the build at
`<root>\modules\utility\deploy\360common\db\sql\Base` and in the
 Point-of-Service installation at
`\OracleRetailStore\Server\360common\db\sql\Base`.

Examples of Seed Data insert statements can be found in the build at
`<root>\modules\utility\deploy\360common\db\sql\Seed` and in the
 Point-of-Service installation at
`\OracleRetailStore\Server\360common\db\sql\Seed`.

Loading base data and Merchandise Operations Management bundles into the Store level database does not provide enough data for the Point-of-Service to initiate; this combination is missing some required data, such as the store's tax geocode and employees to log into the application. You must provide a minimum of additional information. The additional information can be provided through .sql statements:

- InsertTableCodeList.sql – data used in Point-of-Service drop down lists for pay-in/out, Tax overrides and so forth.
- InsertTableEmployee.sql – employee information.

Note: The implementation team must supply this information through a third-party application.

- InsertTableEmployeeHierarchyAssociation.sql – employee information.

Note: The implementation team must supply this information through a third-party application.

- InsertTableStoreSafeTender.sql – tenders used in the store.
- InsertTablePriceDerivationRuleType.sql – price derivation rule types.

Tax Import Configuration

Batch Location

The tax factor calculation batch file is packaged as part of the ORPOS installer. Once the ORPOS server component is installed, the batch file will be found in the below location.

`<INSTALL_DIR>\Server\pos\bin\L10N\IN\Batches\TaxCalculation-Batch`

Execution process

The following are the pre-requisites:

- Update the setenv.bat to point to correct the ORACLE_HOME (Database) and JAVA_HOME.

For example:

```
SET JAVA_HOME=C:\j2sdk1.4.2_16
```

```
SET ORACLE_HOME=D:\oracle\product\10.2.0\db_1
```

```
SET PATH=%JAVA_HOME%\bin;%PATH%
```

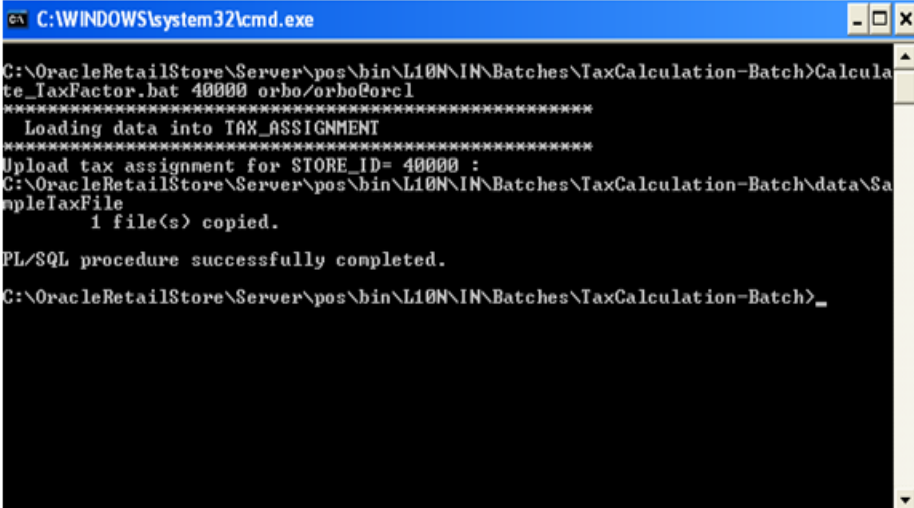
```
Set PATH=%ORACLE_HOME%\bin;%PATH%
```

- India L10N related ddl scripts have already been run against the StoresDB.
CALCULATE_TAXFACTOR_SQL should already be present in the DB package.
- Make sure the tax flat file is always present in the data folder before executing the Calculate_TaxFactor.bat.

Note: Once the Calculate_TaxFactor.bat is executed, the flat file is deleted.

Steps

- Navigate to <INSTALL_DIR>\Server\pos\bin\L10N\IN\Batches\TaxCalculation-Batch in command prompt.
- Place the tax flat file in the data folder (<INSTALL_DIR>\Server\pos\bin\L10N\IN\Batches\TaxCalculation-Batch\data).
- Run the setenv.bat.
- Run the Calculate_TaxFactor.bat with the Store ID as the parameter and the DB connection string.
Usage: Calculate_TaxFactor.bat <<Store_Id>> username/password@SID.
- If the tax information is uploaded successfully, then a successful execution message is displayed.
- Check the TX_ASGMT table to verify whether the tax data is uploaded properly.

Figure 3–3 Tax Assignment successfully completed


```

C:\WINDOWS\system32\cmd.exe

C:\OracleRetailStore\Server\pos\bin\L10N\IN\Batches\TaxCalculation-Batch>Calculate_TaxFactor.bat 40000 orbo/orbo@orcl
*****
Loading data into TAX_ASSIGNMENT
*****
Upload tax assignment for STORE_ID= 40000 :
C:\OracleRetailStore\Server\pos\bin\L10N\IN\Batches\TaxCalculation-Batch\data\Sa
pleTaxFile
1 file(s) copied.

PL/SQL procedure successfully completed.

C:\OracleRetailStore\Server\pos\bin\L10N\IN\Batches\TaxCalculation-Batch>_

```

Data Requirements for India L10N Stores

As part of India L10N, new columns have been added to the existing base tables, new tables have been added, and certain base data has been modified so that it can work for India Locale. The base product's ddl scripts have not been modified. Additional ddl, dml scripts have been written which would run on top of the base ddl scripts. By running the additional India L10N specific database scripts, a database for Indian locale will be created and various other data required for India L10N will be configured. All the India L10N related database scripts can be found at the below mentioned location.

```
<root>\modules\utility\deploy\360common\db\sql\L10N\IN
```

All the alter and create script changes for India L10N are mentioned in the Data Model Documentation for India L10N. Certain data needed to be configured/seeded for India L10N these are achieved through the scripts under Configure Data folder. Below are the scripts which configure data for India L10N

- ConfigureDepartment.sql: Updates the en_IN locale in the ID_DPT_PS_I8 and ID_DPT_PS_I8 table instead of en_US locale.
- ConfigureEmployee.sql: Updates the PA_EM and EMPLOYEE_HIERARCHY_ASSN tables to the store ID entered during installation.
- ConfigureGeoCode.sql: Inserts the Geocode for en_IN locale in the CD_GEO, GEO_TX_JUR and CO_TX_JUR_ATHY_LNK tables. This is done to avoid the "Geo Code Missing" problem in point of service.
- ConfigureINRCurrency.sql: Inserts the INR currency into the currency related tables.
- ConfigureTaxBreakupReport.sql: Inserts the report configuration for the tax breakup summary for the transaction tracker screen in the central office.

The caltaxfactorsql.pls package for calculating the tax factors based on the tax assignment can be found in the "packages" folder.

Capacity Planning

This section lists the approximate hard drive sizes that are required at each store to be able to support the Data Import project.

The following assumptions were made to arrive at an approximate capacity:

- The archival period is one week.
- The frequency is one import bundle per day.
- The TAX Import file is not part of the Import Bundle.
- Peak Load for the EMPLOYEE Import is 30 employees per file.
- The Peak Load Capacity of each file is taken into consideration for the estimation. See [Table 4-1, "File Sizes"](#).
- The average compression ratio in creating a jar file is considered to be 60%.
- As the frequency is one bundle per day, and the archival period is one week, therefore the maximum number of files on the disk is eight.
- A footprint on the DDI (Data Distribution Interface) on the Store Server is considered to be the size of one bundle and added to the final estimate. The footprint on the DDI is not part of the scope of the DIMP.
- Because the peak load size for Merchandise Hierarchy is not defined, a load of 5000 records is estimated.

[Table 4-1](#) identifies the file sizes for components in the data import at a store.

Table 4-1 File Sizes

Type of Import	One-Record Size in Bytes	Peak Load (Number of Records)	Peak File Size in Bytes
Item	950.00	15,000,000.00	14,250,000,000.00
Pricing	1,600.00	820,000.00	1,312,000,000.00
Store	710.00	5,000.00	3,550,000.00
Merchandise	300.00	5,000.00	1,500,000.00
Employee	1,400.00	30.00	42,000.00

Total Size of Files

15,567,092,000.00 Bytes

Table 4–2 identifies the sizes of data import bundles.

Table 4–2 Bundle Size

Bundle Size (jar Size)	Assuming 60% Compression Ratio in creating a jar	9,340,255,200.00	Bytes
		8,900.00	MB
Approximate Bundle Size		8.69	GB

Table 4–3 identifies the required hard-drive capacities to enable a data import.

Table 4–3 Hard Drive Capacity

Seven files in Archive + One File in current	71,200.00	MB
	69.53	GB
Approximate Hard Drive Size to retain the Bundles	70.00	GB
Footprint on DDI Store Server (the DDI remains the responsibility of the implementation team to implement) - assuming size of one Bundle	78.69	GB

Required Hard Drive Capacity (Approximate)

80.00 GB

Table 4–4 Item Import Data Volumes

Data Volumes		
Item	800,000 – 1.5 million for peak season 5000 – 15,000 for delta	1.5 million
Item Location	See Item	See Item
Item (Merchandise) Hierarchy	number of departments	number of departments
	groups	number of groups
	number of hierarchies	number of hierarchies
Organizational (Store) Hierarchy	5000 stores, 6 levels	5000 stores, 6 levels
	number of regions	
	number of districts per region	
	number of stores per district.	
Tax data	See Item (since any tax info is limited to item related attributes such as tax group ID) *Tax info comes from Oracle Retail Merchandising System	See Item (since any tax info is limited to item related attributes such as tax group ID) *Tax info comes from Oracle Retail Merchandising System

Pricing Import Data Volumes

Data Volumes: 800000 price changes per day per store.

Customization Notes

Data Import Extension Points and Development

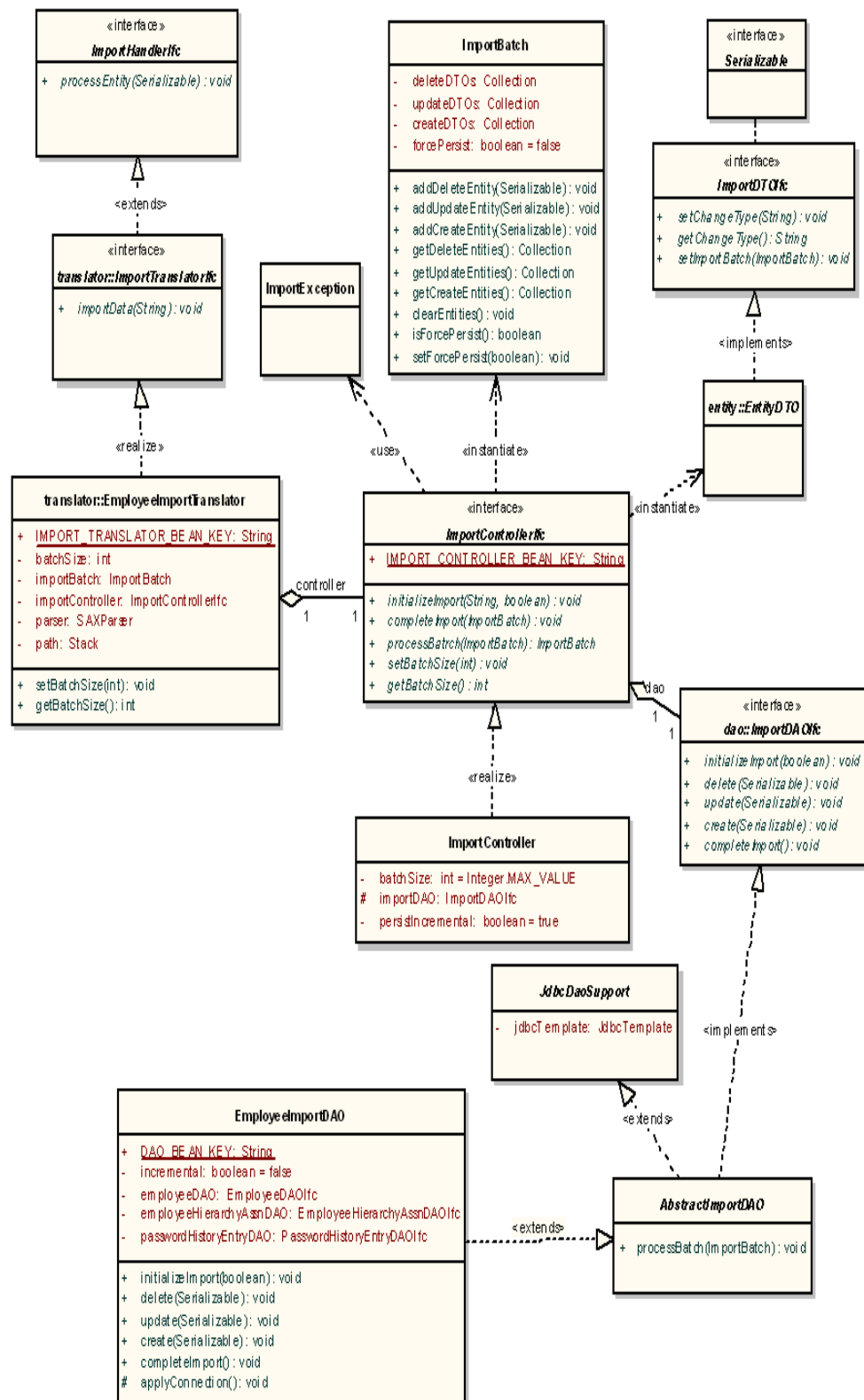
Oracle Store Solutions has provided not only extension points for enhancing or modifying the capabilities of the existing data imports, but there are also tools provided for jump-starting an altogether new data import. Do the following to create a new data import module:

1. Compose XSD that the import data conforms to.
2. Generate sample XML based on the XSD. This can be done using the Eclipse EMF plug-in. See <http://www.eclipse.org/>
3. Map the XSD to the Data Model.
4. Create a Message Driven Bean and update the appropriate deployment descriptors.
5. Use SAXParserGenerator with XSD.
6. Use DAOGenerator to generate data access objects (DAO) for tables mapped to.
7. Rename DAO classes to match logical names of tables.
8. Delete duplicate DTOs or DAOs that might exist in other packages and that can be reused.
9. Update DAOIfc method parameters to pass actual DTO objects.
10. Remove column names from UPDATE_SQL that are not updated during update procedure from DAO and SQLIfc.
11. Update DAO get*Statement() methods to map DTO fields to PreparedStatement buckets.
12. Create a test that reads the XML and sends it to translator. How the XML is created or read is not important at this time, nor is using Spring or JUnit or AppServer.

The following sections discuss these steps in more detail. Where these steps overlap with steps for enhancement (as opposed to steps for creating new imports), the enhancement steps are identified.

First, extension points are identified, and techniques for enhancing existing data imports are described. Each of the previously mentioned DIMP modules (Taxation, Merchandise Hierarchy, Store Hierarchy, and Employee) follow the same patterns of implementation and vary in minor details only. We concentrate on Employee. The following diagram is the Employee Data Import Static Model.

class Import Components



Import Adapter and Translator

The entry point for data imports is the `ImportIOAdapterIfc`. It is configured through a Spring context as either `EEImportIOAdapter`, for JCA implementations, or `FileImportIOAdapter` for direct file I/O implementations. The IO Adapter retrieves the bundles from the file system, determines the processing order, and passes the XML stream data to the `ImportInitiator`, which determines the import type from the payload and passes the string to a translator. The `ImportInitiator` (as the `BeanLocator`) provides an `ImportTranslatorIfc` from the service context by passing the key `EmployeeImportTranslator.IMPORT_TRANSLATOR_BEAN_KEY`, for example.

The following example shows the `EEImportIOAdapter` implementation in use:

```
<!-- Import IO Adapter Implements com._
360commerce.commerceservices.importdata.ImportIOAdapterIfc -->
<bean id="service_ImportIOAdapter" class="com._
360commerce.commerceservices.importdata.EEImportIOAdapter">
</bean>
<!--<bean id="service_ImportIOAdapter" class="com._
360commerce.commerceservices.importdata.FileImportIOAdapter">
```

SAXParserGenerator

If creating a new data import module and starting with a defined XSD, a simple utility can be run to generate code for a Translator, SAX handlers, simple DTO, and a skeleton Import DAO. The following is an example of how to run this utility.

Example 5–1 *SAXParserGenerator utility command prompt*

```
<root>\modules\utility>java
com._360commerce.codegen.importtranslator.SAXParserGenerator "C:\Data
Import\Design\Employee\EmployeeImport.xsd"
com._360commerce.commerceservices.employee.importdata
../../commerceservices/employee/src
```

This command line example shows that the utility program is Java-based and takes three arguments:

- The location of the XSD file.
- The desired package name for the generated source code.
- The directory in which to place new source code files.

This utility can be configured as an executable target in your favorite Integrated Development Environment (IDE) so this utility can be run again as changes continue to be made to the XSD which defines the format of the new data input.

The code generation uses the Java-based Velocity templates and APIs. See <http://jakarta.apache.org/velocity>. The templates can be found at `<root>/modules/utility/templates/`.

Manually Editing Generated Code

The generated code requires additional manual editing before it can be used. For example, the `ImportDAO` has only the barest of implementations in its methods. Add code to pass various DTOs to the correct DAO that can handle it.

Appropriate DTOs might already exist in the codebase. Examine the attributes of the pre-existing DTO to see if it or the generated DTO should be used. In some cases, additional code might need to be added. For example, if you consider that a single-entity DTO usually represents a single record in the database, the SAX handlers are coded to not process child DTOs passed to the SAX handlers until the DTO that a SAX Handler creates is successfully processed.

Example 5-2 EmployeeAccessHandler Process DTO Before Children

```
/**
 * End handling this element. Calls {@link
 * ImportHandlerIfc#processEntity(java.io.Serializable)}
 * @throws SAXException
 */
public void end() throws SAXException
{
    try
    {
        // process this first
        parent.processEntity(employeeAccessDTO);

        // process all its children
        Iterator iter = children.iterator();
        while (iter.hasNext())
        {
            Serializable child = (Serializable)iter.next();
            parent.processEntity(child);
        }
    }
    catch (ImportException e)
    {
        logger.error("Could not end element " + getText(), e);
        throw new SAXException("Could not end element " + getText(), e);
    }
}
```

However, in some cases, such as when there are important attributes that are needed to fill the DTOs, and which need to be persisted immediately, the call to `parent.processEntity(Serializable)` can be commented out of the `end()` method and added to the `start(Attributes)` method. The `start(Attributes)` method is called when parsing the beginning of the XML element. Notice in the following example, the value for "Incremental" defaults to true if it does not exist.

Example 5-3 EmployeeImportHandler Process DTO During Start

```
/**
 * Start handling this element by inspecting its attributes, if any.
 * @param attributes the attributes given.
 * @throws SAXException
 */
public void start(Attributes attributes) throws SAXException
{
    String incremental = attributes.getValue("Incremental");
    Boolean bIncremental = (incremental != null)? Boolean.valueOf(incremental)
: Boolean.TRUE;

    employeeImportDTO.setEmployeeImportIncrementalAttribute(bIncremental.booleanValue(
));

    try
```

```

    {
        // process this first
        parent.processEntity(employeeImportDTO);
    }
    catch (ImportException e)
    {
        logger.error("Error starting import" + employeeImportDTO, e);
        throw new SAXException("Error starting import" + employeeImportDTO,
e);
    }
}

```

There also might be a scenario where parent XML element values, such as IDs, are required for child DTO objects. These attributes might have to be added manually to the DTOs and set by the handlers. See the Merchandise Import DTO, LevelDTO as an example, and the handlers that call its set methods.

If it seems that the SAX handlers or the DTOs are missing attributes for defined XML elements, there might be errors in the XSD that the SAXParserGenerator cannot decipher. Ensure that your XSD validates properly based upon the schema at <http://www.w3.org/2001/XMLSchema>.

Metadata

The top-level element of each import includes metadata pertaining to the import bundle. Among other possible uses, this data is included in import bundle tracking and error logging. The following is an example XML fragment. Consult the development team for the status of data import schemas beyond this release.

```

<ItemImport
    Priority="0"
    FillType="FullIncremental"
    Version="1.0"
    Batch="1"
    CreationDate="2001-12-17T09:30:47.0Z"
    ExpirationDate="2007-12-17T09:30:47.0Z"
    xsi:noNamespaceSchemaLocation="ItemImport.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    . . .

```

The metadata attributes are defined as follows:

Priority

An integer specifying the order, from lowest to highest, in which multiple files of one type in a bundle should be processed.

FillType

The feed method: Kill And Fill, Delta Incremental, or Full Incremental. The XSD specifies which of these are allowed for an import type. For example, Tax allows only Kill And Fill, while Item allows all three.

Version

The version of the application processing the data.

Batch

An integer sequence number, corresponding to the ID of the process that created the file.

CreationDate

A timestamp identifying the file's creation time.

ExpirationDate

A timestamp beyond which a file has become stale and should not be processed.

ImportControllerIfc

The current implementation of the ImportControllerIfc operates well in most circumstances. However, there might be circumstances that call for a different version of the controller to be plugged in. For example, a new controller might put a parsed batch onto one of many secondary queues instead of passing it synchronously to a DAO, then returning control to the translator to continue parsing the import.

The secondary queue is another thread that takes the incoming batch and passes it to an instance of the import DAO. This enables multiple batches to be processed at once.

Strategic Store Solutions to Oracle Retail Sales Audit Extension Points and Development

There are three distinct situations in which a implementation team would need to extend the functionality in the Export File Generator:

- Adding data elements to the RTLog Format.
- Creating an entirely new fixed length export format.
- Creating an entirely new export format which is not fixed length.

Adding Data Elements to the RTLog Format

To add VAT information added to the one or more of the reference fields in the Transaction Item record to the RTLog a implementation team takes the following steps:

1. Define the format of the VAT data.
2. Depending on the outcome of step 1, it might be advantageous to modify the definition of a Reference field in the Transaction Item record. This cause the creation of Acme-specific Export Format Configuration file. If this is desirable, copy RTLogFormat.xml to AcmeRTLogFormat.xml and make the modifications in this file.
3. Define how the columns in the table TR_LTM_SLS_RTN_TX map to the format defined in step 1.
4. Write a FieldMapper class called AcmeItemVATTax.java to perform the mapping.
5. Copy RTLogMappingConfig.xml to AcmeRTLogMappingConfig.xml and make the following change to the new file:

```
<TABLE table="TR_LTM_SLS_RTN_TX">
    <MAP column="MO_TX_RTN_SLS" record="TransactionTax" field="TaxAmount"

    fieldMapper="com.acme.exportfile.RTLog.fieldmappers.AcmeItemVATTax" />
</TABLE>
```

6. Modify StoreServerConduit.xml to use AcmeRTLogMappingConfig.xml and AcmeRTLogFormat.xml instead of RTLogMappingConfig.xml and RTLogFormat.xml.

If the Reference field is partitioned correctly, and the values coming from the database to these new fields do not requires manipulation, then it is possible that the FieldMapper class is not required.

Creating a New Fixed Length Export Record Format

Currently, Oracle Retail has only one way to send transactional data to a customer's back end systems: POSLog. However, it is expensive and time consuming to extend POSLog, to explain it to customers and to develop the code that loads it into the customer back end.

It might be faster and cheaper to use the Export File Generator to generate the transaction log format that the customer is already consuming.

The generation of all three current formats (DTM for Central Office, POSLog for the customer backend, and RTLog for Oracle Retail Sales Audit) simultaneously has been tested in the development environment.

Here are the steps to create transaction log export code for "Acme", a generic customer:

1. Work with Acme developers to create a mapping document that describes the relationship between the Oracle database and the current Acme back end system/transaction log format. A mapping exercise of this type must be done even if the customer eventually chooses to use the POSLog to transfer the data. Understanding the customer's current transaction log can provide valuable insight into the data requirements.
2. Construct an Acme-specific Export Format Configuration file which describes all the records in the Acme transaction log; call this file AcmeTLogFormatConfig.xml.
3. Create an Acme-specific Mapping configuration file; call this file AcmeTLogMappingConfig.xml.
4. Create an Acme-specific Entity Reader configuration file; call this file AcmeTLogExtractConfig.xml.
5. If Acme exports the RTLog for Oracle Retail Sales Audit, the RTLogExportDaemonTechnician and RTLogExportDaemonThread can still be used to export the Acme Tlog formatted data. Just create another entry in StoreServerConduit.xml with a different technician and daemon name. This entry looks like the following:

```
<TECHNICIAN name="AcmeTLogExportDaemonTechnician"
            class="RTLogExportDaemonTechnician"
            package="com.extendyourstore.domain.manager.RTLog"
            export="Y">
  <PROPERTY propname="daemonClassName"

propvalue="com.extendyourstore.domain.manager.RTLog.RTLogExportDaemonThread" />
  <PROPERTY propname="daemonName"
            propvalue="AcmeTLogExportDaemon" />
  .
  .
  .
</TECHNICIAN>
```

6. Modify StoreServerConduit.xml to use AcmeTLogExtractConfig.xml, AcmeTLogFormatConfig.xml and AcmeTLogMappingConfig.xml when exporting the Acme TLog.

7. Determine the batch ID column to use for this process. By convention, DTM uses TR_TRN.ID_TLOG_BTCH, POSLog uses TR_TRN.ID_BTCH_ARCH, and RTLog uses ID_RTLOG_BTCH. If your system exports RTLog, you must override RTLogExportBatchGenerator.retrieveTransactionList() and RTLogDatabaseAdapter.postResults() to change the column your application uses.
8. Over the course of development add table names to AcmeTLogExtractConfig.xml, mapping information to AcmeTLogMappingConfig.xml. Write Acme-specific FieldMapperIfc and AccessorIfc classes.
9. It is necessary to create an Acme-specific implementation for the MappingResultIfc interface to hold the Acme transactional information. Call this class AcmeTLogMappingResult. This necessitates the creation of an Acme-specific EntityMappingObjectFactoryIfc class. Call this class AcmeEntityMappingObjectFactory.
10. It is necessary to create an Acme-specific implementation for the RecordFormatContentBuilderIfc to assemble the Acme-specific export records. Call this class AcmeTLogRecordFormatContentBuilder. This necessitates the creation of an Acme specific RecordFormatObjectFactoryIfc class called AcmeRecordFormatObjectFactory.
11. Modify StoreServerConduit.xml to use the AcmeEntityMappingObjectFactory and the AcmeRecordFormatObjectFactory when exporting the Acme TLog.

Exporting a Non-Fixed-Length Record Format

There are other styles of text besides fixed record length which have been used to transfer transactional information to the enterprise. For example: comma delimited, and tag and value. To support either of these you must complete all the steps in the previous section, as well as the following:

1. It is likely that you need additional information about the export file format. As a result you must add information to the Export Format Configuration file, and create an Acme-specific implementation of the RecordFormatConfiguratorIfc interface; call this class AcmeRecordFormatConfigurator.
2. The FieldFormat class formats its data based on the data type and generates a fixed length field. When all the fields in a record are aggregated, this creates a fixed length record. This class must be replaced by an Acme-specific implementation; call this class AcmeCommaDelimitedFieldFormat. It might also be necessary to create an Acme-specific implementation of RecordFormatIfc; call this class AcmeCommaDelimitedRecordFormat.
3. Modify AcmeRecordFormatObjectFactory to return AcmeRecordFormatConfigurator, AcmeCommaDelimitedFieldFormat, and AcmeCommaDelimitedRecordFormat.

Object Factories

Object factories provide system implementers with the means to replace base product implementations with classes that are more appropriate to their needs. The object factory classes appear as entries in configuration files, and often times a configuration file functions as an object factory. This section discusses the object factory aspects and the configuration aspects of the configuration files.

StoreServerConduit.xml

The Store Server Conduit file (<root>\applications\pos\config\conduit\StoreServerConduit.xml) defines at runtime the classes and configuration files that make up the managers and technicians in the Point-of-Service Store Server. One of the technicians it defines is the RTLogExportDaemonTechnician. Following are the classes the Store Server Conduit file defines for use when exporting the RTLog:

Table 5–1 Store Server Conduit File

Class Name	Interface Name	Description
RTLogExportDaemonTechnician (com.extendyourstore.domain.manager.rtlog)	RTLogExportDaemonTechnicianIfc (com.extendyourstore.domain.manager.rtlog)	Sets up the RTLog Export Process. The Dispatcher instantiates this class and then sets all the other parameters this object. It is also responsible for managing the batch regeneration process.
RTLogExportDaemonThread (com.extendyourstore.domain.manager.rtlog)	RTLogExportDaemonThreadIfc (com.extendyourstore.domain.manager.rtlog)	Sleeps for a configurable amount of time, then wakes up and initiates the export process.
RTLogDatabaseAdapter (com.extendyourstore.domain.manager.rtlog)	DatabaseEntityAdapterIfc (oracle.retail.stores.exportfile)	Provides access to the database for reading each transaction Entity. This particular implementation uses the DataManager/ DataTechnician to retrieve this information.
RTLogEncryptingOutputAdapter (oracle.retail.stores.exportfile.rtlog)	OutputAdapterIfc (oracle.retail.stores.exportfile)	Writes the RTLog file to the configured directory. This particular adapter encrypts the file as it writes the file to disk. There is another adapter, RTLogOutputAdapter, which writes the file in clear text.
RTLogEncryptionAdapter (com.extendyourstore.domain.manager.rtlog)	EncryptionAdapterIfc (oracle.retail.stores.exportfile)	Provides access to the mechanisms for decrypting values which are encrypted in the database.
ExportFileConfiguration (oracle.retail.stores.exportfile)	ExportFileConfigurationIfc (oracle.retail.stores.exportfile)	Contains much the of configuration information in the RTLogExportDaemonTechnician; the technician passes this object to the daemon, which passes it to the batch generator which passes it to the export file generator.
RTLogExportFileResultAuditLog (com.extendyourstore.domain.manager.rtlog)	ExportFileResultAuditLogIfc (oracle.retail.stores.exportfile)	Formats the export result information for logging.
EntityMappingObjectFactory (oracle.retail.stores.exportfile)	EntityMappingObjectFactoryIfc (oracle.retail.stores.exportfile)	Instantiates the classes used to map the database Entity to the export file format.
RecordFormatObjectFactory (oracle.retail.stores.exportfile)	RecordFormatObjectFactoryIfc (oracle.retail.stores.exportfile)	Instantiates the classes used to setup and generate the export the file format.
ExtractorObjectFactory (com.oracle.xmlreplication)	ExtractorObjectFactoryIfc (com.oracle.xmlreplication)	Instantiates the classes used to generate the database Entity.
RTLogCurrencyAdapter (com.extendyourstore.domain.manager.rtlog)	CurrencyAdapterIfc (oracle.retail.stores.exportfile)	Provides currency services.

DomainObjectFactory

The DomainObjectFactory instantiates the RTLogExportBatchGeneratorIfc class. The RTLogExportBatchGenerator builds the WorkUnit (the list of transactions to export) and calls the WorkUnitController (ExportFileGenerator).

RTLogExportBatchGenerator also instantiates the ExportFileGeneratorIfc and the WorkUnitIfc. If you need a different implementation of either class, create a new implementation of RTLogExportBatchGenerator.

ExtractorObjectFactory

The ExtractorObjectFactory instantiates the classes that generate the database Entity class.

One item of note is that the application gains access to this factory through a singleton called ReplicationObjectFactoryContainer. All changes made to these classes must work for both DTM and Export File generation.

EntityMappingObjectFactory

The following table is a list of the classes this factory instantiates:

Table 5–2 EntityMappingObjectFactory Classes

Class Name	Interface Name	Description
MappingCatalogConfigurator (oracle.retail.stores.exportfile.mapper)	MappingCatalogConfiguratorIfc (oracle.retail.stores.exportfile.mapper)	Reads the mapping configuration file and builds an EntityMappingCatalogIfc object.
EntityMappingCatalog (oracle.retail.stores.exportfile.mapper)	EntityMappingCatalogIfc (oracle.retail.stores.exportfile.mapper)	Holds the information that describes the relationship between the tables and columns in the database to the records and fields in the export file. It contains a list of TableMaps and a map of Accessors.
TableMap (oracle.retail.stores.exportfile.mapper)	TableMapIfc (oracle.retail.stores.exportfile.mapper)	Contains a list of ColumnMaps associated with a table.
ColumnMap (oracle.retail.stores.exportfile.mapper)	ColumnMapIfc (oracle.retail.stores.exportfile.mapper)	Describes the relationship between a column and a field in a specific export record. It can contain a ValueMapping Hashmap and/or FieldMapper class to perform more complex mapping actions.
EntityMapper (oracle.retail.stores.exportfile.mapper)	EntityMapperIfc (oracle.retail.stores.exportfile.mapper)	Controls the mapping process. It stores the result in the MappingResultIfc object.
RTLogMappingResult (oracle.retail.stores.exportfile.rtlog)	MappingResultIfc (oracle.retail.stores.exportfile.mapper)	Contains the result of Mapping an Entity to the Export File Format.

RTLogMappingConfig.xml

This configuration file is a factory for FieldMapperIfc and AccessorIfc classes.

The simplest mapping occurs when a value goes directly from a column to a field. However, many times the mapping between a column and a field is more complex. If code is required, the configuration file calls out a FieldMapperIfc class to perform this mapping task. A FieldMapperIfc is associated with a particular table/column record/field mapping.

The values in a particular record are built up by processing of each individual ColumnMapIfc objects. There is no guarantee that all the data for a particular export record resides in a single row in the database. In fact it is unlikely. For example, a row from the Tender Line Item Table supplies the tender amount, but a row from the Credit Debit Tender Line Item Table supplies authorization information. Much processing can take place in between the time that the application has access to each of these rows.

An AccessorIfc object knows how to locate a particular existing “working” export record in the MappingResultIfc object. If a record is not available, the AccessorIfc creates a new one and store it in the MappingResultIfc object.

RecordFormatObjectFactory

Following is a list of the classes this factory instantiates:

Table 5–3 RecordFormatObjectFactory Classes

Class Name	Interface Name	Description
FieldFormat (oracle.retail.stores.exportfile.formater)	FieldFormatIfc (oracle.retail.stores.exportfile.formater)	Contains the attributes associated with a field including name, value, starting index, length, and data type.
RecordFormat (oracle.retail.stores.exportfile.formater)	RecordFormatIfc (oracle.retail.stores.exportfile.formater)	Contains a list of FieldFormatIfc objects.
RecordFormatCatalog (oracle.retail.stores.exportfile.formater)	RecordFormatCatalogIfc (oracle.retail.stores.exportfile.formater)	Contains a list of RecordFormatIfc objects.
RecordFormatConfigurator (oracle.retail.stores.exportfile.formater)	RecordFormatConfiguratorIfc (oracle.retail.stores.exportfile.formater)	Reads the format configuration file and builds a RecordFormatCatalogIfc object.
RTLogRecordFormatContentBuilder (oracle.retail.stores.exportfile.rtlog)	RecordFormatContentBuilderIfc (oracle.retail.stores.exportfile.formater)	Converts MappingResultsIfc object into the text that is written to the export file.
RTLogItemContainedRecords (oracle.retail.stores.exportfile.rtlog)	ContainedRecordsIfc (oracle.retail.stores.exportfile.formater)	A list of records, such as discounts, that are a part of the item information.
RTLogTransactionContainedRecords (oracle.retail.stores.exportfile.rtlog)	ContainedRecordsIfc (oracle.retail.stores.exportfile.formater)	A list of records, such as header total records, that are part of a transaction.

Configuration

Each of the configuration files used by this feature (Store Server Conduit, Entity Reader Configuration, Mapping Configuration, and Record Format Configuration) has already been referred to in this document. This section describes them in more detail.

The Store Server Conduit File

The Store Server Conduit file (<root>\applications\pos\config\conduit\StoreServerConduit.xml) defines the following settings for the RTLog Export process.

Table 5–4 Store Server Conduit File

Setting Name	Installed Product Value	Description
sleepInterval	600 (seconds)	The length of time between each execution of the RTLog export process.
exportDirectoryName	For example, POSLog	The directory where the RTLog is placed.
formatConfigurationFileName	../config/rtlog/RTLogFormat.xml	The relative or absolute path of the Export Format configuration file.
entityReaderConfigurationFileName	../config/rtlog/RTLogExtractConfig.xml	The relative or absolute path of the Entity Reader configuration file.
entityMappingConfigurationFileName	../config/rtlog/RTLogMappingConfig.xml	The relative or absolute path of the Mapping configuration file.
maximumTransactionsToExport	-1	The maximum number of transactions that should be exported to single RTLog file. The value -1 indicates there is no limit on the maximum number.

The Export Format Configuration file

The export format configuration file describes each of the export record types. For example, the RTLog specifies the following records:

- File Header
- File Tail
- Transaction Header
- Transaction Tail
- Transaction Item
- Item Discount
- Transaction Tax
- Transaction Tender

The following is a snippet from RTLogFormat.xml:

```
<?xml version="1.0"?>
<RECORD_FORMATS ... >
  <COMMENT>This file defines the format of the Oracle Retail Sales Audit
RTLog</COMMENT>
  <RECORD_FORMAT_VERSION version="V.12.0.5"/>
  <RECORD_FORMAT name="FileHeader">
    <FIELD_FORMAT name="FileRecordDescriptor" type="char" length="5"
value="FHEAD"/>
    <FIELD_FORMAT name="FileLineIdentifier" type="integer"
length="10"/>
    <FIELD_FORMAT name="FileType" type="char" length="4" value="RTL"/>
    <FIELD_FORMAT name="FileCreateDate" type="datetime" length="14"/>
    <FIELD_FORMAT name="BusinessDate" type="date" length="8"/>
    <FIELD_FORMAT name="LocationNumber" type="char" length="10"/>
    <FIELD_FORMAT name="ReferenceNumber" type="char" length="30"
value=" "/>
  </RECORD_FORMAT>
  .
  .
  .
</RECORD_FORMATS>
```

This snippet shows one Record definition (the File Header) composed of seven fields of various types, lengths and default values.

The Entity Reader Configuration File

This file defines tables that Entity Reader reads.

The Mapping Configuration File

This file describes the relationship between the tables and columns in the database and the records and fields in the export format. The following is a snippet from RTLogMappingConfig.xml:

```
<?xml version="1.0"?>
<ENTITY_MAPPER ... >
  <COMMENT>This is a configuration file for the Point-of-Service Transaction to
RTLog Mapping</COMMENT>
  <TABLE table="TR_TRN">
    <MAP column="DC_DY_BSN" record="FileHeader" field="BusinessDate"

fieldMapper="oracle.retail.stores.exportfile.rtlog.fieldmappers.BusinessDateMapper
"/>
    <MAP column="ID_STR_RT" record="FileHeader" field="LocationNumber"

fieldMapper="oracle.retail.stores.exportfile.rtlog.fieldmappers.StoreNumberMapper"
/>
    <MAP column="TS_TRN_END" record="TransactionHeader"
field="RegisterTransactionDate"

fieldMapper="oracle.retail.stores.exportfile.rtlog.fieldmappers.DateTimeMapper"/>
    .
    .
    .
    <MAP column="TY_TRN" record="TransactionHeader" field="TransactionType"
mappingStrategyOrder="FieldMapperThenValueMapping"
```

```
fieldMapper="oracle.retail.stores.exportfile.rtlog.fieldmappers.ExportItemsAndTaxS
tatusMapper">
    <VALUE_MAPPINGS handleNotFound="error">
        <VALUE_MAPPING DatabaseValue="1" RecordValue="SALE"/>
        <VALUE_MAPPING DatabaseValue="2" RecordValue="RETURN"/>
        <VALUE_MAPPING DatabaseValue="3" RecordValue="PVOID"/>
        .
        .
        .
    </VALUE_MAPPINGS>
</MAP>
.
.
.
</TABLE>
.
.
.
    <ACCESSOR record="FileHeader"

class="oracle.retail.stores.exportfile.rtlog.accessors.AccessFileHeader"/>
    <ACCESSOR record="TransactionHeader"

class="oracle.retail.stores.exportfile.rtlog.accessors.AccessTransactionHeader"/>
.
.
.
</ENTITY_MAPPER>
```

Looking at this snippet, it is easy to see that the column TR_TRN.DC_DY_BSN maps to the BusinessDate field in the FileHeader record using the BusinessDateMapper class to format the data.

Also note that application uses a VALUE_MAPPINGS element to transform the value from the column TR_TRN.TY_TRN to equivalent value in the TransactionType field in the TransactionHeader record.

Development and Testing Tools

There are a number of tools that were developed during the course of this project that are helpful when extending this subsystem.

Classes

The following classes are all located at <root>\modules\exportfile\src\oracle\retail\stores\exportfile\utility:

Table 5–5 Exportfile Utility Classes

Class Name	Description
ExportTestDriver	<p>This class is a test harness that can be used to develop the configuration files, FieldMapperIfc and AccessorIfc classes in isolation from the rest of the application. It uses the classes DatabaseEntityAdapterTest, EncryptionAdapterTest, CurrencyAdapterTest, OutputAdapterTest and ExportFileResultAuditLogTest to emulate system specific adapters.</p> <p>An Eclipse-run configuration for this class should run out of the exportfile project. The classpath should include the domain, foundation-client, foundation-server, common, utility, foundation-shared, clientinterfaces, datareplication projects and /thirdparty/apache-ant-1.6.2/lib/xml-apis.jar, /thirdparty/apache-ant-1.6.2/lib/xercesImpl.jar, and /thirdparty/apache/log4j-1.2.8.jar. It should also include the JDBC jar(s) for the database you are using.</p> <p>You might need to modify this class to use the appropriate JDBC driver, username, password and transaction IDs.</p>
FileDecryptionUtility	<p>By default the application (not the test harness) generates encrypted files. This class reads all the encrypted files from a target directory, decrypts them, and write them to a single target file. The main() method has three command line parameters:</p> <ul style="list-style-type: none"> EncryptedDirectoryName - the pathname of the directory of *.ENC files DecryptedFileName - the pathname of the decrypted file <p>For more information about encryption, see Oracle Retail Strategic Store Solutions RTLog Files in Chapter 1.</p>
RTLOGReportDriver	<p>This class reads an export format configuration file and an export log file then generates a report file (rtlog_rpt.txt) to the current directory. This saves a lot effort when trying to determine if an export file has the correct data in it. The main() method has two command line parameters:</p> <ul style="list-style-type: none"> ExportFileName - full/relative path pathname of the export file. XMLFormatFileName - full/relative path pathname of the format file.

Executables in the bin Directory

The following BAT files are all located at <root>\modules\exportfile\bin:

Table 5–6 bin Directory BAT Files

Class Name	Description
setenv.bat	Sets up the classpath
RTLogFileDecryption.bat	Executes FileDecryptionUtility.class; it points at the bin\POSLog directory in the default installation, writes the decrypted records to RTLOG.DEC, and uses the default encryption key.
RTLogReport.bat	Executes RTLOGReportDriver.class; it reads RTLOG.DEC, and uses to the export format file ..\config\RTLogFormat.xml.

Extending the RTLog Encryption model

The requirements for this release call for the RTLog to be encrypted; however, the solution to encryption key sharing between Strategic Stores Solutions and Oracle Retail Sales Audit applications is not provided with the product. Therefore, Strategic Stores Solutions and Oracle Retail Sales Audit use a single known encryption key, cipher, and set of encryption parameters. Oracle Retail Point-of-Service uses this predetermined information to encrypt the file, and Oracle Retail Sales Audit uses it to decrypt the file.

In many cases this will not be sufficient for a specific implementation; some retailers will choose to deploy third party key store/encryption technology. In this case the implementation team must provide custom code for both Oracle Retail Point-of-Service and Oracle Retail Sales Audit.

In Oracle Retail Point-of-Service there are two classes that appear to be candidates for assisting in this process:

- `RTLogEncryptionAdapter` – provides a service which the application uses to decrypt credit card numbers. Oracle Retail has its own internal approach to encrypting credit card numbers before storing them in the database. The primary purpose of this adapter is to decrypt encrypted credit card numbers.
- `RTLogEncryptingOutputAdapter` – performs the encryption on the RTLog using the simple approach outlined above.

The implementation team should provide a class that implements the `OutputAdapterIfc` interface and interacts with the third-party key store/encryption application to encrypt the contents of the RTLog.

Since the decryption key information is well known to the current applications, no provision has been made to communicate this information to the Oracle Retail Sales Audit application in the RTLog. In a third-party key store scenario the key identifier must be included with each RTLog file. The key identifier must be provided to the key store application in order to obtain the actual key used to perform the decryption. The key identifier could be included as part of the file name or pre-pended in the clear to the encrypted data within the RTLog file.

The supplier of the third party key store application will supply details for how to use their application.

Known Issues and Troubleshooting

DepartmentDefaultTaxGroup

When integrated with Oracle Retail Merchandising System, the `PreloadData > POSDepartment > DepartmentDefaultTaxGroup` field in the `MerchandiseHierarchyImport` is defaulted to 0 (zero). It is the responsibility of the implementation team to update this value in the bundle with a real `TaxGroup` ID for the items in question before the bundle reaches Strategic Store Solutions. Otherwise, a primary key violation might occur if zero is not an actual `TaxGroup` ID in the UDM.

Character Restrictions for UOMs

Retailers are restricted to only creating and using items with 2 character UOMs (Unit of Measure) as part of this integration.

Merchandise Operations Management transforms EA (Each) to UN (Unit) for the UOM in Item extracts to Strategic Store Solutions.

Strategic Store Solutions does not transform any other UOM in RTLogs to Merchandise Operations Management.

Oracle Retail Point-of-Service translates UN back to EA for the RTLog.

POSlog

For more information about the POSlog, see "POSlog Import Service" in the *Oracle Retail Central Office Operations Guide Release* and in the *Oracle Retail Back Office Operations Guide Release*.

Preload Section of ItemImport

Data in the Preload section of `ItemImport` is treated as an UPS which stands for **Upsert**. DIMP tries to Update data and if fails to update, then it Inserts data.

UTF-8

UTF-8 is a required character set for the database. DIMP supports multi-byte characters in the XML and puts this data into the database as UTF-8 character set.

Transaction Level Items

Oracle Retail Merchandising System extracts transaction-level items only.

Need To Escape Special Characters In XML File

Special characters in an XML file, such as <, >, & and so forth, must be escaped. For more information, see the following:

<http://www.w3.org/TR/REC-xml/>

Geocode Tag Missing For Store

The geocode is not sent to Oracle Retail Point-of-Service.

See *Oracle Retail Merchandising System Operations Guide - Batch Overviews and Designs - Volume 1 Release 12.0.5N* for more information.

Missing Encryption Key For Saencrypt.pc

It is assumed that clients will generate their key. So a key file is not part of the release. Strategic Store Solutions generates the key (file) and Oracle Retail Merchandising System reads the key from the file.

For more information about keys see [Extending the RTLog Encryption model](#) in chapter 5.

Clearance Pricing

Strategic Store Solutions does not support Clearance pricing.

Oracle Retail Price Management Price Promotion endDateTime in Pricing Import XSD

Price Promotions imported through DIMP that have no specified end dates will default to December 31, 2099.

Data Import Failure

If an individual batch fails during a data import, there is no retry mechanism to import only the batch that failed. It is left to an administrator to resolve the issue that caused the batch failure and recreate the data of the failed batch. If the integrity of the incoming data cannot be guaranteed as Data Import expects, it is possible to avoid rolling back valid data within failed batches by adjusting the size of the import batches from the default size of 1000 down to 1 by editing the `spring.properties` file and restarting the application server.

Note: This resolution will have a negative impact on performance.

Integration with Oracle Retail Sales Audit

The integration with Oracle Retail Sales Audit requires that tills are only opened and closed once per business day.

Total ID in the RTLog

The same Total ID is used for more than one till. This causes the following error message in Oracle Retail Sales Audit:

```
Duplicate declaration: this total id has already been used by another transaction  
xxxxxx.
```

Debit Card Problem with Oracle Retail Sales Audit

If an unrecognized debit card or credit card is used for tender, an entry is not made in the RTLog. Oracle Retail Sales Audit is then unable to process the transaction. The transaction needs to be reconciled manually.

Data Import Field Width Maximums

All VARCHAR(255) sizes were changed to VARCHAR(250) to match MOM sizes

Price Change Applied Before Start Date

Oracle Retail Point-of-Service only supports a physical deployment model where the Point-of-Service clients and in-store server are set to the same system time as the store database. If the in-store server and database are set to different system times, that is, the clock is not set correctly or they exist in different time zones, it is possible that items will ring with incorrect prices as prices changes and discount rules applied at the Point-of-Service client rely on the system time of the store database.

The Sub-Transaction Type in the RTLog Entry for Closing A Register is Not Correct

The mapping value for 9 in the TY_TRN column of the TR_TRN table is changed to CRGRC.

When a Gift Certificate is Issued, the Item Type and Voucher Number Are Not Correct in the RTLog

For gift certificates that are issued or voided, the voucher number is hard-coded to zero and the TITEM record contains GCN as the ItemType.

DiscountReferenceNumber Field Format in the RTLog is Incorrect

The format of the DiscountReferenceNumber field in the RTLog for a sale transaction that includes a price promotion is incorrect. The field is right-justified and padded with zeroes.

The method `getIntegerFormattedValue(String value, int len)` is updated to correctly set the field as left-justified and padded with blanks.

If the Refund Amount for a Deleted Layaway is Zero, A TTEND Record is Not Captured in the RTLog

The code is corrected to capture the TTEND record when the refund amount is **zero**.

For a Layaway Complete Transaction, the Voucher Number is Not Included in the TTEND Record

`LayawayPrepaidTenderAmountMapper.map()` is modified to map the voucher number for TenderType **vouch**.

UnitRetail Amount Does Not Match the OriginalUnitRetail Amount in the TITEM Record

For a price promotion created in Oracle Retail Price Management, the UnitRetail amount does not match the OriginalUnitRetail amount in the TITEM record in the RTLog.

`ItemPriceMapper.map()` is modified to call `getPromotionDiscount()` to get the promotion discount for the promotion line item and apply it to the item price.

In `PriceChange.java`, methods `getNewPrice()` and `isNewPriceBigger()` are modified to set the new price to the permanent price of the item.

ItemType and Voucher_no Fields are Not Correct in the TITEM Record in the RTLog

When a store credit is redeemed, the ItemType and Voucher_no fields are not correct in the TITEM record in the RTLog.

In the TITEM record, ItemType is set to **GCN** and Voucher_no is set to the store credit number.

The Reason Code is Missing in the THEAD Record in the RTLog

For a partial pickup order transaction, the reason code is missing in the THEAD record in the RTLog.

A new value mapping is added for the ReasonCode field in column TY_TRN of the TR_TRN table for a partial pickup order transaction.

The Vouch_no Field is Empty in the THEAD Record in the RTLog

For a partial pickup order transaction, the Vouch_no field is empty in the THEAD record in the RTLog.

The order number is used as the voucher number. Since there is only one deposit for a special order, the order number uniquely identifies the order.

Item Import Fails Because an Item Already Exists

If an item already exists in the database and an item import is done, the import fails because an item already exists.

UPSERT is used for updates to the CO_EV, MA_PRC_ITM, CO_EV_MNT, MA_ITM_PRN_PRC_ITM, and TR_CHN_PRN_PRC tables.

The Cost Field for an Item is Updated from the Import Instead of Using 0.00

After an item import from Oracle Retail Merchandising System, the cost field for an item is updated from the import instead of using 0.00.

loadSupplierItemCatalogBaseCoseBreakDTO is updated to set the cost field to 0.00.

Empty Item Classes Lists for DIMP

In Oracle Retail Back Office, **Available Classes** and **Assigned Classes** lists are empty for an item.

The menu is empty in an Oracle Retail Merchandising Products-integrated environment. The retailer must define these.

Discountable Attribute from Oracle Retail Merchandising System

The Discountable attribute for an item imported from Oracle Retail Merchandising System is always set to **true**.

RegistryEligible Field

The RegistryEligible field is hardcoded with the value true in Oracle Retail Merchandising System extracts.

Authorized for Sale

The Oracle Retail Back Office data field **Authorized for Sale** is mapped to the status of an item at a store (item_loc). If the item is **Active** at that location, then **true** is extracted.

Other statuses, such as **Discontinued** and **Delete** cause the value **false** to be extracted.

CatchWeight Item in RTLog

Oracle Retail Point-of-Service does not support the CatchWeight attribute for items, so the value of the field will always be set to false by Oracle Retail Point-of-Service.

Customer-Specific Pricing in Pricing Data Import

Customer-specific pricing on items is not currently supported in the Pricing data import.

Item-Level Coupon Number is Not Captured in the RTLog

If a transaction includes an item-level coupon, the coupon number is not captured in the RTLog.

In RTLogMapping.xml, a mapping is added for the field ID_DSC_REF for table CO_MDFR_RTL_PRC.

In DiscountRuleImportUtil.java, the method getPriceDerivagionRuleDTO is updated to set the reason code of price derivation to -1 instead of getting the value from the pricing rule.

ReferenceNumber1 Field Does Not Contain Number of RTLogs Sent to Oracle Retail Sales Audit

When a store is closed, the ReferenceNumber1 field does not contain the number of RTLogs sent to Oracle Retail Sales Audit during the business day.

The number of RTLogs sent for a business day is maintained in the table CA_DY_BSN_BTCH. But the record was updated only after creating the RTLog. If there was only one RTLog to be generated for one day, this table was empty and therefore no number was sent.

The code is updated to write the number of RTLogs based on the transaction type. If the transaction type is store close, the number of files is obtained from the table and the number is incremented by one. If there is no record, 1 is sent.

Temporary Price Change Loaded Using DIMP Not Going Into Effect

When a temporary price change is loaded using DIMP, the price change is not going into effect.

In the method getNextIDs(), code is added to get the event ID from the import instead of starting the event ID at **zero**.

Unexpected Exception Occurs When Trying to Create a Store Group

An unexpected exception occurs when trying to create a store group after the store hierarchy is imported using DIMP.

The data load sets the functionID field to **zero** in the CO_STRG_FNC, CO_STRGP_LV, and CO_STRGP tables. When importing the store hierarchy, data is not deleted from tables where the functionID field is equal to **zero**.

Tax Import Troubleshooting

Issue : TX_ASGMT table is not populated after running the Calculate_TaxFactor.bat

Root Cause: If the store ID is passed as a parameter to the .bat and there are no matching records for it in the tax file, then the TX_ASGMT will not be populated.

Issue : TXBL_FCT and TX_FCT are '0'

Root Cause: If there no matching ID_CTGY_TX from AS_ITM in the tax flat file for a given store, then the TXBL_FCT and TX_FCT will be calculated as 0 or if the tax applied on is different than retail, the TXBL_FCT and TX_FCT will be calculated as 0.

Issue : Message 2100 not found; No message file for Product=RDBMS

Root Cause: Tax data file is not found in the “data” folder or ORACLE_HOME is pointing to Oracle Applcation server.

Figure 6–1 Tax Import Troubleshooting

```

C:\WINDOWS\system32\cmd.exe - Calculate_TaxFactor.bat 4000 orbo/orbo@orcl
C:\OracleRetailStore\Server\pos\bin\L10N\IN\Batches\TaxCalculation-Batch>Calculate_TaxFactor.bat 4000 orbo/orbo@orcl
*****
Loading data into TAX_ASSIGNMENT
*****
Upload tax assignment for STORE_ID= 4000 :
The process cannot access the file because it is being used by another process.
0 file(s) moved.
1 file(s) copied.
Message 2100 not found; No message file for product=RDBMS, facility=ULMessage 2100 not found; No message file for product=RDBMS, facility=ULERROR:
ORA-12154: TNS:could not resolve the connect identifier specified

```

Employee Information

Currently, employee information must be presented in a specific file format for consumption by Central Office.

Implementation teams need to be aware of this file format.

Tax and Employee files each have an XML Schema Definition just like other DIMPs:

Example 6–1 Employee File XML Schema Definition

```
<?xml version="1.0" encoding="windows-1252" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">

    <!-- $Log:$ -->

    <xs:annotation><xs:documentation>
        Employee Import Schema. Copyright 2006 Oracle. All rights reserved.
    </xs:documentation></xs:annotation>

    <xs:element name="EmployeeImport" type="EmployeeImport">
        <xs:annotation><xs:documentation>
            Top-level element holding a collection of Employee elements.
        </xs:documentation></xs:annotation>
    </xs:element>

    <xs:complexType name="EmployeeImport">
        <xs:sequence>
            <xs:element name="Employee" type="EmployeeType" minOccurs="1"
maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="FillType" type="FillType" use="required"/>
        <xs:attribute name="CreationDate" type="xs:dateTime"/>
        <xs:attribute name="ExpirationDate" type="xs:dateTime"/>
        <xs:attribute name="Version" type="xs:string"/>
        <xs:attribute name="Priority" type="xs:int"/>
        <xs:attribute name="Batch" type="xs:int"/>
    </xs:complexType>

    <xs:complexType name="EmployeeType">
        <xs:annotation><xs:documentation>
            Represents a single employee's information.
        </xs:documentation></xs:annotation>
        <xs:sequence>
            <xs:element name="ChangeType" type="ChangeType" default="ADD"
minOccurs="1" maxOccurs="1" />
            <xs:element name="EmployeeID" type="ID" minOccurs="1" maxOccurs="1" />
            <xs:element name="EmployeeFirstName" type="xs:string" minOccurs="0"
maxOccurs="1" />
            <xs:element name="EmployeeLastName" type="xs:string" minOccurs="0"
maxOccurs="1" />
            <xs:element name="EmployeeMiddleName" type="xs:string" minOccurs="0"
maxOccurs="1" />
            <xs:element name="EmployeeFullName" type="xs:string" minOccurs="0"
maxOccurs="1" />
            <xs:element name="EmployeeSSN" type="SSN" minOccurs="0" maxOccurs="1"
/>
            <xs:element name="EmployeeRole" type="xs:string" minOccurs="0"
maxOccurs="1" />
            <xs:element name="PartyID" type="xs:int" minOccurs="0" maxOccurs="1"
/>
            <xs:element name="StatusCode" type="StatusCode" minOccurs="0"
maxOccurs="1" />
            <xs:element name="Locale" type="ID" minOccurs="0" maxOccurs="1" />
            <xs:element name="EmployeeAccess" type="EmployeeAccess" minOccurs="0"
maxOccurs="1" />
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

```

        <xs:element name="EmployeeType" type="StatusCode">
            <xs:annotation><xs:documentation>
                0 means 'Standard' employee, 1 means Temporary employee
            </xs:documentation></xs:annotation>
        </xs:element>
        <xs:element name="NumberDaysValid" type="xs:int" minOccurs="0"
maxOccurs="1">
            <xs:annotation><xs:documentation>
                Only applies to temporary employee
            </xs:documentation></xs:annotation>
        </xs:element>
        <xs:element name="TempEmployeeExpirationDate" type="xs:date"
minOccurs="0" maxOccurs="1">
            <xs:annotation><xs:documentation>
                Only applies to temporary employee
            </xs:documentation></xs:annotation>
        </xs:element>
        <xs:element name="EmployeeStoreOrHierarchyAssn"
type="EmployeeStoreOrHierarchyAssn" minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
</xs:complexType>

<xs:simpleType name="ID">
    <xs:restriction base="xs:string">
        <xs:maxLength value="10" />
    </xs:restriction>
</xs:simpleType>

<xs:simpleType name="SSN">
    <xs:restriction base="xs:string">
        <xs:maxLength value="9" />
    </xs:restriction>
</xs:simpleType>

<xs:simpleType name="StatusCode">
    <xs:restriction base="xs:string">
        <xs:enumeration value="0" />
        <xs:enumeration value="1" />
    </xs:restriction>
</xs:simpleType>

<xs:complexType name="EmployeeAccess">
    <xs:annotation><xs:documentation>
        Holds all information regarding access to the system.
    </xs:documentation></xs:annotation>
    <xs:sequence>
        <xs:element name="EmployeeLoginID" type="xs:string" />
        <xs:element name="AccessPassword" type="xs:string" />
        <xs:element name="WorkGroupID" type="xs:int" />
        <xs:element name="EmployeeAltID" type="xs:string" minOccurs="0"
maxOccurs="1" />
        <xs:element name="NewPasswordRequired" type="xs:boolean" />
        <xs:element name="PasswordCreationDate" type="xs:dateTime" />
        <xs:element name="PasswordHistory" type="PasswordHistory"
minOccurs="0" maxOccurs="1">
            </xs:element>
    </xs:sequence>
</xs:complexType>

<xs:complexType name="PasswordHistory">

```

```
<xs:sequence>
  <xs:element name="PasswordHistoryEntry" type="PasswordHistoryEntry"
minOccurs="1" maxOccurs="unbounded" />
</xs:sequence>
</xs:complexType>

<xs:complexType name="PasswordHistoryEntry">
  <xs:annotation><xs:documentation>
    Holds a single password history entry.
  </xs:documentation></xs:annotation>
  <xs:sequence>
    <xs:element name="PasswordCreationDate" type="xs:dateTime" />
    <xs:element name="AccessPassword" type="xs:string" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="EmployeeStoreOrHierarchyAssn">
  <xs:annotation><xs:documentation>
    Holds an employee association to a store and/or a hierarchy node.
    Generally, only one of the
    enclosed elements is provided; however, there may be cases where an
    employee needs both a store
    association and a hierarchy association, so a sequence with optional
    elements is used instead of
    a choice.
  </xs:documentation></xs:annotation>
  <xs:sequence>
    <xs:element name="EmployeeStoreID" type="RetailStoreId" minOccurs="0"
maxOccurs="1" />
    <xs:element name="EmployeeHierarchyAssn" type="EmployeeHierarchyAssn"
minOccurs="0" maxOccurs="1" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="EmployeeHierarchyAssn">
  <xs:sequence>
    <xs:element name="NodeID" type="xs:string" minOccurs="1"
maxOccurs="1" />
    <xs:element name="NodeType" type="xs:string" minOccurs="1"
maxOccurs="1" />
    <xs:element name="StoreGroupFunctionID" type="xs:int" minOccurs="1"
maxOccurs="1" />
  </xs:sequence>
</xs:complexType>

<xs:simpleType name="ChangeType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="ADD" />
    <xs:enumeration value="UPD" />
    <xs:enumeration value="DEL" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="RetailStoreId">
  <xs:annotation><xs:documentation>
    Store Id's can only be five characters long and preferably only
    numerals.
  </xs:documentation></xs:annotation>
  <xs:restriction base="xs:string">
    <xs:minLength value="1"></xs:minLength>
```

```
        <xs:maxLength value="5"></xs:maxLength>
    </xs:restriction>
</xs:simpleType>

<xs:simpleType name="FillType">
    <xs:restriction base="xs:string">
        <xs:enumeration value="KillAndFill"/>
        <xs:enumeration value="FullIncremental"/>
    </xs:restriction>
</xs:simpleType>

</xs:schema>
```

Glossary

L10N

Localization.

Batch

A collection of data operations that are processed at one time. The size is determined by a configurable parameter.

Bundle

A collection of import files, one file per data type, stored as a compressed file containing a manifest. It is expected that the retailer or implementation team is responsible for packaging and delivering to the Store the bundle along with manifest for all data feeds to the Store

Corporate

Used interchangeably with *enterprise*. The enterprise environment of the retailer where enterprise applications are deployed. Oracle Retail Central Office is deployed in the enterprise.

Data Access Object (DAO)

A class that can retrieve and persist data to and from a data source.

Data Distribution Infrastructure (DDI)

The infrastructure and application components that are responsible for distributing seed data from enterprise applications to Store applications, ODS at Corporate (or enterprise), and Store Database at the stores.

Data Transfer Object (DTO)

A class that contains data records from a received payload. The DTO's attributes are populated with the parsed data.

DIMP

Data Import

Incremental

There are two types of update operation, full incremental and delta incremental. Full incremental assumes that all the fields for a data type are supplied in the XML. A delta incremental import contains only the fields that are being changed.

ISP

In-Store-Processor

J2EE

Java 2 Enterprise Edition is a set of APIs designed to support tier 1 type business models.

Java Database Connectivity (JDBC)

An API used to communicate with relational databases.

Kill And Fill

Kill And Fill refers to a data operation where all the existing data in a table is deleted (kill) and then replaced with new data (fill).

Manifest

A file within a bundle that lists the data files in the bundle and their interdependencies.

Operational Data Store (ODS)

The corporate data repository that services Oracle Retail Central Office.

ORBO

Oracle Retail Back Office

ORCO

Oracle Retail Central Office

ORLT

Oracle Retail Labels and Tags

ORMPOS

Oracle Retail Mobile Point of Service

ORPOS

Oracle Retail Point of Service

ORRM

Oracle Retail Returns Management

ReSA

Oracle Retail Sales Audit

RMS

Oracle Retail Merchandising System

RPM

Oracle Retail Price Management

RTLog

Retail Transaction Log

Seed Data

Seed Data is defined as data that must be supplied by our customers in order for our applications to fully use all features and functions which the customer decided to enable.

SIM

Oracle Retail Store Inventory Management

Store Applications

Oracle Retail applications that run in the store environment. This includes:

- Oracle Retail Back Office
- Oracle Retail Point-of-Service
- Oracle Retail Mobile Point-of-Service
- Oracle Retail Strategic Store Solutions
- Oracle Retail Labels and Tags
- Oracle Retail Store Inventory Management
- Oracle Retail Central Office
- Oracle Retail Returns Management.

It must be noted that even though Oracle Retail Central Office runs in the corporate environment, it is classified as a store application.

Store Database

The data repository for store applications.

Strategic Store Solutions

The Oracle Retail business unit that assumes responsibility for applications running in the Store environment.

Appendix: Discount Rules – Any or All

Previous versions of Oracle Retail Point-of-Service treats a collection of Buy items (also called sources) in a DiscountRule, as well as the collection of Get items (also called targets), as **All required** before that rule's criteria are considered satisfied. This means that all sources and targets of a rule must exist in a purchase before that rule is applied to the transaction. Oracle Retail Price Management enables Any item in a collection of sources or Any item in a collection of targets to exist in the transaction for the discount to take affect.

Current enhancements to Stores behavior enable either **Any** or **All**. This provides for much tighter integration to Oracle retail Price Management.

During import of a DiscountRule, a quantity must be specified when an Any qualifier is given for either the source or target. These two new quantities are added as columns to the PriceDerivationRule (RU_PRDV) table:

```
QU_AN_SRC
QU_AN_TGT
```

When the Any quantities for source or target are **zero** or less, Oracle Retail Point-of-Service considers this to mean that all sources or targets are required. Otherwise, when the Any quantize for source or target are **one** or greater, that quantity is the minimum required for the source or target to activate the discount.

When left unspecified during import, sources and targets are imported as **Any 1**.

When a discount rule contains the **Any** option, and the number of available choices of sources or targets exceed the any quantity, the system must determine how to sort the items in order to know which items participate in the discount rule. The sorting algorithm varies based on the discount rule and whether or not the items participate as both sources and targets within that rule (that is, whether the sources are discounted):

- When the same items participate as both sources items and targets (that is, whether the sources receive the discount), the system sorts the source items from most expensive to least expensive to determine which source items should participate in the discount rule.
- When the same items do not participate as both sources and targets, the system sorts the source items from least expensive to most expensive and chooses the first options until the any quantity is met.
- Targets are always sorted and chosen from most expensive to least expensive and chosen in order, unless the rule specifies "BuyNofXgetLowestPricedXatZ%off", in which case the least expensive target items are chosen first.

Appendix: XSD Files and Data Element Definition Tables

This chapter provides the XML Schema Definitions (XSD) of the following Data Import data types:

- [Employee Import](#)
- [Item Import](#)
- [Merchandise Hierarchy Import](#)
- [Pricing Import](#)
- [Store Hierarchy Import](#)

The XSD defines the rules for which external systems may interface with Stores applications through Data Import. An XSD specifies the format for XML documents that are sent to Data Import. Any XML that is imported through Data Import is expected to validate successfully against the appropriate XSD for its type. Data Import does not perform a validity check. It is the responsibility of the sending party to send proper, conforming data. Invalid XML is not parsed correctly and either the invalid parts are ignored or a parsing exception is generated.

Employee Import

[Table B–1](#) identifies the XSD elements in the EmployeeImport.xsd file.

Table B–1 Employee Import XSD Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
Employee PA_EM	EmployeeID	ID_EM		Employee > EmployeeID	
	PartyID	ID_PRTY		Employee > PartyID	
	EmployeeLoginID	ID_LOGIN		Employee > EmployeeAccess > EmployeeLoginID	
	EmployeeAlternateID	ID_ALT		Employee > EmployeeAccess > EmployeeAltID	
	EmployeeAccess Password	PW_ACS_EM		Employee > EmployeeAccess > AccessPassword	

Table B-1 Employee Import XSD Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	EmployeeName	NM_EM	VARCHAR(150)	Employee > EmployeeFullName	
	Employee LastName	LN_EM	VARCHAR(50)	Employee > EmployeeLastName	
	Employee FirstName	FN_EM	VARCHAR(50)	Employee > EmployeeFirstName	
	EmployeeMiddleName	MD_EM	VARCHAR(50)	Employee > EmployeeMiddleName	
	EmployeeRole	ROLE_EM		Employee > EmployeeRole	
	SocialSecurityNumber	UN_NMB_SCL_SCTY		Employee > EmployeeSSN	
	EmployeeStatusCode	SC_EM		Employee > StatusCode	
	WorkGroupID	ID_GP_WRK		Employee > EmployeeAccess > WorkGroupID	
	EmployeeLocale	LCL		Employee > Locale	Only applies to temporary employees.
	NumberOfDaysValidFor TempEmployees	NUMB_DYS_VLD		Employee > NumberDaysValid	Only applies to temporary employees.
	ExpirationTimeFor TempEmployees	DC_EXP_TMP		Employee > TempEmployeeExpirationDate	0 means Standard employee. 1 means Temporary employee.
	EmployeeType	TYPE_EMP		Employee > EmployeeType	
	EmployeeStore Assignment	ID_STR_RT		Employee > EmployeeStoreOrHierarchyAssn > EmployeeStoreID	
	NewPassword RequiredFlag	FL_PW_NW_REQ		Employee > EmployeeAccess > NewPasswordRequired	
	PasswordCreatedDate	TS_CRT_PW		Employee > EmployeeAccess > PasswordCreationDate	If date is not specified, a new date is used.

Table B-1 Employee Import XSD Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	NumberOfFailed Passwords	NUMB_FLD_PW		0	No failed passwords inserted.
Employee Password History MA_HST_PW_EM	EmployeeID	ID_EM		Employee > EmployeeID	
	PasswordCreateDate	TS_CRT_PW		Employee > EmployeeAccess > PasswordHistory Entry > PasswordCreation Date	If date is not specified, a new date is used.
	EmployeeAccessPassword	PW_ACS_EM		Employee > EmployeeAccess > PasswordHistoryEntry > AccessPassword	
Employee Hierarchy Association EMPLOYEE_HIERARCHY_ASSN	LoginID	ID_LOGIN		Employee > EmployeeAccess > EmployeeLoginID	
	FunctionID	ID_STRGP_FNC		Employee > EmployeeStoreOrHierarchyAssn > EmployeeHierarchyAssn > StoreGroupFunctionID	
	GroupID	ID		Employee > EmployeeStoreOrHierarchyAssn > EmployeeHierarchyAssn > NodeID	
	GroupType	TYPE		Employee > EmployeeStoreOrHierarchyAssn > EmployeeHierarchyAssn > NodeType	

Example B-1 EmployeeImport.xsd

```

<?xml version="1.0" encoding="windows-1252" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">

    <!-- $Log:$ -->

    <xs:annotation><xs:documentation>
        Employee Import Schema. Copyright 2006 Oracle. All rights reserved.
    </xs:documentation></xs:annotation>

    <xs:element name="EmployeeImport" type="EmployeeImport">
        <xs:annotation><xs:documentation>
            Top-level element holding a collection of Employee elements.
        </xs:documentation></xs:annotation>
    </xs:element>

    <xs:complexType name="EmployeeImport">
        <xs:sequence>
            <xs:element name="Employee" type="EmployeeType" minOccurs="1"
maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="FillType" type="FillType" use="required"/>
        <xs:attribute name="CreationDate" type="xs:dateTime"/>
        <xs:attribute name="ExpirationDate" type="xs:dateTime"/>
        <xs:attribute name="Version" type="xs:string"/>
        <xs:attribute name="Priority" type="xs:int"/>
        <xs:attribute name="Batch" type="xs:int"/>
    </xs:complexType>

    <xs:complexType name="EmployeeType">
        <xs:annotation><xs:documentation>
            Represents a single employee's information.
        </xs:documentation></xs:annotation>
        <xs:sequence>
            <xs:element name="ChangeType" type="ChangeType" default="ADD"
minOccurs="1" maxOccurs="1" />
            <xs:element name="EmployeeID" type="ID" minOccurs="1" maxOccurs="1" />
            <xs:element name="EmployeeFirstName" type="xs:string" minOccurs="0"
maxOccurs="1" />
            <xs:element name="EmployeeLastName" type="xs:string" minOccurs="0"
maxOccurs="1" />
            <xs:element name="EmployeeMiddleName" type="xs:string" minOccurs="0"
maxOccurs="1" />
            <xs:element name="EmployeeFullName" type="xs:string" minOccurs="0"
maxOccurs="1" />
            <xs:element name="EmployeeSSN" type="SSN" minOccurs="0" maxOccurs="1"
/>
            <xs:element name="EmployeeRole" type="xs:string" minOccurs="0"
maxOccurs="1" />
            <xs:element name="PartyID" type="xs:int" minOccurs="0" maxOccurs="1"
/>
            <xs:element name="StatusCode" type="StatusCode" minOccurs="0"
maxOccurs="1" />
            <xs:element name="Locale" type="ID" minOccurs="0" maxOccurs="1" />
            <xs:element name="EmployeeAccess" type="EmployeeAccess" minOccurs="0"
maxOccurs="1" />
            <xs:element name="EmployeeType" type="StatusCode">
                <xs:annotation><xs:documentation>
                    0 means 'Standard' employee, 1 means Temporary employee

```



```

        </xs:documentation></xs:annotation>
    </xs:element>
    <xs:element name="NumberDaysValid" type="xs:int" minOccurs="0"
maxOccurs="1">
        <xs:annotation><xs:documentation>
            Only applies to temporary employee
        </xs:documentation></xs:annotation>
    </xs:element>
    <xs:element name="TempEmployeeExpirationDate" type="xs:date"
minOccurs="0" maxOccurs="1">
        <xs:annotation><xs:documentation>
            Only applies to temporary employee
        </xs:documentation></xs:annotation>
    </xs:element>
    <xs:element name="EmployeeStoreOrHierarchyAssn"
type="EmployeeStoreOrHierarchyAssn" minOccurs="0" maxOccurs="unbounded" />
</xs:sequence>
</xs:complexType>

<xs:simpleType name="ID">
    <xs:restriction base="xs:string">
        <xs:maxLength value="10" />
    </xs:restriction>
</xs:simpleType>

<xs:simpleType name="SSN">
    <xs:restriction base="xs:string">
        <xs:maxLength value="9" />
    </xs:restriction>
</xs:simpleType>

<xs:simpleType name="StatusCode">
    <xs:restriction base="xs:string">
        <xs:enumeration value="0" />
        <xs:enumeration value="1" />
    </xs:restriction>
</xs:simpleType>

<xs:complexType name="EmployeeAccess">
    <xs:annotation><xs:documentation>
        Holds all information regarding access to the system.
    </xs:documentation></xs:annotation>
    <xs:sequence>
        <xs:element name="EmployeeLoginID" type="xs:string" />
        <xs:element name="AccessPassword" type="xs:string" />
        <xs:element name="WorkGroupID" type="xs:int" />
        <xs:element name="EmployeeAltID" type="xs:string" minOccurs="0"
maxOccurs="1" />
        <xs:element name="NewPasswordRequired" type="xs:boolean" />
        <xs:element name="PasswordCreationDate" type="xs:dateTime" />
        <xs:element name="PasswordHistory" type="PasswordHistory"
minOccurs="0" maxOccurs="1">
            </xs:element>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="PasswordHistory">
        <xs:sequence>
            <xs:element name="PasswordHistoryEntry" type="PasswordHistoryEntry"
minOccurs="1" maxOccurs="unbounded" />

```

```
</xs:sequence>
</xs:complexType>

<xs:complexType name="PasswordHistoryEntry">
  <xs:annotation><xs:documentation>
    Holds a single password history entry.
  </xs:documentation></xs:annotation>
  <xs:sequence>
    <xs:element name="PasswordCreationDate" type="xs:dateTime" />
    <xs:element name="AccessPassword" type="xs:string" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="EmployeeStoreOrHierarchyAssn">
  <xs:annotation><xs:documentation>
    Holds an employee association to a store and/or a hierarchy node.
    Generally, only one of the
    enclosed elements is provided; however, there may be cases where an
    employee needs both a store
    association and a hierarchy association, so a sequence with optional
    elements is used instead of
    a choice.
  </xs:documentation></xs:annotation>
  <xs:sequence>
    <xs:element name="EmployeeStoreID" type="RetailStoreId" minOccurs="0"
maxOccurs="1" />
    <xs:element name="EmployeeHierarchyAssn" type="EmployeeHierarchyAssn"
minOccurs="0" maxOccurs="1" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="EmployeeHierarchyAssn">
  <xs:sequence>
    <xs:element name="NodeID" type="xs:string" minOccurs="1"
maxOccurs="1" />
    <xs:element name="NodeType" type="xs:string" minOccurs="1"
maxOccurs="1" />
    <xs:element name="StoreGroupFunctionID" type="xs:int" minOccurs="1"
maxOccurs="1" />
  </xs:sequence>
</xs:complexType>

<xs:simpleType name="ChangeType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="ADD" />
    <xs:enumeration value="UPD" />
    <xs:enumeration value="DEL" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="RetailStoreId">
  <xs:annotation><xs:documentation>
    Store Id's can only be five characters long and preferably only
    numerals.
  </xs:documentation></xs:annotation>
  <xs:restriction base="xs:string">
    <xs:minLength value="1"></xs:minLength>
    <xs:maxLength value="5"></xs:maxLength>
  </xs:restriction>
</xs:simpleType>
```

```

    <xs:simpleType name="FillType">
      <xs:restriction base="xs:string">
        <xs:enumeration value="KillAndFill"/>
        <xs:enumeration value="FullIncremental"/>
      </xs:restriction>
    </xs:simpleType>

  </xs:schema>

```

Item Import

Table B–2 identifies the PreloadData element mapping for the ItemImport.xsd file.

Table B–2 Item Import XSD PreloadData Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
ItemColor CO_CLR	ColorCode	ED_CLR	VARCHAR(20)	PreloadData > Color @Code	
	ColorNames	NM_CLR	VARCHAR(40)	PreloadData > Color @Names	Contains a short list of names given to this color.
	Description	DE_CLR	VARCHAR(250)	PreloadData > Color @Description	
	RecordCreated Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	
ItemSize CO_SZ	SizeCode	ED_SZ	VARCHAR(6)	PreloadData > Size @Code	
	ActualSize Proportion Description	DE_PRPTN_ACT_SZ	VARCHAR(250)	PreloadData > Size @ProportionDesc	
	ActualSize Type Description	DE_TYP_ACT_SZ	VARCHAR(40)	PreloadData > Size @TypeDesc	
	ActualSize Code	ED_SZ_ACT	VARCHAR(20)	PreloadSize > Size @ActualSizeCode	
	TableName	NM_TB_SZ	VARCHAR(40)	PreloadSize > Size @TableName	
	TableCode	ED_TB_SZ	VARCHAR(20)	PreloadSize > Size @TableCode	
	Table Description	DE_TB_SZ	VARCHAR(250)	PreloadSize > Size @TableDesc	
	RecordCreated Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	

Table B–2 Item Import XSD PreloadData Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
ItemStyle CO_STYL	StyleCode	LU_STYL	VARCHAR(4)	PreloadData > Style @Code	
	StyleName	NM_STYL	VARCHAR(40)	PreloadData > Style @Name	Contains a short list of names given to this color.
	Description	DE_STYL	VARCHAR(250)	PreloadData > Style @Description	
	RecordCreated Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	
UnitOf Measure CO_UOM	UnitOf MeasureCode	LU_UOM	VARCHAR(2)	PreloadData > UOM @Code	
	UnitOf MeasureType Code	TY_UOM	VARCHAR(2)	PreloadData > UOM @TypeCode	
	EnglishMetric Flag	FL_UOM_ENG_MC		PreloadData > UOM @System	Metric=true
	Name	NM_UOM	VARCHAR(40)	PreloadData > UOM @Name	
	Description	DE_UOM	VARCHAR(250)	PreloadData > UOM @Description	
	DefaultUnitOf MeasureFlag	FL_DFLT_UOM		PreloadData > UOM @IsDefault	
	DefaultEntry Code	FL_CD_ENT_DFLT		PreloadData > UOM @DefaultEntry Code	
	EnabledFlag	FL_CD_ENT_ENAB		PreloadData > UOM @Enabled	
	ListSortIndex	CD_ENT_SRT		PreloadData > UOM @SortIndex	
	RecordCreated Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	
Merchandise Classification Code LU_CD_STRC_MR	Merchandise Classification Code	ID_STRC_MR_CD	VARCHAR(10)	PreloadData > Merchandise Classification @Code	

Table B–2 Item Import XSD PreloadData Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	Merchandise Classification Code Description	DE_STRC_MR_CD	VARCHAR(250)	PreloadData > Merchandise Classification @Description	
	RecordCreated Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	
Supplier PA_SPR	Supplier	ID_SPR	VARCHAR(20)	PreloadData > Supplier @ID	
	DUNSNumber	DU_SPR	VARCHAR(9)	PreloadData > Supplier @DUNSNumber	
	Name	NM_SPR	VARCHAR(40)	PreloadData > Supplier @Name	
	SupplierIs Manufacturer Flag	FL_MF_SPR_IS		PreloadData > Supplier @IsManufacturer	
	PartyRoleType Code	TY_RO_PRTY		No Mapping Found	Null value is entered. Column is not used in database.
	PartyID	ID_PRTY		No Mapping Found	Null value is entered. Column is used in database.
	RecordCreated Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	
Manufacturer PA_MF	Manufacturer ID	ID_MF		PreloadData > Manufacturer @ID	
	Name	NM_MF	VARCHAR(80)	PreloadData > Manufacturer @Name	
	PartyID	ID_PRTY		No mapping available	Null value to be stored.
	RecordCreated Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	

Table B–3 identifies the item element mapping for the ItemImport.xsd file.

Table B–3 Item Import XSD Item Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
Item	ItemID	ID_ITM	VARCHAR(14)	Item @ID	
AS_ITM					
	ItemProductID	ID_ITM_PDT	VARCHAR(14)	No Mapping	
	DiscountFlag	FL_ITM_DSC		Item @Discountable	true = 1, false= 0
	Damage DiscountFlag	FL_ITM_DSC_DMG		Item @Damage Discountable	true = 1, false= 0
	ItemSize RequiredFlag	FL_ITM_SZ_REQ		Item @SizeRequired	true = 1, false= 0
	POS DepartmentID	ID_DPT_POS	VARCHAR(14)	Item @POS DepartmentID	
	AuthorizedFor SaleFlag	FL_AZN_FR_SLS		Item @Authorized ForSale	true = 1, false= 0
	TaxExempt Code	LU_EXM_TX	VARCHAR(20)	Item @Taxable	true = 1, false= 0
	Description	DE_ITM	VARCHAR(250)	Item > Description	
	Abbreviated Description	DE_ITM_SHRT	VARCHAR(120)	Item > ShortName	Based on the default locale. The ShortName specific to the locale is inserted into the column. When application is i81N aware, locale-specific data is inserted into the locale table.
	TypeCode	TY_ITM	VARCHAR(20)	Item @Type	Stock=STCK Service=SRVC Coupon=SCPN
	KitSetCode	LU_KT_ST	VARCHAR(20)	Item @KitSetCode	0 (Default Value) means item is not part of a kit. 1 means it is a kit and this item is the header of the kit. 2 means this item is one of the component of the kit.
	Merchandise StructureID	ID_STRC_MR		Item > Merchandise Hierarchy @StructureID	Notes: Some question as to whether we are actually using this.
	Merchandise Hierarchy LevelCode	LU_HRC_MR_LV	VARCHAR(4)	Item > Merchandise Hierarchy @Level	
	Merchandise HierarchyID	ID_MRHRC_GP	VARCHAR(14)	Item > Merchandise Hierarchy	

Table B-3 Item Import XSD Item Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	TaxGroupID	ID_GP_TX		<ul style="list-style-type: none"> RetailStore Item@VatCode RetailStore Item@Tax Group Item@Tax Group 	If the vatcode or the taxgroup attributes are given in the retailstoreitem tag the corresponding value only will be inserted in the ID_GP_TX column. If both of the vatcode and taxgroup attributes are not provided, the Item@taxgroup attribute is considered; otherwise it is ignored.
	Activation RequiredFlag	FL_ACTVN_RQ		Item @Activation Required	true = 1, false= 0
	Registry EligibleFlag	FL_ITM_RGSTRY		Item @RegistryEligible	true = 1, false= 0
	Merchandise Classification Code00	ID_STRC_MR_CD0	VARCHAR(10)	Item @Classification1	
	Merchandise Classification Code01	ID_STRC_MR_CD1	VARCHAR(10)	Item @Classification2	
	Merchandise Classification Code02	ID_STRC_MR_CD2	VARCHAR(10)	Item @Classification3	
	Merchandise Classification Code03	ID_STRC_MR_CD3	VARCHAR(10)	Item @Classification4	
	Merchandise Classification Code04	ID_STRC_MR_CD4	VARCHAR(10)	Item @Classification5	
	Merchandise Classification Code05	ID_STRC_MR_CD5	VARCHAR(10)	Item @Classification6	
	Merchandise Classification Code06	ID_STRC_MR_CD6	VARCHAR(10)	Item @Classification7	
	Merchandise Classification Code07	ID_STRC_MR_CD7	VARCHAR(10)	Item @Classification8	
	Merchandise Classification Code08	ID_STRC_MR_CD8	VARCHAR(10)	Item @Classification9	
	Merchandise Classification Code09	ID_STRC_MR_CD9	VARCHAR(10)	Item @Classification10	
	PriceAudit Flag2	FL_ADT_ITM_PRC		No Mapping	Null value to be entered.
	UsageCode	LU_ITM_USG		No Mapping	Null value to be entered.
	Name	NM_ITM		No Mapping	Null value to be entered.

Table B-3 Item Import XSD Item Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	Tax Category	ID_CTGY_TX		Item@TaxCategory	
	Retail less than MRP flag	FL_MRP_RT_LT		Item@RetailLessThanMRP	true=1 false=0
	Multiple MRP Indicator	FL_MRP_MU		Item@MultipleMRPIndicator	true=1 false=0
	Substitute IdentifiedFlag	FL_ITM_SBST_IDN		No Mapping	Default value of 0.
	Order Collection Code	LU_CLN_ORD		No Mapping	Null value to be entered.
	PriceLineID	ID_LN_PRC		No Mapping	Null value to be entered.
	BrandName	NM_BRN		No Mapping	Null value to be entered.
	SeasonCode	LU_SN		No Mapping	Null value to be entered.
	FiscalYear	FY		No Mapping	Null value to be entered.
	Subseason Code	LU_SBSN		No Mapping	Null value to be entered.
	RecordCreated Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	
StockItem AS_ITM_STK	ItemID	ID_ITM	VARCHAR(14)	Item @ID	
	StockItemSale UnitOf MeasureCode	LU_UOM_SLS	VARCHAR(20)	Item @UOMCode	Default to UN for units is not specified.
	ColorCode	ED_CLR	VARCHAR(20)	Item @Color	
	SizeCode	ED_SZ	VARCHAR(6)	Item @Size	
	StyleCode	LU_STYL	VARCHAR(4)	Item @Style	
	SupplierID	ID_SPR	VARCHAR(20)	Item > RetailStoreItem > POSIdentity @SupplierID	
	PackItem WeightCount	QW_ITM_PCK		Item @PackItemWeight Count	
	SerializedItem ValidationFlag	FL_VLD_SRZ_ITM		Item @SerializedItem	true = 1, false= 0
	RestockingFee Flag	FL_FE_RSTK		Item @RestockingFee	true = 1, false= 0
	Record Creation Timestamp	TS_CRT_RCRD		Now()	

Table B-3 Item Import XSD Item Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	
RetailStore Item AS_ITM_RTL_STR	RetailStoreID	ID_STR_RT	VARCHAR(5)	StoreID from the Manifest Meta Data	
	ItemID	ID_ITM	VARCHAR(14)	Item @ID	
	Primary Maximum Retail Price	PRC_MXM_RT		Item > RetailStoreItem@MRP	Primary MRP is always imported from RMS.
	TaxGroupID	ID_GP_TX		Item > RetailStoreItem@TaxGroup	If vatcode is not provided, then only ID_GP_TX is filled with the value of the taxgroup attribute.
	VatCode	ID_GP_TX		Item > RetailStoreItem@VatCode	The Vat Code is the VAT Code Name. VatCode must be translated from some String (xs:string) to an Integer. The VatCode should match a name specified in RU_TX_GP.NM_RU_TX. The ID_GP_TX of the name is the ID used to insert into AS_ITM_RTL_STR.ID_GP_TAX for the incoming VatCode.
				Item > RetailStoreItem > SellingPrice@IncludesTax	
				Item > RetailStoreItem > SellingPrice@CurrencyCode	Reserved for future use for foreign currency support.
	PermanentSale UnitRetail PriceAmount	RP_PR_SLS		Item > RetailStoreItem > SellingPrice@PermanentPrice	
	CompareAt SaleUnitRetail PriceAmount	RP_PRC_CMPR_AT_SLS		Item > RetailStoreItem > SellingPrice@CompareAtPrice	
	SalesAge Restriction Identifier	IDN_SLS_AG_RST		Item > RetailStoreItem@AgeRestrictionID	
	TemplateID	ID_TMPLT_LB	VARCHAR(8)	Item > RetailStoreItem@TemplateID	

Table B-3 Item Import XSD Item Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	Record Creation Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	
POSIdentity ID_IDN_PS	RetailStoreID	ID_STR_RT	VARCHAR(5)	StoreID from the Manifest Meta Data	
	POSItemID	ID_ITM_POS	VARCHAR(14)	Item > RetailStoreItem > POSIdentity @POSItemID	
	ItemID	ID_ITM	VARCHAR(14)	Item @ID	
	CurrentSale UnitPOSRetail PriceAmount	RP_SLS_POS_CRT		Item > RetailStoreItem > SellingPrice	
	Manufacturer UPC	ID_ITM_MF_UPC	VARCHAR(14)	Item > RetailStoreItem > POSIdentity @UPC	
	Manufacturer ID	ID_MF		Item > RetailStoreItem > POSIdentity@ManufacturerID	
	Employee Discount AllowedFlag	FL_DSC_EM_ALW		Item > RetailStoreItem > POSIdentity @Employee DiscountAllowed	true = 1, false= 0
	PriceEntry RequiredFlag	FL_ENTR_PRC_RQ		Item > RetailStoreItem > POSIdentity @PriceEntry Required	true = 1, false= 0
	QuantityKey ProhibitFlag	FL_KY_PRH_QTY		Item > RetailStoreItem > POSIdentity @Quantity Modifiable	true = 1, false= 0
	ProhibitReturn Flag	FL_RTN_PRH		Item > RetailStoreItem > POSIdentity @Returnable	true = 1, false= 0
	Price Modifiable Flag	FL_MDFR_RT_PRC		Item > RetailStoreItem > POSIdentity @PriceModifiable	true = 1, false= 0

Table B-3 Item Import XSD Item Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	MinimumSale UnitCount	QU_UN_BLK_MNM		Item > RetailStoreItem > POSIdentity @MinimumSale UnitCount	
	MaximumSale UnitCount	QU_UN_BLK_MXM		Item > RetailStoreItem > POSIdentity @MaximumSale UnitCount	
	AllowCoupon MultiplyFlag	FL_CPN_ALW_MULTY		Item > RetailStoreItem > POSIdentity @AllowCoupon Multiply	true = 1, false= 0
	Electronic CouponFlag	FL_CPN_ELNTC		Item > RetailStoreItem > POSIdentity @Electronic Coupon	true = 1, false= 0
	Coupon RestrictedFlag	FL_CPN_RST		Item > RetailStoreItem > POSIdentity @Coupon Restricted	true = 1, false= 0
	SpecialOrder EligibleFlag	FL_SPO_ITM		Item > RetailStoreItem > POSIdentity @SpecialOrder Eligible	true = 1, false= 0
	Record Creation Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	
SupplierItemCatalogBaseCostBreak CO_BRK_SPR_ITM_BS	SupplierID	ID_SPR	VARCHAR(20)	Item > RetailStoreItem > POSIdentity @SupplierID	Note that SupplierID is required for deleting items.
	SupplierItem ID	ID_ITM_SPR	VARCHAR(20)	Item @ID	
	SupplierItem CostPerUnit TypeCode	TY_UN_CST	VARCHAR(3)	SLU	
	SupplierItem UnitBreak PointCount	QU_PNT_UND_BRK		0	

Table B–3 Item Import XSD Item Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	CostPerUnit Amount	CP_PNT_ BRK_BS_CST		Item @ItemCost	
	Record Creation Timestamp	TS_CRT_ RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_ RCRD		Now()	

Table B-4 identifies the Oracle Retail Merchandising System export files element mapping for the ItemImport.xsd file.

Table B-4 Item Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Record Header	Field Name	XSD Element/Attribute Path	Notes
POSDNLD	FHEAD FDETL	File Create Date	ItemImport @CreationDate	YYYYMMDD to YYYY-MM-DD T HH24:MI:SS
		Record Type	Item @Type	type=Stock
		Location Number	Item > RetailStoreItem > RetailStoreID	
		Update Type		No direct mapping. Can be used during transform process to determine which fields are appropriate.
		Start Date		No item fields use this value any longer. Basically, this is a hold over from when Oracle Retail Price Management was still part of Oracle Retail Merchandising System.
		Time		No item fields use this value any longer. Basically, this is a hold over from when Oracle Retail Price Management was still part of Oracle Retail Merchandising System.
		Transaction Type		No direct mapping. Can be used during transform process to determine which fields are appropriate.
		Item Number ID	Item @ID	Item field length needs to be lengthened in database and XML Schema.
		Item Number Type		Unused
		Format ID		Unused
		Prefix		Unused
		Reference Item		Point-of-Service only supports transaction level items.
		Reference Item Number Type		Unused
		Reference Item Format ID		Unused
		Reference Item Prefix		Unused
		Item Short Description	Item > ShortName	DE_ITM_SHRT column to be expanded to support multibyte characters.

Table B-4 Item Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Record Header	Field Name	XSD Element/Attribute Path	Notes
		Item Long Description	Item > Description	DE_ITM column to be expanded to support multibyte characters.
		Department ID	Item @POSDepartmentID	
		Class ID	Item @Classification1	
		Subclass ID	Item @Classification2	
		New Price	Item > RetailStoreItem > SellingPrice	
		New Selling UOM	Item @UOMCode	
		New Multi Units		No item fields use this value any longer. Basically, this is a hold over from when Oracle Retail Price Management was still part of Oracle Retail Merchandising System.
		New Multi Selling UOM		No item fields use this value any longer. Basically, this is a hold over from when Oracle Retail Price Management was still part of Oracle Retail Merchandising System.
		Status	Item @ChangeType tem @AuthorizedForSale	If Transaction Type = 1 (new item added), ChangeType = ADD. If Status = A, AuthorizedForSale = true . If Status = I or C, AuthorizedForSale = false . If Status = D, ChangeType = DEL, else ChangeType = UPD (except for Transaction Type = 1).
		Taxable Indicator	Item @Taxable	Y = true, N = false
		Promotion Number		No item fields use this value any longer. Basically, this is a hold over from when Oracle Retail Price Management was still part of Oracle Retail Merchandising System.
		Mix Match Number		No item fields use this value any longer. Basically, this is a hold over from when Oracle Retail Price Management was still part of Oracle Retail Merchandising System.
		Mix Match Type		No item fields use this value any longer. Basically, this is a hold over from when Oracle Retail Price Management was still part of Oracle Retail Merchandising System.

Table B-4 Item Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Record Header	Field Name	XSD Element/Attribute Path	Notes
		Threshold Number		No item fields use this value any longer. Basically, this is a hold over from when Oracle Retail Price Management was still part of Oracle Retail Merchandising System.
		Launch Date		Unused
		Quantity Key Options	Item > RetailStoreItem > POSIdentity @QuantityModifiable	R = Required P = Prohibited O = Optional
		Manual Price Entry	Item > RetailStoreItem > POSIdentity @PriceEntryRequired Item > RetailStoreItem > POSIdentity @PriceModifiable	PriceEntryRequired = true if Manual Price Entry = R, else false . PriceModifiable = false if Manual Price Entry = P, else true .
		Deposit Code		Unused
		Food Stamp Indicator		Unused
		WIC Indicator		Unused
		Proportional Tare Percent		Unused. Point-of-Service does not record tare values.
		Fixed Tare Value		Unused. Point-of-Service does not record tare values.
		Fixed Tare UOM		Unused. Point-of-Service does not record tare values.
		Reward Eligible Indicator		Unused
		Elective Marketing Clubs		Unused
		Return Policy		Unused
		Stop Sale Indicator	Item @AuthorizedForSale	Y = true, N = false
		Returnable Indicator	Item > RetailStoreItem > POSIdentity @Returnable	Y = true, N = false
		Refundable Indicator		Unused
		Back Order Indicator	Item > RetailStoreItem > POSIdentity @SpecialOrderEligible	Y = true, N = false
		VAT Code	Item > RetailStoreItem @VATCode	
		VAT Rate		VAT rates are populated through TaxImport.
		Class VAT Indicator		Unused. Point-of-Service does not support VAT rates at a class level.

Table B-4 Item Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Record Header	Field Name	XSD Element/Attribute Path	Notes
		Promotion Item Type		No item fields use this value any longer. Basically, this is a hold over from when Oracle Retail Price Management was still part of Oracle Retail Merchandising System.
		Catch Weight Indicator		Unused
		Sale Type		Unused. Point-of-Service does not support variable or loose weights.
		Container Item		Unused
POSCDNLD	FHEAD	File Date	ItemImport @CreationDate	YYYYMMDD HHMMSS to YYYY-MM-DD T HH24:MI:SS This is only required if POSDNLD is not processed as part of the batch.
	TCOUP	Record Type	Item @Type	type= Coupon
		Coupon id	Item @Id	
		Coupon Desc	Item > Description	
		Currency Code	Item > RetailStoreItem > SellingPrice @CurrencyCode	
		Max Discount Amount		
		Amount		
		Percent Indicator		
		Profit Center		
		Tax Class	Item @TaxGroup	
		Export Code		
		Effective Date		
		Expiration Date		
		Prompted Ind		
		Display Ind		
		Status	Item @ChangeType	A = ADD C = UPD D = DEL
		Vendor	Item @ManufacturerID	
		Vendor Type		
		Promotion		
		Coupon Barcode	Item > RetailStoreItem > POSIdentity @UPC	

Table B-4 Item Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Record Header	Field Name	XSD Element/Attribute Path	Notes
		Coupon Max Qty	Item > RetailStoreItem > POSIdentity @MaximumSaleUnitCount	
	TPRES	POS Product Restriction ID		No direct mapping.
		POS Product Restriction Desc		No direct mapping.
		POS Product Restriction Type		No direct mapping. Point-of-Service supports: MNAG -- Minimum Age NDSC -- Nondiscountable RTRN -- Returnable QLMT -- Quantity Limit
		Effective Date		
		Currency Code		
		Product Restriction Amount		
		Age Minimum	Item > RetailStoreItem @AgeRestrictionId	
		Date Restriction		
		Before Time Restriction		
		After Time Restriction		
		Day Restriction		
		Max Qty Amount	Item > RetailStoreItem > POSIdentity @MaximumSaleUnitCount	
		Tender Type Group		
		Status	Item > RetailStoreItem @ChangeType Item > RetailStoreItem > POSIdentity @ChangeType	A = ADD C = UPD D = DEL Depending on which attribute is affected, either the RetailStoreItem@ChangeType or the POSIdentity@ChangeType are populated here. Likely not both.
	TMORD	Record Type		Unused
	TTTYP	Record Type		Unused
	TBTTN	Record Type		Unused
	TPYIO	Record Type		Unused
	TSPAY	Record Type		Unused

Table B-4 Item Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Record Header	Field Name	XSD Element/Attribute Path	Notes
	TSTOR	Record Type		Unused. Additions and deletions of store IDs are supported through the StoreHierarchy Data Import, which maps to other Oracle Retail Merchandising System files, rmse_store.dat and rmse_orghier.dat documents.
	TITEM	Item	Item @ID	
		Status	Item @ChangeType	A = ADD C = UPD D = DEL
		Button ID		Unused
TXRPOSDN	FHEAD	File Date	ItemImport @CreationDate	YYYYMMDD HHMMSS to YYYY-MM-DD T HH24:MI:SS This is only required if POSDNLD is not processed as part of the batch.
		UNKNOWN	Item @TaxGroup Item > RetailStoreItem @CreationDate	

Example B-2 ItemImport.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">

  <!-- $Log:$ -->

  <xs:annotation>
    <xs:documentation>
      Item Import Schema. Copyright 2007 Oracle Inc. All rights reserved.

      Use this schema in conjunction with a Oracle Store Systems Data Dictionary
      and the relations between the element and attribute names should be
      apparent.
    </xs:documentation>
  </xs:annotation>

  <xs:element name="ItemImport">
    <xs:annotation><xs:documentation>
      Top-level element holding a collection of Item records.
    </xs:documentation></xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element name="PreloadData" type="PreloadData" minOccurs="0" maxOccurs="1"/>
        <xs:element name="Item" type="Item" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="FillType" type="FillType" use="required"/>
      <xs:attribute name="CreationDate" type="xs:dateTime"/>
    </xs:complexType>
  </xs:element>

```

```

<xs:attribute name="ExpirationDate" type="xs:dateTime"/>
<xs:attribute name="Version" type="xs:string"/>
<xs:attribute name="Priority" type="xs:int"/>
<xs:attribute name="Batch" type="xs:int"/>
</xs:complexType>
</xs:element>

<xs:complexType name="PreloadData">
<xs:sequence>
<xs:element name="Color" type="Color" minOccurs="0" maxOccurs="unbounded"/>
<xs:element name="Size" type="Size" minOccurs="0" maxOccurs="unbounded"/>
<xs:element name="Style" type="Style" minOccurs="0" maxOccurs="unbounded"/>
<xs:element name="UOM" type="UOM" minOccurs="0" maxOccurs="unbounded"/>
<xs:element name="Manufacturer" type="Manufacturer" minOccurs="0"
maxOccurs="unbounded"/>
<xs:element name="MerchandiseClassification" type="MerchandiseClassification"
minOccurs="0" maxOccurs="unbounded"/>
<xs:element name="Supplier" type="Supplier" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="Color">
<xs:attribute name="ChangeType" type="PreLoadChangeType" default="UPS"/>
<xs:attribute name="Code" type="Code" use="required"/>
<xs:attribute name="Description" type="Description"/>
<xs:attribute name="Names" type="Name">
<xs:annotation><xs:documentation>
A list of short names given to this color.
</xs:documentation></xs:annotation>
</xs:attribute>
</xs:complexType>

<xs:complexType name="Size">
<xs:attribute name="ChangeType" type="PreLoadChangeType" default="UPS"/>
<xs:attribute name="Code" use="required">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:maxLength value="6"/>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
<xs:attribute name="ProportionDesc" type="Description"/>
<xs:attribute name="TypeDesc" type="Name"/>
<xs:attribute name="ActualSizeCode" type="Code"/>
<xs:attribute name="TableName" type="Name"/>
<xs:attribute name="TableCode" type="Code"/>
<xs:attribute name="TableDesc" type="Description"/>
</xs:complexType>

<xs:complexType name="Style">
<xs:attribute name="ChangeType" type="PreLoadChangeType" default="UPS"/>
<xs:attribute name="Code" use="required">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:maxLength value="4"/>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
<xs:attribute name="Name" type="Name"/>
<xs:attribute name="Description" type="Description"/>

```

```
</xs:complexType>

<xs:complexType name="UOM">
  <xs:attribute name="ChangeType" type="PreLoadChangeType" default="UPS"/>
  <xs:attribute name="Code" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:maxLength value="2"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="TypeCode">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:maxLength value="2"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="System">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="Standard"/>
        <xs:enumeration value="Metric"/>
        <!-- xs:enumeration value="Imperial"/ -->
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="Name" type="Name" use="required"/>
  <xs:attribute name="Description" type="Description"/>
  <xs:attribute name="IsDefault" type="xs:boolean" default="false"/>
  <xs:attribute name="DefaultEntryCode" type="xs:boolean" default="false"/>
  <xs:attribute name="Enabled" type="xs:boolean" default="true"/>
  <xs:attribute name="SortIndex" type="xs:int" use="required"/>
</xs:complexType>

<xs:complexType name="Manufacturer">
  <xs:attribute name="ChangeType" type="PreLoadChangeType" default="UPS"/>
  <xs:attribute name="ID" type="Code" use="required"/>
  <xs:attribute name="Name" type="Name"/>
</xs:complexType>

<xs:complexType name="MerchandiseClassification">
  <xs:attribute name="ChangeType" type="PreLoadChangeType" default="UPS"/>
  <xs:attribute name="Code" type="Class" use="required"/>
  <xs:attribute name="Description" type="Description"/>
</xs:complexType>

<xs:complexType name="Supplier">
  <xs:attribute name="ChangeType" type="PreLoadChangeType" default="UPS"/>
  <xs:attribute name="ID" type="Code" use="required"/>
  <xs:attribute name="DUNSNumber">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:maxLength value="9"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="Name" type="Name"/>
  <xs:attribute name="IsManufacturer" type="xs:boolean" default="false"/>
</xs:complexType>
```

```

<xs:complexType name="Item">
  <xs:annotation><xs:documentation>
    Upper level item information. This element requires a child element
    to specify which store it belongs to. This element can be repeated
    if this item should belong to multiple stores. The name and des-
    cription elements may also be repeated with the intention that each
    specifies a different language or country.
  </xs:documentation></xs:annotation>
  <xs:sequence>
    <xs:element name="ShortName" type="LocalizedName" minOccurs="0"
      maxOccurs="unbounded" />
    <xs:element name="LongDescription" type="LocalizedDescription" minOccurs="0"
      maxOccurs="unbounded" />
    <xs:element name="MerchandiseHierarchy" type="MerchandiseHierarchy"
      minOccurs="0" />
    <xs:element name="RetailStoreItem" type="RetailStoreItem" maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="ChangeType" type="ChangeType" default="ADD" />
  <xs:attribute name="ID" type="ID" use="required" />
  <xs:attribute name="Type">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="Stock" />
        <xs:enumeration value="Service" />
        <xs:enumeration value="Coupon" />
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="POSDepartmentID" type="Class" />
  <xs:attribute name="ItemCost" type="Currency" />
  <xs:attribute name="KitSetCode" type="Code" default="0" />
  <xs:attribute name="UOMCode" type="Code" />
  <xs:attribute name="PackItemWeightCount" type="xs:decimal" />
  <xs:attribute name="Size">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:maxLength value="6" />
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="Color" type="Code" />
  <xs:attribute name="Style">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:maxLength value="4" />
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="Classification1" type="Class" />
  <xs:attribute name="Classification2" type="Class" />
  <xs:attribute name="Classification3" type="Class" />
  <xs:attribute name="Classification4" type="Class" />
  <xs:attribute name="Classification5" type="Class" />
  <xs:attribute name="Classification6" type="Class" />
  <xs:attribute name="Classification7" type="Class" />
  <xs:attribute name="Classification8" type="Class" />
  <xs:attribute name="Classification9" type="Class" />
  <xs:attribute name="Classification10" type="Class" />
  <xs:attribute name="TaxGroup" type="xs:int" />

```

```

<xs:attribute name="Taxable" type="xs:boolean" default="true"/>
<xs:attribute name="Discountable" type="xs:boolean" default="true"/>
<xs:attribute name="DamageDiscountable" type="xs:boolean" default="true"/>
<xs:attribute name="RegistryEligible" type="xs:boolean"/>
<xs:attribute name="RetailLessThanMRP" type="xs:boolean"/>
<xs:attribute name="MultipleMRP" type="xs:boolean"/>
<xs:attribute name="TaxCategory" type="xs:int"/>
<xs:attribute name="AuthorizedForSale" type="xs:boolean"/>
<xs:attribute name="RestockingFee" type="xs:boolean"/>
<xs:attribute name="SerializedItem" type="xs:boolean"/>
<xs:attribute name="SizeRequired" type="xs:boolean"/>
<xs:attribute name="ActivationRequired" type="xs:boolean"/>
</xs:complexType>

<xs:complexType name="RetailStoreItem">
  <xs:annotation><xs:documentation>
    Item-location information. This element requires a child element to
    specify a store id. This element can be repeated if this same info
    should belong to multiple stores. The price element may be repeated
    to support foreign currency by specifying differnt currency codes.
  </xs:documentation></xs:annotation>
  <xs:sequence>
    <xs:element name="RetailStoreID" type="RetailStoreId" maxOccurs="unbounded"/>
    <xs:element name="SellingPrice" type="SellingPrice" minOccurs="0"
      maxOccurs="unbounded"/>
    <xs:element name="POSIdentity" type="POSIdentity" minOccurs="0"
      maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="ChangeType" type="ChangeType" default="ADD"/>
  <xs:attribute name="TaxGroup" type="xs:int" use="optional"/>
  <xs:attribute name="VatCode" type="Code"/>
  <xs:attribute name="AgeRestrictionId" type="xs:int"/>
  <xs:attribute name="MRP" type="xs:decimal"/>
  <xs:attribute name="TemplateId">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:maxLength value="8"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
</xs:complexType>

<xs:complexType name="POSIdentity">
  <xs:annotation><xs:documentation>
    Multiple POSIdentity elements may be specified when different
    UPCs apply to the same item.
  </xs:documentation></xs:annotation>
  <xs:attribute name="ChangeType" type="ChangeType" default="ADD"/>
  <xs:attribute name="POSItemID" type="ID" use="required"/>
  <xs:attribute name="UPC" type="ID"/>
  <xs:attribute name="SupplierID" type="xs:string"/>
  <xs:attribute name="ManufacturerID" type="xs:int"/>
  <xs:attribute name="QuantityModifiable" default="Optional">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="Required"/>
        <xs:enumeration value="Prohibited"/>
        <xs:enumeration value="Optional"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
</xs:complexType>

```

```

</xs:attribute>
<xs:attribute name="Returnable" type="xs:boolean"/>
<xs:attribute name="PriceEntryRequired" type="xs:boolean" default="false"/>
<xs:attribute name="PriceModifiable" type="xs:boolean"/>
<xs:attribute name="AllowCouponMultiply" type="xs:boolean"/>
<xs:attribute name="ElectronicCoupon" type="xs:boolean"/>
<xs:attribute name="CouponRestricted" type="xs:boolean"/>
<xs:attribute name="SpecialOrderEligible" type="xs:boolean"/>
<xs:attribute name="EmployeeDiscountAllowed" type="xs:boolean" default="true"/>
<xs:attribute name="MinimumSaleUnitCount" type="xs:decimal" default="1.0"/>
<xs:attribute name="MaximumSaleUnitCount" type="xs:decimal" default="-1.0"/>
</xs:complexType>

<xs:complexType name="MerchandiseHierarchy">
  <xs:annotation><xs:documentation>
    This is the ID of the group in the MerchandiseHierarchy that this
    item belongs to. Usually this is a class or subclass.
  </xs:documentation></xs:annotation>
  <xs:simpleContent>
<xs:extension base="xs:string">
<xs:attribute name="StructureID" type="xs:string" default="-1">
<xs:annotation><xs:documentation>
Merchandise Structure ID.
</xs:documentation></xs:annotation>
</xs:attribute>
<xs:attribute name="Level" default="UNDF">
<xs:annotation><xs:documentation>
Merchandise Hierarchy Level Code.
</xs:documentation></xs:annotation>
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:maxLength value="4"/>
</xs:restriction>
</xs:simpleType>
</xs:attribute>
</xs:extension>
</xs:simpleContent>
</xs:complexType>

<xs:complexType name="SellingPrice">
<xs:simpleContent>
<xs:extension base="Currency">
<xs:attribute name="CurrencyCode" type="CurrencyCode">
<xs:annotation><xs:documentation>
Not specifying the currency code will assume that the
currency type should default to the system default
currency.
</xs:documentation></xs:annotation>
</xs:attribute>
<xs:attribute name="CompareAtPrice" type="Currency"/>
<xs:attribute name="PermanentPrice" type="Currency"/>
<xs:attribute name="IncludesTax" type="xs:boolean" default="false">
<xs:annotation><xs:documentation>
Attribute reserved for future use. To be implemented at
a future date.
</xs:documentation></xs:annotation>
</xs:attribute>
</xs:extension>
</xs:simpleContent>
</xs:complexType>

```

```
<xs:simpleType name="Class">
<xs:restriction base="xs:string">
<xs:maxLength value="10"/>
</xs:restriction>
</xs:simpleType>
<xs:simpleType name="Code">
<xs:restriction base="xs:string">
<xs:maxLength value="20"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="Currency">
<xs:restriction base="xs:decimal">
<xs:totalDigits value="10"/>
<xs:fractionDigits value="2"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="CurrencyCode">
<xs:annotation><xs:documentation>
ISO-4217 based three characters codes to specify what currency
that an amount is being specified in. Usually, if left unspecified,
the system's default (country of origin) currency type is assumed.
</xs:documentation></xs:annotation>
<xs:restriction base="xs:string">
<xs:length value="3"/>
</xs:restriction>
</xs:simpleType>

<xs:complexType name="LocalizedName">
<xs:simpleContent>
<xs:extension base="Name">
<xs:attribute name="Language" type="Language"/>
<xs:attribute name="Country" type="Country"/>
</xs:extension>
</xs:simpleContent>
</xs:complexType>

<xs:complexType name="LocalizedDescription">
<xs:simpleContent>
<xs:extension base="Description">
<xs:attribute name="Language" type="Language"/>
<xs:attribute name="Country" type="Country"/>
</xs:extension>
</xs:simpleContent>
</xs:complexType>

    <xs:simpleType name="Country">
<xs:annotation><xs:documentation>
ISO-3166 based two-character codes to specify what country
that a text is being provided in. if left unspecified, the
system's default (country of origin) country is assumed.
</xs:documentation></xs:annotation>
    <xs:restriction base="xs:string">
        <xs:length value="2" />
        <xs:enumeration value="US" />
        <xs:enumeration value="PR" />
        <xs:enumeration value="CA" />
    </xs:restriction>
```

```
</xs:simpleType>
```

```
<xs:simpleType name="Language">
<xs:annotation><xs:documentation>
ISO-639 based two-character codes to specify what language
that a text is being provided in. if left unspecified, the
system's default (country of origin) language is assumed.
</xs:documentation></xs:annotation>
  <xs:restriction base="xs:string">
    <xs:length value="2" />
    <xs:enumeration value="en" />
    <xs:enumeration value="es" />
    <xs:enumeration value="fr" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ID">
<xs:restriction base="xs:string">
<xs:maxLength value="14"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="Description">
<xs:restriction base="xs:string">
<xs:maxLength value="250"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="Name">
<xs:restriction base="xs:string">
<xs:maxLength value="40"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="ChangeType">
<xs:restriction base="xs:string">
<xs:enumeration value="ADD"/>
<xs:enumeration value="UPD"/>
<xs:enumeration value="DEL"/>
<xs:enumeration value="UPS"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="PreLoadChangeType">
  <xs:annotation><xs:documentation>
    UPSERT and DELETE are the only operations supported for
    Preload data. If "DEL" is not specified as ChangeType,
    Then "UPS" is assumed.
  </xs:documentation></xs:annotation>
  <xs:restriction base="xs:string">
    <xs:enumeration value="UPS"/>
    <xs:enumeration value="DEL"/>
  </xs:restriction>
</xs:simpleType>
```

```

<xs:simpleType name="FillType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="KillAndFill"/>
    <xs:enumeration value="DeltaIncremental"/>
    <xs:enumeration value="FullIncremental"/>
  </xs:restriction>
</xs:simpleType>

  <xs:simpleType name="RetailStoreId">
    <xs:annotation><xs:documentation>
      Store Id's can only be five characters long and preferably only
      numerals.
    </xs:documentation></xs:annotation>
    <xs:restriction base="xs:string">
      <xs:minLength value="1"/></xs:minLength>
      <xs:maxLength value="5"/></xs:maxLength>
    </xs:restriction>
  </xs:simpleType>

</xs:schema>

```

Merchandise Hierarchy Import

Table B–5 identifies the PreloadData element mapping for the MerchandiseHierarchyImport.xsd file.

Table B–5 Merchandise Hierarchy Import XSD PreloadData Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
Merchandise HierarchyGroup CO_MRHRC_GP	Merchandise HierarchyGroupID	ID_MRHRC_GP	VARCHAR(14)	PreloadData > MerchandiseGroup > ID	
	Merchant	ID_PST		PreloadData > MerchandiseGroup > MerchantID	
	Name	NM_MRHRC_GP	VARCHAR(120)	PreloadData > MerchandiseGroup > Name	
	Description	DE_MRHRC_GP	VARCHAR(250)	PreloadData > MerchandiseGroup > Description	
	RecordCreate Timestamp	TS_CRT_RCRD		Now()	
	RecordModify Timestamp	TS_MDF_RCRD		Now()	
POSDepartment ID_DPT_PS	POSDepartmentID	ID_DPT_POS	VARCHAR(14)	PreloadData > POSDepartment > POSDepartmentID	
	ParentPOS DepartmentID	ID_DPT_POS_PRNT	VARCHAR(14)	PreloadData > POSDepartment > ParentPOS DepartmentID	

Table B–5 Merchandise Hierarchy Import XSD PreloadData Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	Name	NM_DPT_POS	VARCHAR(120)	PreloadData > POSDepartment > POSDepartment Name @Text	
	TaxGroupID	ID_GP_TX		PreloadData > POSDepartment > DepartmentDefault TaxGroup	
POSDepartment I18N ID_DPT_PS_I8	POSDepartmentID	ID_DPT_POS	VARCHAR(14)	PreloadData > POSDepartment > POSDepartmentID	
	Locale	LCL	VARCHAR(10)	PreloadData > POSDepartment > POSDepartment Name @LanguageCode PreloadData > POSDepartment > POSDepartment Name @CountryCode	Concatenate Lower (Language Code)+ "_" +Upper (Country Code)
	POSDepartment Name	NM_DPT_POS	VARCHAR(40)	PreloadData > POSDepartment > POSDepartment Name @Text	
RetailStorePOS Department LO_DPT_POS_RTL_STR	RetailStoreID	ID_STR_RT	VARCHAR(5)	PreloadData > POSDepartment > RetailStorePOS Department > RetailStoreID	
	POSDepartmentID	ID_DPT_POS	VARCHAR(14)	PreloadData > POSDepartment > POSDepartmentID	
	DefaultEntryCode	FL_CD_ENT_DFLT		PreloadData > POSDepartment > RetailStorePOS Department > DefaultEntryCode	
	EnabledFlag	FL_CD_ENT_ENAB		PreloadData > POSDepartment > RetailStorePOS Department > EnabledFlag	
	ListSortIndex	CD_ENT_SRT		PreloadData > POSDepartment > RetailStorePOS Department > ListSortIndex	

Table B–6 identifies the element mapping for the MerchandiseHierarchyImport.xsd file.

Table B–6 Merchandise Hierarchy Import XSD Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
Merchandise HierarchyFunction CO_MRHRC_FNC	Merchandise Hierarchy FunctionID	ID_MRHRC_FNC		HierarchyList > Hierarchy@FunctionID	
	Name	NM_MRHRC_FNC	VARCHAR(250)	HierarchyList > Hierarchy@Name	
	RecordCreate Timestamp	TS_CRT_RCRD		Now()	
	RecordModify Timestamp	TS_MDF_RCRD		Now()	
Merchandise HierarchyLevel CO_MRHRC_LV	Merchandise Hierarchy FunctionID	ID_MRHRC_FNC		HierarchyList > Hierarchy@FunctionID	
	Merchandise Hierarchy LevelCode	ID_MRHRC_LV		HierarchyList > Hierarchy > LevelList > Level@ID	
	Parent Merchandise Hierarchy LevelID. Merchandise Hierarchy LevelCode	ID_MRHRC_LV_PRNT		HierarchyList > Hierarchy > LevelList > Level@ParentID	
	Name	NM_MRHRC_LV	VARCHAR(120)	HierarchyList > Hierarchy > LevelList > Level@Name	
	RecordCreate Timestamp	TS_CRT_RCRD		Now()	
	RecordModify Timestamp	TS_MDF_RCRD		Now()	
Merchandise Hierarchy Association ST_ASCTN_MRHRC	Merchandise Hierarchy FunctionID	ID_MRHRC_FNC		HierarchyList > Hierarchy@FunctionID	
	Parent Merchandise Hierarchy GroupID	ID_MRHRC_GP_PRNT	VARCHAR(14)	HierarchyList > Hierarchy > NodeList > Node@ParentNodeID	
	Child Merchandise Hierarchy GroupID	ID_MRHRC_GP_CHLD	VARCHAR(14)	HierarchyList > Hierarchy > NodeList > Node@ID	

Table B–6 Merchandise Hierarchy Import XSD Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	Parent Merchandise Hierarchy LevelID	ID_MRHRC_LV		HierarchyList > Hierarchy > NodeList > Node@LevelID	
	RecordCreate Timestamp	TS_CRT_RCRD		Now()	
	RecordModify Timestamp	TS_MDF_RCRD		Now()	

Table B–7 identifies the Oracle Retail Merchandising System export files mapping for the MerchandiseHierarchyImport.xsd file.

Table B–7 Merchandise Hierarchy Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Field Name	XSD Element/Attribute Path	Notes
rmse_merchhier	SUBCLASS	PreloadData > MerchandiseGroup > ID HierarchyList > Hierarchy > NodeList > Node@ID HierarchyList > Hierarchy > NodeList > Node@LevelID="6"	Point-of-Service maps all six Oracle Retail Merchandising System levels, even though it only typically uses the bottom four. Company and Division are usually assumed in a stores application. Note that SUBCLASS does not map to a ParentNodeID because the SUBCLASS level will never be any other level's parent. Note: Each ID is a MerchandiseGroup ID within the PreloadData section as well as a Node ID in the Hierarchy section.
	SUB_NAME	PreloadData > MerchandiseGroup > Name	
	CLASS	PreloadData > MerchandiseGroup > ID HierarchyList > Hierarchy > NodeList > Node@ID HierarchyList > Hierarchy > NodeList > Node@ParentNodeID HierarchyList > Hierarchy > NodeList > Node@LevelID=""5"	Class is the first level that can be defined as a parent. Any CLASS ID are specified within a SUBCLASS node definition. Higher levels continue in the same fashion.
	CLASS_NAME	PreloadData > MerchandiseGroup > Name	

Table B–7 Merchandise Hierarchy Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Field Name	XSD Element/Attribute Path	Notes
	DEPT	PreloadData > MerchandiseGroup > ID PreloadData > POSDepartment > POSDepartmentID HierarchyList > Hierarchy > NodeList > Node@ID HierarchyList > Hierarchy > NodeList > Node@ParentNodeID HierarchyList > Hierarchy > NodeList Node@LevelID="4"	Departments not only map to MerchandiseGroups like other nodes, but also map to POSDepartment data. Oracle Retail Merchandising System does not supply specific stores that this POSDepartment is valid for, so the RetailStorePOSDepartment element can be excluded and the destination store receiving this output is assumed.
	DEPT_NAME	PreloadData > MerchandiseGroup > Name PreloadData > POSDepartment > POSDepartmentName @Text	
	GROUP_NO	PreloadData > MerchandiseGroup > ID HierarchyList > Hierarchy > NodeList > Node@ID HierarchyList > Hierarchy > NodeList > Node@ParentNodeID HierarchyList > Hierarchy > NodeList > Node@LevelID="3"	
	GROUP_NAME	PreloadData > MerchandiseGroup > Name	
	DIVISION	PreloadData > MerchandiseGroup > ID HierarchyList > Hierarchy > NodeList > Node@ID HierarchyList > Hierarchy > NodeList > Node@ParentNodeID HierarchyList > Hierarchy > NodeList > Node@LevelID="2"	

Table B–7 Merchandise Hierarchy Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Field Name	XSD Element/Attribute Path	Notes
	DIV_NAME	PreloadData > MerchandiseGroup > Name	
	COMPANY	PreloadData > MerchandiseGroup > ID HierarchyList > Hierarchy > NodeList > Node@ID HierarchyList > Hierarchy > NodeList > Node@ParentNodeID HierarchyList > Hierarchy > NodeList > Node@LevelID="1"	
	CO_NAME	PreloadData > MerchandiseGroup > Name	

Example B–3 MerchandiseHierarchyImport.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <!-- $Log:$ -->

    <xs:annotation><xs:documentation>
        Merchandise Hierarchy Import Schema. Copyright 2006 Oracle.
        All rights reserved.
    </xs:documentation></xs:annotation>

    <xs:element name="MerchandiseHierarchy">
        <xs:annotation><xs:documentation>
            Top level element containing the hierarchy and the data that must be
            preloaded before the hierarchy.
        </xs:documentation></xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element name="PreloadData" type="PreloadData" minOccurs="0"
maxOccurs="1">
                    <xs:annotation><xs:documentation>
                        The data that must be preloaded into the datasource before
                        the actual hierarchy is persisted. Consists of departments
                        and merchandise groups.
                    </xs:documentation></xs:annotation>
                </xs:element>
                <xs:element name="HierarchyList" type="HierarchyList" minOccurs="0"
maxOccurs="unbounded">
                    <xs:annotation><xs:documentation>
                        The actual merchandise hierarchy data being imported.
                        Contains a grouping (list) of hierarchies.
                    </xs:documentation></xs:annotation>
                </xs:element>
            </xs:sequence>
            <xs:attribute name="FillType" type="FillType" use="required"
fixed="KillAndFill"/>
            <xs:attribute name="CreationDate" type="xs:dateTime"/>
            <xs:attribute name="ExpirationDate" type="xs:dateTime"/>
        </xs:complexType>
    </xs:element>

```



```

        <xs:attribute name="Version" type="xs:string"/>
        <xs:attribute name="Priority" type="xs:int"/>
        <xs:attribute name="Batch" type="xs:int"/>
    </xs:complexType>
</xs:element>

<xs:complexType name="PreloadData">
    <xs:sequence>
        <xs:element name="POSDepartment" type="POSDepartment" minOccurs="0"
maxOccurs="unbounded" />
        <xs:element name="MerchandiseGroup" type="MerchandiseGroup"
minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="MerchandiseGroup">
    <xs:sequence>
        <xs:element name="ChangeType" type="ChangeType" minOccurs="1"
maxOccurs="1" />
        <xs:element name="ID" type="xs:string" minOccurs="1" maxOccurs="1" />
        <xs:element name="Name" minOccurs="0" maxOccurs="1">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:maxLength value="40"/></xs:maxLength>
                </xs:restriction>
            </xs:simpleType>
        </xs:element>
        <xs:element name="MerchantID" type="xs:int" minOccurs="0"
maxOccurs="1" />
        <xs:element name="Description" type="xs:string" minOccurs="0"
maxOccurs="1" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="POSDepartment">
    <xs:sequence>
        <xs:element name="ChangeType" type="ChangeType" minOccurs="1"
maxOccurs="1" />
        <xs:element name="POSDepartmentID" type="xs:string" minOccurs="1"
maxOccurs="1" />
        <xs:element name="ParentPOSDepartmentID" type="xs:string"
minOccurs="0" maxOccurs="1" />
        <xs:element name="POSDepartmentName" type="LocalizedText"
minOccurs="0" maxOccurs="unbounded" />
        <xs:element name="DepartmentDefaultTaxGroup" type="xs:int"
minOccurs="1" maxOccurs="1" />
        <xs:element name="RetailStorePOSDepartment"
type="RetailStorePOSDepartment" minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="RetailStorePOSDepartment">
    <xs:sequence>
        <xs:element name="ChangeType" type="ChangeType" minOccurs="1"
maxOccurs="1" />
        <xs:element name="RetailStoreId" type="RetailStoreId" minOccurs="1"
maxOccurs="1" />
        <xs:element name="DefaultEntryCode" type="xs:string" minOccurs="1"
maxOccurs="1" />
    </xs:sequence>
</xs:complexType>

```

```
        <xs:element name="EnabledFlag" type="xs:boolean" minOccurs="1"
maxOccurs="1" />
        <xs:element name="ListSortIndex" type="xs:int" minOccurs="1"
maxOccurs="1" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="HierarchyList">
    <xs:sequence>
        <xs:element name="Hierarchy" type="Hierarchy" minOccurs="0"
maxOccurs="unbounded" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="Hierarchy">
    <xs:sequence>
        <xs:element name="LevelList" type="LevelList" minOccurs="0"
maxOccurs="1" />
        <xs:element name="NodeList" type="NodeList" minOccurs="0"
maxOccurs="1" />
    </xs:sequence>
    <xs:attribute name="FunctionID" type="xs:int" use="required" />
    <xs:attribute name="Name" type="xs:string"/>
</xs:complexType>

<xs:complexType name="LevelList">
    <xs:sequence>
        <xs:element name="Level" type="Level" minOccurs="0"
maxOccurs="unbounded" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="NodeList">
    <xs:sequence>
        <xs:element name="Node" type="Node" minOccurs="0"
maxOccurs="unbounded" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="Level">
<xs:attribute name="ID" type="xs:int" use="required" />
<xs:attribute name="Name" type="xs:string" />
<xs:attribute name="ParentID" type="xs:int">
    <xs:annotation><xs:documentation>
        If the parent id is missing, this is assumed to be the root.
    </xs:documentation></xs:annotation>
</xs:attribute>
</xs:complexType>

<xs:complexType name="Node">
<xs:attribute name="ID" type="xs:string" use="required" />
<xs:attribute name="Name" type="xs:string" />
<xs:attribute name="LevelID" type="xs:int" use="required" />
<xs:attribute name="ParentNodeID" type="xs:string" />
</xs:complexType>

<xs:complexType name="LocalizedText">
    <xs:annotation><xs:documentation>
        If the language and country attributes are missing, it is assumed
        that the locale is the system's default locale.
    </xs:documentation></xs:annotation>
</xs:complexType>
```

```

        </xs:documentation></xs:annotation>
<xs:attribute name="Text" use="required">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:minLength value="1" />
      <xs:maxLength value="20" />
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>
<xs:attribute name="LanguageCode">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:length value="2" />
      <xs:enumeration value="en" />
      <xs:enumeration value="es" />
      <xs:enumeration value="fr" />
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>
<xs:attribute name="CountryCode">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:length value="2" />
      <xs:enumeration value="US" />
      <xs:enumeration value="PR" />
      <xs:enumeration value="CA" />
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>
</xs:complexType>

<xs:simpleType name="ChangeType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="ADD" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="FillType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="KillAndFill"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="RetailStoreId">
  <xs:annotation><xs:documentation>
    Store Id's can only be five characters long and preferably only
    numerals.
  </xs:documentation></xs:annotation>
  <xs:restriction base="xs:string">
    <xs:minLength value="1"></xs:minLength>
    <xs:maxLength value="5"></xs:maxLength>
  </xs:restriction>
</xs:simpleType>

</xs:schema>

```

Pricing Import

Table B–8 identifies the PriceChange element mapping for the PricingImport.xsd file.

Table B–8 Pricing Import XSD PriceChange Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
Event CO_EV	EventID	ID_EV		Generated at Stores	Generated at Stores
	RetailStoreID	ID_STR_RT	VARCHAR(5)	PricingImport > PriceChange > StoreID	
	External EventID	ID_EV_EXT	VARCHAR(20)	PricingImport > PriceChange @ID	This value is used as an external index. Oracle Retail Price Management prepends a 1 for regular price changes or a 2 for clearance price changes when sending price change IDs.
	Name	NM_EV	VARCHAR(160)	PricingImport > PriceChange > Description @Text	
	TypeCode	TY_EV	VARCHAR(20)	PricingImport > PriceChange @Type	PPC = Permanent Price Change IPC = Immediate Price Change
	PlanStartDate Timestamp	TS_EV_PL_EF		PricingImport > PriceChange @StartDate	
	StatusCode	SC_EV	VARCHAR(20)	Derived from PricingImport > PriceChange @StartDate	Default = PENDING
Permanent PriceChange Item MA_ITM_ PRN_PRC_ ITM	EventID	ID_EV		Generated at Stores	Same ID as Event table
	ItemID	ID_ITM	VARCHAR(14)	PricingImport > PriceChange > Item @ID	
	RetailStoreID	ID_STR_RT	VARCHAR(5)	PricingImport > PriceChange > StoreID	
	PriceOverride Amount	MO_OVRD_PRC		PricingImport > PriceChange > Item > Price	

Table B–8 Pricing Import XSD PriceChange Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	Label TemplateID	ID_TMPLT_ LB	VARCHAR(8)	PricingImport > PriceChange > Item @TemplateType	
ItemPrice Maintenance MA_PRC_ITM	EventID	ID_EV		Generated at Stores	Same ID as Event table
	Retail StoreID	ID_STR_RT	VARCHAR(5)	PricingImport > PriceChange > StoreID	
	Label TemplateID	ID_TMPLT_ LB	VARCHAR(8)	PricingImport > PriceChange @TemplateType	
	TypeCode	TY_PRC_MNT	VARCHAR(20)	PricingImport > PriceChange @Type	PPC = Permanent Price Change IPC = Immediate Price Change
Maintenance Event CO_EV_MNT	EventID	ID_EV		Generated at Stores	Same ID as Event table.
	RetailStoreID	ID_STR_RT	VARCHAR(5)	PricingImport > PriceChange > StoreID	
	Name	NM_EV_MNT	VARCHAR(40)	PricingImport > PriceChange > Description @Text	
	TypeCode	TY_EV	VARCHAR(20)	PricingImport > PriceChange @Type	PPC = Permanent Price Change IPC = Immediate Price Change
	EffectiveDate Timestamp	TS_EV_MNT_ EF		PricingImport > PriceChange @StartDate	
	StatusCode	SC_EV_MNT	VARCHAR(20)	Derived from PricingImport > PriceChange @StartDate	Default = PENDING
Item Maintenance Event CO_MNT_ ITM	EventID	ID_EV		Generated at Stores	Same ID as Event table.
	RetailStoreID	ID_STR_RT	VARCHAR(5)	PricingImport > PriceChange > StoreID	
	FunctionCode	LU_EV_ITM_ MNT	VARCHAR(20)	No mapping available	Default = PRICE CHANGE

Table B–8 Pricing Import XSD PriceChange Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
Permanent PriceChange TR_CHN_PRN_PRC	EventID	ID_EV		Generated at Stores	Same ID as Event table.
	RetailStoreID	ID_STR_RT	VARCHAR(5)	PricingImport > PriceChange > StoreID	

Table B–9 identifies the Price Promotion element mapping for the PricingImport.xsd file.

Table B–9 Pricing Import XSD Price Promotion Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
CO_EV Event	EventID	ID_EV			The Promotion ID in this column is the Stores Promotion ID that is created in the import process. The Oracle Retail Price Management promotion ID is not updated in this column.
	RetailStoreID	ID_STR_RT	VARCHAR(5)	PricingImport > PricePromotion > StoreID	
	External EventID	ID_EV_EXT	VARCHAR(20)	PricingImport > PricePromotion @ID	The Oracle Retail Price Management promotion ID is used to derive the stores promotion ID. Stores database is altered to accommodate the Oracle Retail Price Management promotion ID.
	Name	NM_EV	VARCHAR(160)	PricingImport > PricePromotion > Name @Text	
	Description	DE_EV	VARCHAR(640)	PricingImport > PricePromotion > Description @Text	
	PlanStartDate	TS_EV_PL_EF		PricingImport > PricePromotion @StartDateTime	
	PlanEndDate	TS_EV_PL_EP		PricingImport > PricePromotion @EndDateTime	

Table B–9 Pricing Import XSD Price Promotion Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	StatusCode	SC_EV	VARCHAR(20)	No mapping found	Derived from start date.
	TypeCode	TY_EV	VARCHAR(20)	No mapping found	Default value = TPC (Temporary PriceChange)
	RecordCreation Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	
CO_EV_MNT Maintenance Event	EventID	ID_EV		Generated at Stores	Same ID as Event table.
	RetailStoreID	ID_STR_RT	VARCHAR(5)	PricingImport > PricePromotion > StoreID	
	Promotion Name	NM_EV_MNT	VARCHAR(40)	PricingImport > PricePromotion > Name @Text	
	Promotion desc	DE_EV_MNT	VARCHAR(250)	PricingImport > PricePromotion > Description @Text	
	Start date	TS_EV_MNT_EF		PricingImport > PricePromotion @StartDateTime	
	End date	TS_EV_MNT_EP		PricingImport > PricePromotion @EndDateTime	
	Status	SC_EV_MNT	VARCHAR(20)	No mapping found	Derived from start date.
	TypeCode	TY_EV_MNT		No mapping found	Default value = "TPC" for Temporary PriceChange
	RecordCreation Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	
CO_MNT_ITM Item Maintenance Event	EventID	ID_EV		Generated at Stores	Same ID as Event table.
	RetailStoreID	ID_STR_RT	VARCHAR(5)	PricingImport > PricePromotion > StoreID	

Table B–9 Pricing Import XSD Price Promotion Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	FunctionCode	LU_EV_ITM_MNT	VARCHAR(20)	No mapping found	Default value = "PRICE CHANGE"
	RecordCreation Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	
TR_CHN_TMP_PRC TemporaryPrice Change	EventID	ID_EV		Generated at Stores	Same ID as Event table.
	RetailStoreID	ID_STR_RT	VARCHAR(5)	PricingImport > PricePromotion > StoreID	
	SaleUnit Amount	MO_UN_TMP_PRC_CHN		PricingImport > PricePromotion > DiscountAmount> Amount	It can be any of the following: <ul style="list-style-type: none"> ■ Discount amount ■ Discount percent ■ New price
				PricingImport > PricePromotion > DiscountPercent	
				PricingImport > PricePromotion > NewPrice> Amount>Currency @Amount	
	SaleUnit AmountType Code	TY_UN_TMP_PRC_CHN	VARCHAR(20)	PricingImport > PricePromotion @Type	Indicator to denote: <ul style="list-style-type: none"> ■ 0= AmountOff ■ 1= PercentOff ■ 2= New Price
	RecordCreation Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	
MA_PRC_ITM ItemPrice Maintenance	EventID	ID_EV		Generated at Stores	Same ID as Event table.
	RetailStoreID	ID_STR_RT	VARCHAR(5)	PricingImport > PricePromotion > StoreID	

Table B-9 Pricing Import XSD Price Promotion Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	Priority	UN_PRI_EV		PricingImport > PricePromotion @Priority	
	Template	ID_TMPLT_LB	VARCHAR(8)	PricingImport > PricePromotion @TemplateType	"DEFAULT"
	TypeCode	TY_PRC_MNT	VARCHAR(20)	No mapping found	Default value = "TPC" for Temporary Price Change
	PriceLastDigit	UN_DG_LS_PRC		No mapping found	
	RecordCreation Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	
MA_ITM_TMP_PRC_CHN TemporaryPriceChangeItem	EventID	ID_EV		Generated at Stores	Same ID as Event table.
	RetailStoreID	ID_STR_RT	VARCHAR(5)	PricingImport > PricePromotion > StoreID	
	Item ID	ID_ITM	VARCHAR(14)	PricingImport > PricePromotion > Item @ID	Here Item ID is required, but Item occurrence can be zero , in this case the promotion details are stored without storing the item details.
	Template	ID_TMPLT_LB	VARCHAR(8)	PricingImport > PricePromotion @TemplateType	DEFAULT
	Price Override Amount	MO_OVRD_PRC		PricingImport > PricePromotion > Item > Price > Amount	
	PromotionID	ID_PRM		PricingImport > PricePromotion @ID	
	Promotion ComponentID	ID_PRM_CMP		PricingImport > PricePromotion @PromoCompID	Defaults to zero .

Table B–9 Pricing Import XSD Price Promotion Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	Promotion Component DetailID	ID_PRM_CMP_DTL		PricingImport > PricePromotion @PromoComp DetlID	Defaults to zero.
	RecordCreation Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	

Table B–10 identifies the Discount Rule element mapping for the PricingImport.xsd file.

Table B–10 Pricing Import XSD Discount Rule Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
PriceDerivationRule RU_PRDV	PriceDerivationRuleID	ID_RU_PRDV		ID from the stores system. This is not the Oracle Retail Price Management promotion ID	
	RetailStoreID	ID_STR_RT	VARCHAR(5)	DiscountRule > PricingRule > StoreID	
	PromotionID	ID_PRM		DiscountRule > PricingRule @ID	
	Promotion ComponentID	ID_PRM_CMP		DiscountRule > PricingRule @PromoCompID	No database field exists. Mapping happens once the table is created.
	Promotion Component DetailID	ID_PRM_CMP_DTL		DiscountRule > PricingRule @PromoComp DetlID	No database field exists. Mapping happens once the table is created.
	EffectiveDate	DC_RU_PRDV_EF		DiscountRule > PricingRule @StartDateTime	
	ExpirationDate	DC_RU_PRDV_EP		DiscountRule > PricingRule @EndDateTime	
	Description	DE_RU_PRDV	VARCHAR(250)	DiscountRule > PricingRule @Type	
	Assignment BasisCode	CD_BAS_PRDV		DiscountRule > Sources@Type	3=Coupon 2=Other Default it to 2

Table B-10 Pricing Import XSD Discount Rule Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	Source Comparison BasisCode	CD_BAS_CMP_SRC	VARCHAR(20)	DiscountRule > Sources@Type	0=Item 1=Department 2=Class 3=Coupon. Default it to 0.
	Target Comparison BasisCode	CD_BAS_CMP_TGT	VARCHAR(20)	DiscountRule > Targets@Type	0=Item, 1=Department, 2=Class. Default it to 0.
	Application Limit	QU_LM_APLY		DiscountRule > PricingRule @NbrTimes PerTrans	
	Department LedgerStock Modifier	DP_LDG_STK_MDFR		DiscountRule > PricingRule @Accounting Method	1 = Markdown, 0 = Discount
	AllowRepeating SourcesFlag	FL_ALW_RPT_SRC		DiscountRule > PricingRule @AllowSource ToRepeat	0= false, 1= true
	Deal Distribution Flag	FL_DL_DST		DiscountRule > PricingRule @DealDistribution	1=SourceTarget, 0=Target
	Name	NM_RU_PRDV	VARCHAR(160)	DiscountRule > PricingRule > Name @Text	
				DiscountRule > PricingRule > Name @LanguageCode	No database mapping exists. Reserved for future use.
				DiscountRule > PricingRule > Name @CountryCode	No database mapping exists. Reserved for future use.
				DiscountRule > PricingRule > SourceThreshold @CurrencyCode	No database mapping exists. Reserved for future use.
	Source Threshold Amount	MO_TH_SRC		DiscountRule > PricingRule > SourceThreshold	
				DiscountRule > PricingRule > SourceLimit @CurrencyCode	No database mapping exists. Reserved for future use.
	SourceLimit Amount	MO_LM_SRC		DiscountRule > PricingRule > SourceLimit	

Table B–10 Pricing Import XSD Discount Rule Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
				DiscountRule > PricingRule > TargetThreshold @CurrencyCode	No database mapping exists. Reserved for future use.
	TargetThresholdAmount	MO_TH_TGT		DiscountRule > PricingRule > TargetThreshold	
				DiscountRule > PricingRule > TargetLimit @CurrencyCode	No database mapping exists. Reserved for future use.
	TargetLimitAmount	MO_LM_TGT		DiscountRule > PricingRule > TargetLimit	
	SourceAnyQuantity	QU_AN_SRC		DiscountRule > Sources @Qty	The Any Quantity is only populated if Sources@Qualifier is set to Any .
	TargetAnyQuantity	QU_AN_TGT		DiscountRule > Targets @Qty	The Any Quantity is only populated if Targets@Qualifier is set to Any .
	RecordCreationTimestamp	TS_CRT_RCRD		Now()	
	RecordLastModifiedTimestamp	TS_MDF_RCRD		Now()	
ItemPrice Derivation RuleEligibility CO_EL_PRDV_ITM	ItemID	ID_ITM	VARCHAR(14)	DiscountRule > Sources > Source @ID	
	PriceDerivationRuleEligibilityID	ID_RU_PRDV		DiscountRule > PricingRule @ID	
	RetailStoreID	ID_STR_RT	VARCHAR(5)	DiscountRule > PricingRule > StoreID	
	ThresholdQuantity	QU_TH		DiscountRule > Sources > Source @Qty	
				DiscountRule > Sources > SourceAmount @CurrencyCode	No database mapping exists. Reserved for future use.
	ThresholdAmount	MO_TH		DiscountRule > Sources > Source > SourceAmount	

Table B–10 Pricing Import XSD Discount Rule Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	EffectiveDate Timestamp	TS_RU_ DRVN_EF		DiscountRule > PricingRule @StartDateTime	
	ExpirationDate Timestamp	TS_RU_ DRVN_EP		DiscountRule > PricingRule @EndDateTime	
	RecordCreation Timestamp	TS_CRT_ RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_ RCRD		Now()	
MixAndMatch PriceDerivation Item TR_ITM_ MXMH_PRDV	PriceDerivation RuleID	ID_RU_PRDV		DiscountRule > PricingRule @ID	
	RetailStoreID	ID_STR_RT	VARCHAR(5)	DiscountRule > PricingRule > StoreID	
	Promotional ProductID	ID_PRM_PRD	VARCHAR(14)	DiscountRule > Targets > Target @ID	
				DiscountRule > Targets > DiscountAmount @CurrencyCode	No database mapping exists. Reserved for future use.
	Reduction Monetary Amount	MO_RDN_ PRC_MXMH		DiscountRule > Targets > DiscountAmount	
	Reduction Percent	PE_RDN_ PRC_MXMH		DiscountRule > Targets > DiscountPercent	
				DiscountRule > Targets > NewPrice @CurrencyCode	No database mapping exists. Reserved for future use.
	ReductionPrice Point	PNT_PRC_ RDN_MXMH		DiscountRule > Targets > NewPrice	
	MixAndMatch LimitCount	QU_LM_ MXMH		DiscountRule > Targets > Target @Qty	
	RecordCreation Timestamp	TS_CRT_ RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_ RCRD		Now()	

Table B–10 Pricing Import XSD Discount Rule Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
ItemPrice Derivation CO_PRDV_ITM	RetailStoreID	ID_STR_RT	VARCHAR(5)	DiscountRule > PricingRule > StoreID	
	PriceDerivation RuleID	ID_RU_PRDV		DiscountRule > PricingRule @ID	
	Reduction Amount	MO_UN_ITM_PRDV		DiscountRule > Targets > DiscountAmount	
	Reduction Percent	PE_UN_ITM_PRDV		DiscountRule > Targets > DiscountPercent	
	DiscountPrice Point	PNT_PRC_UN_ITM_PRDV		DiscountRule > Targets > NewPrice	
	RecordCreation Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	
MixAndMatch PriceDerivation Rule RU_PRDV_MXMH	RetailStoreID	ID_STR_RT	VARCHAR(5)	DiscountRule > PricingRule > StoreID	
	PriceDerivation RuleID	ID_RU_PRDV		DiscountRule > PricingRule @ID	
	MixAndMatch LimitCount	QU_LM_MXMH		DiscountRule > Targets > Target @Qty	
DepartmentPrice Derivation RuleEligibility CO_EL_PRDV_DPT Populated only if DiscountRule > Sources > Source @type is "Department"	POSDepartment ID	ID_DPT_POS	VARCHAR(14)	DiscountRule > Sources > Source @ID	Might be derived from the table ID_DPT_PS column ID_DPT_POS
	PriceDerivation RuleID	ID_RU_PRDV		ID from the stores system. This is not the Oracle Retail Price Management promotion ID.	
	RetailStoreID	ID_STR_RT	VARCHAR(5)	Store ID	
	StoreFinancial Ledger AccountID	ID_ACTN_LDG		No mapping available	-1

Table B-10 Pricing Import XSD Discount Rule Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	EventID	ID_EV		No mapping available	-1
	Accounting Disposition Code	DP_ACNT_DPT_PRDV		No mapping available	
	Threshold Amount	MO_TH		DiscountRule > Sources > Source > SourceAmount	
	Threshold Quantity	QU_TH		DiscountRule > Sources > Source @Qty	
	LimitQuantity	QU_UL		No mapping available	
	LimitAmount	MO_UL		No mapping available	
	Effective Timestamp	TS_RU_MRST_EF		DiscountRule > PricingRule @StartDateTime	
	Expiration Timestamp	TS_RU_MRST_EP		DiscountRule > PricingRule @EndDateTime	
	RecordCreated Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	
Merchandise StructurePrice Derivation RuleEligibility CO_EL_MRST_PRDV Populated only if DiscountRule > Sources > Source @type is "Class"	PriceDerivation RuleID	ID_RU_PRDV		ID from the stores system. This is not the Oracle Retail Price Management promotion ID.	
	RetailStoreID	ID_STR_RT	VARCHAR(5)	Store ID	
	Merchandise Classification Code	ID_STRC_MR_CD	VARCHAR(10)	DiscountRule > Sources > Source @ID	Might be derived from the table LU_CD_STRC_MR column ID_STRC_MR_CD
	StoreFinancial Ledger AccountID	ID_ACTN_LDG		No mapping available	-1
	EventID	ID_EV		No mapping available	-1

Table B–10 Pricing Import XSD Discount Rule Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	EffectiveDate Timestamp	TS_RU_MRST_EF		DiscountRule > PricingRule @StartDateTime	
	ExpirationDate Timestamp	TS_RU_MRST_EP		DiscountRule > PricingRule @EndDateTime	
	Accounting Disposition Code	DP_ACNT_MRST		No mapping available	null
	Threshold Amount	MO_TH		DiscountRule > Sources > Source > SourceAmount	
	Quantity Threshold	QU_TH		DiscountRule > Sources > Source @Qty	
	AmountLimit	MO_UL		No mapping available	0
	QuantityLimit	QU_UL		No mapping available	0
	RecordCreated Timestamp	TS_CRT_RCRD		Now()	
	RecordLast Modified Timestamp	TS_MDF_RCRD		Now()	

[Table B–11](#) identifies the Oracle Retail Merchandising System export files element mapping for the PricingImport.xsd file.

Table B–11 Pricing Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Record Header	Field Name	Transform	XSD Element/Attribute Path	Notes
REGPC	FHEAD	Record Descriptor			
		Line Id			
		File Type			
		Export Timestamp		PricingImport @CreationDate	
		Format Version		PricingImport @Version	
	FDETL	Record Descriptor			
		Line Id			
		Event Type		PricingImport > PriceChange @ChangeType	
		Id		PricingImport > PriceChange @ID	
		Item		PricingImport > PriceChange > Item @ID	

Table B–11 Pricing Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Record Header	Field Name	Transform	XSD Element/Attribute Path	Notes
		Location		PricingImport > PriceChange > StoreID	
		LocationType			
		Effective Date		PricingImport > PriceChange @StartDate	
		Selling Unit Change Ind			
		Selling Retail		PricingImport > PriceChange > Item > Price	
		Selling Retail UOM			
		Selling Retail Currency		PricingImport > PriceChange > Item > Price @CurrencyCode	
		Multi-Unit Change Ind			
		Multi-Units	trim()		
		Multi-Unit Retail	trim()		
		Multi-Unit UOM			
		Multi-Unit Currency			
	FDELE	Record Descriptor			
		Line Id			
		Id		PricingImport > PriceChange > Item @ID	
		Item		PricingImport > PriceChange > Item @ID	
		Location		PricingImport > PriceChange > StoreID	
		LocationType			
	FTAIL	Record Descriptor			
		Line Id			
		Number of lines			
CLRPC	FHEAD	Record Descriptor			
		Line Id			
		File Type			
		Export Timestamp		PricingImport @CreationDate	
		Format Version		PricingImport @Version	

Table B–11 Pricing Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Record Header	Field Name	Transform	XSD Element/Attribute Path	Notes
	FDETL	Record Descriptor			
		Line Id			
		Event Type		PricingImport > PriceChange @ChangeType	
		Id		PricingImport > PriceChange @ID	
		Item		PricingImport > PriceChange > Item @ID	
		Location		PricingImport > PriceChange > StoreID	
		LocationType			
		Effective Date		PricingImport > PriceChange @StartDate	
		Selling Retail		PricingImport > PriceChange > Item > Price	
		Selling Retail UOM			
		Selling Retail Currency		PricingImport > PriceChange > Item > Price @CurrencyCode	
		ResetClearanceId			
	FDELE	Record Descriptor			
		Line Id			
		Id		PricingImport > PriceChange > Item @ID	
		Item		PricingImport > PriceChange > Item @ID	
		Location		PricingImport > PriceChange > StoreID	
		LocationType			
	FTAIL	Record Descriptor			
		Line Id			
		Number of lines			
PRMPC	FHEAD	Record Descriptor			
		Line Id			
		File Type			
		Export Timestamp		PricingImport @CreationDate	
		Format Version		PricingImport @Version	

Table B–11 Pricing Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Record Header	Field Name	Transform	XSD Element/Attribute Path	Notes
	TSMPP	Record Descriptor			
		Line Id			
		Event Type		PricingImport > PricePromotion @ChangeType	
		Promotion Id		PricingImport > PricePromotion @Id	
		Promo Comp Id		PricingImport > PricePromotion @PromoCompId	
		Promo Comp Detl Id		PricingImport > PricePromotion @PromoCompDetlId	
		Start Date		PricingImport > PricePromotion @StartDateTime	
		End Date		PricingImport > PricePromotion @EndDateTime	
		Promo Name		PricingImport > PricePromotion > Name @Text	
		Promo Desc		PricingImport > PricePromotion > Description @Text	
		Promo Comp Name			
		Apply Order			
		Change Type		PricingImport > PricePromotion @Type	
		Change Value		PricingImport > PricePromotion > DiscountPercent PricingImport > PricePromotion > DiscountAmount PricingImport > PricePromotion > Item > Price	
		Change Value UOM			
	TSDTL	Record Descriptor			
		Line Id			
		Item Id		PricingImport > PricePromotion > Item @ID	
		Location Id		PricingImport > PricePromotion > StoreID	

Table B–11 Pricing Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Record Header	Field Name	Transform	XSD Element/Attribute Path	Notes
		Location Type			
		Selling Retail		PricingImport > PricePromotion > Item > Price	
		Selling Retail UOM			
	TTAIL	Record Descriptor			
		Line Id			
	FSDEL	Record Descriptor			
		Line Id			
		Promo Comp Detail Id		PricingImport > PricePromotion @PromoCompDetlId	
		Item Id		PricingImport > PricePromotion > Item @ID	
		Location Id		PricingImport > PricePromotion > StoreID	
		Location Type			
PRMPC	TTHRP	Record Descriptor			
		Line Id			
		Event Type		PricingImport > DiscountRule > PricingRule @ChangeType	
		Promotion Id		PricingImport > DiscountRule > PricingRule @Id	
		Promo Comp Id		PricingImport > DiscountRule > PricingRule @PromoCompId	
		Promo Comp Detl Id		PricingImport > DiscountRule > PricingRule @PromoCompDetlId	
		Start Date		PricingImport > DiscountRule > PricingRule @StartDateTime	
		End Date		PricingImport > DiscountRule > PricingRule @EndDateTime	
		Promo Name		PricingImport > DiscountRule > PricingRule > Name @Text	
		Promo Desc			

Table B–11 Pricing Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Record Header	Field Name	Transform	XSD Element/Attribute Path	Notes
		Promo Comp Name			
		Apply Order			
		Threshold Id			
		Threshold Name			
		Threshold Qual Type			
		Threshold Type			
		Change Type		PricingImport > DiscountRule > PricingRule @Type	Possible values: <ul style="list-style-type: none"> ■ BuyNoFXforZ%off ■ BuyNoFXforZ\$off ■ BuyNoFXforZ\$
	TTLVL	Record Descriptor			
		Line Id			
		Threshold Value		PricingImport > DiscountRule > Sources > Source @qty	
		Change Value		PricingImport > DiscountRule > Targets > DiscountPercent PricingImport > DiscountRule > Targets > DiscountAmount PricingImport > DiscountRule > Targets > NewPrice	
		Change Value UOM			
	TTDTL	Record Descriptor			
		Line Id			
		Item Id		PricingImport > DiscountRule > Sources > Source @ID	
		Location Id		PricingImport > DiscountRule > StoreID	
		Location Type			
	TTAIL	Record Descriptor			
		Line Id			
	FTDEL	Record Descriptor			
		Line Id			

Table B–11 Pricing Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Record Header	Field Name	Transform	XSD Element/Attribute Path	Notes
		Promo Comp Detl Id		PricingImport > DiscountRule > PricingRule @PromoCompDetlId	
		Item Id		PricingImport > DiscountRule > Sources > Source @ID	
		Location Id		PricingImport > DiscountRule > StoreID	
		Location Type			
PRMPC	TBGTP	Record Descriptor			
		Line Id			
		Event Type		PricingImport > DiscountRule > Pricing Rule @ChangeType	
		Promotion Id		PricingImport > DiscountRule > PricingRule @Id	
		Promo Comp Id		PricingImport > DiscountRule > PricingRule @PromoCompId	
		Promo Comp Detl Id		PricingImport > DiscountRule > PricingRule @PromoCompDetlId	
		Start Date		PricingImport > DiscountRule > PricingRule @StartDateTime	
		End Date		PricingImport > DiscountRule > PricingRule @EndDateTime	
		Promo Name		PricingImport > DiscountRule > Pricing Rule > Name @Text	
		Promo Desc			
		Promo Comp Name			
		Apply Order			
		AllIndicator			
		Buy Quantity		PricingImport > DiscountRule > Sources > Source @qty	

Table B–11 Pricing Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Record Header	Field Name	Transform	XSD Element/Attribute Path	Notes
		Change Type		PricingImport > DiscountRule > PricingRule @Type	Possible values (not all are supported by Oracle Retail Price Management): <ul style="list-style-type: none"> ■ BuyNofXgetYatZ%off ■ BuyNofXgetYatZ\$off ■ BuyNofXgetYatZ\$ ■ BuyNofXgetHighestPricedXatZ%off ■ BuyNofXgetLowestPricedXatZ%off ■ Buy\$NorMoreOfXgetYatZ\$off ■ Buy\$NorMoreOfXgetYatZ%off ■ Buy\$NorMoreOfXgetYatZ\$
		Change Value		PricingImport > DiscountRule > Targets > DiscountPercent PricingImport > DiscountRule > Targets > DiscountAmount PricingImport > DiscountRule > Targets > NewPrice	
		Change Value UOM			
	TBITM	Record Descriptor			
		Line Id			
		Buy Item Id		PricingImport > DiscountRule > Sources > Source @ID	
	TGITM	Record Descriptor			
		Line Id			
		Get Item Id		PricingImport > DiscountRule > Targets > Target @ID	
	TLOCN	Record Descriptor			
		Line Id			
		Location Id		PricingImport > DiscountRule > StoreID	
		Location Type			
	TTAIL	Record Descriptor			

Table B–11 Pricing Import XSD Oracle Retail Merchandising System Export Files Mapping Table

File Name	Record Header	Field Name	Transform	XSD Element/Attribute Path	Notes
		Line Id			
	FBDEL	Record Descriptor			
		Line Id			
		Promo Comp Detail Id		PricingImport > DiscountRule > PricingRule @PromoCompDetId	
		Item Id		PricingImport > DiscountRule > Sources > Source @ID	
		Location Id		PricingImport > DiscountRule > StoreID	
		Location Type			
	FTAIL	Record Descriptor			
		Line Id			
		Number of lines			

Example B–4 PricingImport.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">

  <!-- $Log:$ -->

  <xs:annotation><xs:documentation>
    Pricing Import Schema. Copyright 2007 Oracle Inc. All rights reserved.

    Use this schema in conjunction with a Oracle Store Systems Data Dictionary
    and the relations between the element and attribute names should be
    apparent.
  </xs:documentation></xs:annotation>

  <xs:element name="PricingImport">
    <xs:annotation><xs:documentation>
      Top-level element holding a collection of Price records.
    </xs:documentation></xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element name="PriceChange" type="PriceChange" minOccurs="0"
maxOccurs="unbounded"/>
        <xs:element name="PricePromotion" type="PricePromotion" minOccurs="0"
maxOccurs="unbounded"/>
        <xs:element name="DiscountRule" type="DiscountRule" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="FillType" type="FillType" use="required"/>
      <xs:attribute name="CreationDate" type="xs:dateTime"/>
      <xs:attribute name="ExpirationDate" type="xs:dateTime"/>
      <xs:attribute name="Version" type="xs:string"/>
      <xs:attribute name="Priority" type="xs:int"/>
    </xs:complexType>
  </xs:element>

```



```

        <xs:attribute name="Batch" type="xs:int"/>
    </xs:complexType>
</xs:element>

<xs:complexType name="PriceChange">
    <xs:sequence>
        <xs:element name="Description" type="LocalizedText" minOccurs="0"
maxOccurs="1"/>
        <xs:element name="Item" type="ItemAndPrice" minOccurs="1"
maxOccurs="unbounded" />
        <xs:element name="StoreID" type="RetailStoreId" minOccurs="0"
maxOccurs="unbounded" />
    </xs:sequence>
    <xs:attribute name="ChangeType" type="ChangeType" default="ADD"/>
    <xs:attribute name="ID" type="xs:string" use="required"/>
    <xs:attribute name="StartDate" type="xs:date"/>
    <xs:attribute name="TemplateType" type="xs:string"/>
</xs:complexType>

<xs:complexType name="ItemAndPrice">
<xs:sequence>
<xs:element name="Price" type="Amount" minOccurs="1" maxOccurs="unbounded" />
</xs:sequence>
<xs:attribute name="ID" type="xs:string" use="required"/>
<xs:attribute name="TemplateType" type="xs:string"/>
</xs:complexType>

<xs:group name="DiscountTypeChoice">
    <xs:choice>
        <xs:element name="DiscountPercent" type="xs:decimal"/>
        <xs:element name="DiscountAmount" type="Amount"/>
        <xs:element name="NewPrice" type="Amount"/>
    </xs:choice>
</xs:group>

<xs:attributeGroup name="PromotionComponentAttributes">
<xs:attribute name="PromoCompID" type="xs:int" use="optional"/>
<xs:attribute name="PromoCompDetlID" type="xs:int" use="optional"/>
</xs:attributeGroup>

<xs:complexType name="PricePromotion">
<xs:sequence>
<xs:element name="Name" type="LocalizedText" minOccurs="1" maxOccurs="1"/>
<xs:element name="Description" type="LocalizedText" minOccurs="0" maxOccurs="1"/>
<xs:group ref="DiscountTypeChoice" minOccurs="0" maxOccurs="1"/>
<xs:element name="Item" type="ItemAndPrice" minOccurs="0" maxOccurs="unbounded" />
<xs:element name="StoreID" type="RetailStoreId" minOccurs="0"
maxOccurs="unbounded" />
</xs:sequence>
<xs:attribute name="ChangeType" type="ChangeType" default="ADD"/>
<xs:attribute name="ID" type="xs:string" use="required"/>
<xs:attributeGroup ref="PromotionComponentAttributes"/>
<xs:attribute name="StartDateTime" type="xs:dateTime"/>
<xs:attribute name="EndDateTime" type="xs:dateTime"/>
<xs:attribute name="Type" type="PricePromotionType" use="required"/>
<xs:attribute name="Priority" type="xs:int" default="0"/>
<xs:attribute name="TemplateType" type="xs:string"/>
</xs:complexType>

<xs:simpleType name="PricePromotionType">

```

```

        <xs:restriction base="xs:string">
            <xs:enumeration value="AmountOff"/>
            <xs:enumeration value="PercentOff"/>
            <xs:enumeration value="NewPrice"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:complexType name="DiscountRule">
        <xs:sequence>
            <xs:element name="PricingRule" type="PricingRule" minOccurs="1"
maxOccurs="1"/>
            <xs:element name="Sources" type="Sources" minOccurs="1" maxOccurs="1"/>
            <xs:element name="Targets" type="Targets" minOccurs="1" maxOccurs="1"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="PricingRule">
        <xs:sequence>
            <xs:element name="Name" type="LocalizedText" minOccurs="1" maxOccurs="1"/>
            <xs:element name="SourceThreshold" type="Amount" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="SourceLimit" type="Amount" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="TargetThreshold" type="Amount" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="TargetLimit" type="Amount" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="StoreID" type="RetailStoreId" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="ChangeType" type="ChangeType" default="ADD"/>
        <xs:attribute name="ID" type="xs:int" use="required"/>
        <xs:attributeGroup ref="PromotionComponentAttributes"/>
        <xs:attribute name="StartDateTime" type="xs:dateTime" use="required"/>
        <xs:attribute name="EndDateTime" type="xs:dateTime" use="required"/>
        <xs:attribute name="Type" type="RuleTypeType" use="required"/>
        <xs:attribute name="NbrTimesPerTrans" type="xs:int" default="-1"/>
        <xs:attribute name="AccountingMethod" type="AccountingMethodType"
default="Discount" />
        <xs:attribute name="AllowSourceToRepeat" type="xs:boolean" default="true"/>
        <xs:attribute name="DealDistribution" type="DealDistributionType"
default="Target"/>
        <xs:attribute name="Scope" type="ScopeType" default="Item" />
    </xs:complexType>

    <xs:attributeGroup name="SourceTargetAttributes">
        <xs:attribute name="Type" type="SourceTargetType" default="Item" />
        <xs:attribute name="Qualifier" type="QualifierType" default="Any"/>
        <xs:annotation><xs:documentation>
            If not specified, it is assumed that the Qualifier is Any.
        </xs:documentation></xs:annotation>
    </xs:attribute>
    <xs:attribute name="Qty" type="xs:int" default="1">
        <xs:annotation><xs:documentation>
            It is only necessary to specify Qty if Qualifier has been
            set to Any. If not specified, it is assumed that Qty for
            Any is one (1).
        </xs:documentation></xs:annotation>
    </xs:attribute>
</xs:attributeGroup>

```

```

<xs:complexType name="Sources">
  <xs:sequence>
    <xs:element name="Source" minOccurs="1" maxOccurs="unbounded">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="SourceAmount" type="Amount" minOccurs="0" maxOccurs="unbounded" />
        />
      </xs:sequence>
      <xs:attribute name="ID" type="xs:string" use="required" />
      <xs:attribute name="Qty" type="xs:int" use="required" />
    </xs:complexType>
  </xs:element>
</xs:sequence>
<xs:attributeGroup ref="SourceTargetAttributes"/>
</xs:complexType>

<xs:complexType name="Targets">
  <xs:sequence>
    <xs:group ref="DiscountTypeChoice" minOccurs="1" maxOccurs="1"/>
    <xs:element name="Target" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:attribute name="ID" type="xs:string" use="required"/>
        <xs:attribute name="Qty" type="xs:int" default="1"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
<xs:attributeGroup ref="SourceTargetAttributes"/>
</xs:complexType>

<xs:simpleType name="RuleTypeType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="BuyNoFXgetYatZ%off"/>
    <xs:enumeration value="BuyNoFXgetYatZ$off"/>
    <xs:enumeration value="BuyNoFXgetYatZ$"/>
    <xs:enumeration value="BuyNoFXgetHighestPricedXatZ%off"/>
    <xs:enumeration value="BuyNoFXgetLowestPricedXatZ%off"/>
    <xs:enumeration value="Buy$NorMoreOfXgetYatZ$off"/>
    <xs:enumeration value="Buy$NorMoreOfXgetYatZ$off"/>
    <xs:enumeration value="Buy$NorMoreOfXgetYatZ$"/>
    <xs:enumeration value="BuyNoFXforZ$"/>
    <xs:enumeration value="BuyNoFXforZ%off"/>
    <xs:enumeration value="BuyNoFXforZ$off"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="AccountingMethodType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Discount"/>
    <xs:enumeration value="Markdown"/>
  </xs:restriction>
</xs:simpleType>

  <xs:simpleType name="DealDistributionType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="Target"/>
      <xs:enumeration value="SourceTarget"/>
    </xs:restriction>
  </xs:simpleType>

```

```

<xs:simpleType name="ScopeType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Item"/>
    <xs:enumeration value="Group"/>
    <xs:enumeration value="Transaction"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="SourceTargetType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Item"/>
    <xs:enumeration value="Coupon"/>
    <xs:enumeration value="Class"/>
    <xs:enumeration value="Department"/>
  </xs:restriction>
</xs:simpleType>

<xs:complexType name="Amount">
  <xs:annotation><xs:documentation>
    An array of multiple values; each a different currency type.
  </xs:documentation></xs:annotation>
  <xs:simpleContent>
    <xs:extension base="Currency">
      <xs:attribute name="CurrencyCode" type="CurrencyCode">
        <xs:annotation><xs:documentation>
          Not specifying the currency code will assume that the
          currency type should default to the system default
          currency.
        </xs:documentation></xs:annotation>
      </xs:attribute>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>

<xs:simpleType name="Currency">
  <xs:restriction base="xs:decimal">
    <xs:totalDigits value="10"/>
    <xs:fractionDigits value="2"/>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="CurrencyCode">
  <xs:annotation><xs:documentation>
    ISO-4217 based three characters codes to specify what currency
    that an amount is being specified in. Usually, if left unspecified,
    the system's default (country of origin) currency type is assumed.
  </xs:documentation></xs:annotation>
  <xs:restriction base="xs:string">
    <xs:length value="3" />
  </xs:restriction>
</xs:simpleType>

<xs:complexType name="LocalizedText">
  <xs:annotation><xs:documentation>
    If the language and country attributes are missing, it is assumed
    that the locale is the system's default locale.
  </xs:documentation></xs:annotation>
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="Language" type="Language"/>
    </xs:extension>
  </xs:simpleContent>

```

```

<xs:attribute name="Country" type="Country"/>
</xs:extension>
</xs:simpleContent>
</xs:complexType>

<xs:simpleType name="Language">
  <xs:annotation><xs:documentation>
    ISO-639 based two characters codes to specify what language
    that a text is being provided in. Usually, if left unspecified,
    the system's default (country of origin) language is assumed.
  </xs:documentation></xs:annotation>
  <xs:restriction base="xs:string">
    <xs:length value="2" />
    <xs:enumeration value="en" />
    <xs:enumeration value="es" />
    <xs:enumeration value="fr" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="Country">
  <xs:annotation><xs:documentation>
    ISO-3166 based two characters codes to specify what country
    that a text is being provided in. Usually, if left unspecified,
    the system's default (country of origin) country is assumed.
  </xs:documentation></xs:annotation>
  <xs:restriction base="xs:string">
    <xs:length value="2" />
    <xs:enumeration value="US" />
    <xs:enumeration value="PR" />
    <xs:enumeration value="CA" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="FillType">
<xs:restriction base="xs:string">
  <xs:enumeration value="KillAndFill"/>
  <xs:enumeration value="DeltaIncremental"/>
  <xs:enumeration value="FullIncremental"/>
</xs:restriction>
</xs:simpleType>

  <xs:simpleType name="RetailStoreId">
<xs:annotation><xs:documentation>
Store Id's can only be five characters long and preferably only
numerals.
</xs:documentation></xs:annotation>
<xs:restriction base="xs:string">
  <xs:minLength value="1"></xs:minLength>
  <xs:maxLength value="5"></xs:maxLength>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="QualifierType">
  <xs:annotation><xs:documentation>
    Used to qualify a list whereby Any element in the list must be
    used versus requiring All elements in the list.
  </xs:documentation></xs:annotation>
  <xs:restriction base="xs:string">
<xs:enumeration value="Any"></xs:enumeration>
<xs:enumeration value="All"></xs:enumeration>

```

```

</xs:restriction>
</xs:simpleType>

<xs:simpleType name="ChangeType">
  <xs:annotation><xs:documentation>
Whether the record in question is being ADDED, UPdated, or DELETED.
  </xs:documentation></xs:annotation>
  <xs:restriction base="xs:string">
    <xs:enumeration value="ADD" />
    <xs:enumeration value="UPD" />
    <xs:enumeration value="DEL" />
  </xs:restriction>
</xs:simpleType>

</xs:schema>

```

Store Hierarchy Import

Table B-12 identifies the PreloadData element mapping for the StoreHierarchyImport.xsd file.

Table B-12 Store Hierarchy Import XSD Preload Data Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
StoreRegions LO_STR_RGN	RegionID	ID_STR_RGN(LO_STR_DSTRCT, PA_STR_RTL)	VARCHAR(14)	PreloadData > StoreRegion > RegionID	
	RegionName	NM_STR_RGN	VARCHAR(120)	PreloadData > StoreRegion > RegionName	
	RecordCreate Timestamp	TS_CRT_RCRD		Now()	
	RecordModify Timestamp	TS_MDF_RCRD		Now()	
StoreDistricts LO_STR_DSTRCT	DistrictID	ID_STR_DISTRICT	VARCHAR(14)	PreloadData > StoreDistrict > DistrictID	
	RegionID	ID_STR_RGN	VARCHAR(14)	PreloadData > StoreDistrict > RegionID	
	DistrictName	NM_STR_DSTRCT	VARCHAR(120)	PreloadData > StoreDistrict > DistrictName	
	RecordCreate Timestamp	TS_CRT_RCRD		Now()	
	RecordModify Timestamp	TS_MDF_RCRD		Now()	
RetailStore PA_STR_RTL	RetailStoreID	ID_STR_RT	VARCHAR(5)	PreloadData > RetailStore > RetailStoreID	

Table B-12 Store Hierarchy Import XSD Preload Data Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	TaxIdentificationNumber1	ID_IDTN_TX_NMB1	VARCHAR(100)	PreloadData > RetailStore > TaxIdentificationNumber1	
	TaxIdentificationNumber2	ID_IDTN_TX_NMB2	VARCHAR(100)	PreloadData > RetailStore > TaxIdentificationNumber2	
	TaxIdentificationNumber3	ID_IDTN_TX_NMB3	VARCHAR(100)	PreloadData > RetailStore > TaxIdentificationNumber3	
	TaxIdentificationNumber4	ID_IDTN_TX_NMB4	VARCHAR(100)	PreloadData > RetailStore > TaxIdentificationNumber4	
	TaxIdentificationNumber5	ID_IDTN_TX_NMB5	VARCHAR(100)	PreloadData > RetailStore > TaxIdentificationNumber5	
	LocationName	NM_LOC	VARCHAR(150)	PreloadData > RetailStore > LocationName	
	DistrictID	ID_STR_DSTRCT	VARCHAR(14)	PreloadData > RetailStore > DistrictID	
	RegionID	ID_STR_RGN	VARCHAR(14)	PreloadData > RetailStore > RegionID	
	GeoCode	ID_CD_GEO	VARCHAR(10)	PreloadData > RetailStore > GeoCode	
	RecordCreateTimestamp	TS_CRT_RCRD		Now()	
	RecordModifyTimestamp	TS_MDF_RCRD		Now()	
Address LO_ADS	AddressID	ID_ADS		PreloadData > RetailStore > Address > AddressID	
	AddressTypeCode	TY_ADS	VARCHAR(30)	PreloadData > RetailStore > Address > AddressTypeCode	Home=0 Work=0 Mail=3 Other=2
	PartyID	ID_PRTY			Derive from TY_ADS
	ContactAddressLine1	A1_CNCT	VARCHAR(30)	PreloadData > RetailStore > Address > AddressLine1	

Table B–12 Store Hierarchy Import XSD Preload Data Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	ContactAddress Line2	A2_CNCT	VARCHAR(30)	PreloadData > RetailStore > Address > AddressLine2	
	ContactAddress Line3	A3_CNCT	VARCHAR(30)	PreloadData > RetailStore > Address > AddressLine3	
	ContactAddress City	CI_CNCT	VARCHAR(30)	PreloadData > RetailStore > Address > City	
	ContactAddress State	ST_CNCT	VARCHAR(30)	PreloadData > RetailStore > Address > State	
	ContactAddress PostalCode	PC_CNCT	VARCHAR(30)	PreloadData > RetailStore > Address > PostalCode	
	ContactAddress Territory	TE_CNCT	VARCHAR(30)	PreloadData > RetailStore > Address > Territory	
	ContactAddress Country	CO_CNCT	VARCHAR(30)	PreloadData > RetailStore > Address > Country	
	Contact Telephone CountryCode	CC_CNCT	VARCHAR(30)	PreloadData > RetailStore > Address > TelephoneCountryCode	
	Contact Telephone AreaCode	TA_CNCT	VARCHAR(30)	PreloadData > RetailStore > Address > TelephoneAreaCode	
	Contact Telephone LocalNumber	TL_CNCT	VARCHAR(30)	PreloadData > RetailStore > Address > TelephoneLocalNumber	

Table B–13 identifies the element mapping for the StoreHierarchyImport.xsd file.

Table B–13 Store Hierarchy Import XSD Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
RetailStoreGroup Function CO_STRGP_FNC	RetailStore Group FunctionID	ID_STRGP_ FNC		HierarchyList > Hierarchy@ FunctionID	
	RetailStore GroupFunction Name	NM_STRGP_ FNC	VARCHAR(120)	HierarchyList > Hierarchy@Name	
	RecordCreate Timestamp	TS_CRT_ RCRD		Now()	
	RecordModify Timestamp	TS_MDF_ RCRD		Now()	
RetailStore GroupLevel CO_STRGP_LV	RetailStore Group FunctionID	ID_STRGP_ FNC		HierarchyList > Hierarchy@ FunctionID	
	StoreHierarchy LevelID	ID_STRGP_LV		HierarchyList > Hierarchy > LevelList > Level@ID	
	RetailStore GroupLevel Parent	ID_STRGP_ LV_PRNT		HierarchyList > Hierarchy > LevelList > Level@ParentID	
	RetailStore GroupLevel Name	NM_STRGP_ LV	VARCHAR(120)	HierarchyList > Hierarchy > LevelList > Level@Name	
	RecordCreate Timestamp	TS_CRT_ RCRD		Now()	
	RecordModify Timestamp	TS_MDF_ RCRD		Now()	
AssociatedRetail StoreGroup ST_ASCTN_STRGP	RetailStore Group FunctionID	ID_STRGP_ FNC		HierarchyList > Hierarchy@ FunctionID	
	RetailStore GroupParentID	ID_STRGP_ PRNT		HierarchyList > Hierarchy > NodeList > Node@ParentNode ID	
	RetailStore GroupChildID	ID_STRGP_ CHLD		HierarchyList > Hierarchy > NodeList > Node@ID	
	RecordCreate Timestamp	TS_CRT_ RCRD		Now()	
	RecordModify Timestamp	TS_MDF_ RCRD		Now()	

Table B–13 Store Hierarchy Import XSD Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
AssociatedRetailStoreStoreGroup ST_ASCTN_STRGP_STR	RetailStoreID	ID_STR_RT	VARCHAR(5)	HierarchyList > Hierarchy > NodeList > RetailStoreID	
	RetailStore GroupID	ID_STRGP	VARCHAR(14)	HierarchyList > Hierarchy > NodeList > Node@ID	
	RetailStore Group FunctionID	ID_STRGP_FNC		HierarchyList > Hierarchy@FunctionID	
	RecordCreate Timestamp	TS_CRT_RCRD		Now()	
	RecordModify Timestamp	TS_MDF_RCRD		Now()	
RetailStoreGroupCO_STRGP	RetailStore GroupID	ID_STRGP	VARCHAR(14)	HierarchyList > Hierarchy > NodeList > Node@ID	
	RetailStore Group FunctionID	ID_STRGP_FNC		HierarchyList > Hierarchy@FunctionID	
	ParentStore Hierarchy LevelID	ID_STRGP_LV		HierarchyList > Hierarchy > NodeList > Node@LevelID	
	RetailStore GroupName	NM_STRGP	VARCHAR(120)	HierarchyList > Hierarchy > NodeList > Node@Name	
	RetailStore Group Description	DE_STRGP	VARCHAR(250)	HierarchyList > Hierarchy > NodeList > Node@Descripton	
	RecordCreate Timestamp	TS_CRT_RCRD		Now()	
	RecordModify Timestamp	TS_MDF_RCRD		Now()	
StoreHierarchyST_STR_HRY	StoreHierarchy GroupID	ID_STRGP	VARCHAR(14)	HierarchyList > Hierarchy > NodeList > Node@ID	
	RetailStoreID	ID_STR_RT	VARCHAR(5)	HierarchyList > Hierarchy > NodeList > RetailStoreID	

Table B-13 Store Hierarchy Import XSD Element Mapping Table

Log/Physical table	Target	Physical Column Name	Maximum Column Size	XSD Element/Attribute Path	Notes
	RecordCreate Timestamp	TS_CRT_RCRD		Now()	
	RecordModify Timestamp	TS_MDF_RCRD		Now()	

Example B-5 StoreHierarchyImport.xsd

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">

  <!-- $Log:$ -->

  <xs:annotation><xs:documentation>
    Store Hierarchy Import Schema. Copyright 2006 Oracle.
    All rights reserved.
  </xs:documentation></xs:annotation>

  <xs:element name="StoreHierarchy">
    <xs:annotation><xs:documentation>
      Top level element containing the hierarchy and the data that must be
      preloaded before the hierarchy.
    </xs:documentation></xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element name="PreloadData" type="PreloadData" minOccurs="0"
maxOccurs="1">
          <xs:annotation>
            <xs:documentation>
              The data that must be preloaded into the datasource
              before the actual hierarchy is persisted.
              Consists of regions, districts and stores.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="HierarchyList" type="HierarchyList" minOccurs="0"
maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>
              The actual store hierarchy data being imported. Contains
              a grouping (list) of hierarchies.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="FillType" type="FillType" use="required"
fixed="KillAndFill"/>
      <xs:attribute name="CreationDate" type="xs:dateTime"/>
      <xs:attribute name="ExpirationDate" type="xs:dateTime"/>
      <xs:attribute name="Version" type="xs:string"/>
      <xs:attribute name="Priority" type="xs:int"/>
      <xs:attribute name="Batch" type="xs:int"/>
    </xs:complexType>
  </xs:element>

```

```
<xs:complexType name="PreloadData">
  <xs:sequence>
    <xs:element name="StoreRegion" type="StoreRegion" minOccurs="0"
maxOccurs="unbounded" />
    <xs:element name="StoreDistrict" type="StoreDistrict" minOccurs="0"
maxOccurs="unbounded" />
    <xs:element name="RetailStore" type="RetailStore" minOccurs="0"
maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="StoreRegion">
  <xs:sequence>
    <xs:element name="ChangeType" type="ChangeType" maxOccurs="1"
minOccurs="1" />
    <xs:element name="RegionID" type="xs:string" maxOccurs="1"
minOccurs="1" />
    <xs:element name="RegionName" type="xs:string" maxOccurs="1"
minOccurs="0" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="StoreDistrict">
  <xs:sequence>
    <xs:element name="ChangeType" type="ChangeType" maxOccurs="1"
minOccurs="1" />
    <xs:element name="DistrictID" type="xs:string" maxOccurs="1"
minOccurs="1" />
    <xs:element name="RegionID" type="xs:string" maxOccurs="1"
minOccurs="1" />
    <xs:element name="DistrictName" type="xs:string" maxOccurs="1"
minOccurs="0" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="RetailStore">
  <xs:sequence>
    <xs:element name="ChangeType" type="ChangeType" maxOccurs="1"
minOccurs="1" />
    <xs:element name="RetailStoreID" type="RetailStoreId" maxOccurs="1"
minOccurs="1" />
    <xs:element name="LocationName" type="xs:string" maxOccurs="1"
minOccurs="0" />
    <xs:element name="DistrictID" type="xs:string" maxOccurs="1"
minOccurs="0" />
    <xs:element name="RegionID" type="xs:string" maxOccurs="1"
minOccurs="0" />
    <xs:element name="GeoCode" type="xs:string" maxOccurs="1"
minOccurs="0" />
    <xs:element name="Address" type="Address" maxOccurs="1" minOccurs="0"
/>
    <xs:element name="TaxIdentificationNumber1" type="xs:string" maxOccurs="1"
minOccurs="0" />
    <xs:element name="TaxIdentificationNumber2" type="xs:string" maxOccurs="1"
minOccurs="0" />
    <xs:element name="TaxIdentificationNumber3" type="xs:string" maxOccurs="1"
minOccurs="0" />
    <xs:element name="TaxIdentificationNumber4" type="xs:string" maxOccurs="1"
minOccurs="0" />
```

```

<xs:element name="TaxIdentificationNumber5" type="xs:string" maxOccurs="1"
minOccurs="0"/>

    </xs:sequence>
</xs:complexType>

<xs:complexType name="Address">
    <xs:sequence>
        <xs:element name="AddressID" type="xs:int" maxOccurs="1"
minOccurs="1"/>
        <xs:element name="AddressTypeCode" maxOccurs="1" minOccurs="1">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:enumeration value="Home"></xs:enumeration>
                    <xs:enumeration value="Work"></xs:enumeration>
                    <xs:enumeration value="Mail"></xs:enumeration>
                    <xs:enumeration value="Other"></xs:enumeration>
                </xs:restriction>
            </xs:simpleType>
        </xs:element>
        <xs:element name="AddressLine1" type="xs:string" maxOccurs="1"
minOccurs="1"/>
        <xs:element name="AddressLine2" type="xs:string" maxOccurs="1"
minOccurs="0"/>
        <xs:element name="AddressLine3" type="xs:string" maxOccurs="1"
minOccurs="0"/>
        <xs:element name="City" type="xs:string" maxOccurs="1" minOccurs="1"/>
        <xs:element name="State" type="xs:string" maxOccurs="1"
minOccurs="0"/>
        <xs:element name="PostalCode" type="xs:string" maxOccurs="1"
minOccurs="1"/>
        <xs:element name="Territory" type="xs:string" maxOccurs="1"
minOccurs="0"/>
        <xs:element name="Country" type="xs:string" maxOccurs="1"
minOccurs="0"/>
        <xs:element name="TelephoneCountryCode" type="xs:string" maxOccurs="1"
minOccurs="0"/>
        <xs:element name="TelephoneAreaCode" type="xs:string" maxOccurs="1"
minOccurs="0"/>
        <xs:element name="TelephoneLocalNumber" type="xs:string" maxOccurs="1"
minOccurs="0"/>
    </xs:sequence>
</xs:complexType>

<xs:complexType name="HierarchyList">
    <xs:sequence>
        <xs:element name="Hierarchy" type="Hierarchy" minOccurs="0"
maxOccurs="unbounded" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="Hierarchy">
    <xs:sequence>
        <xs:element name="LevelList" type="LevelList" minOccurs="0"
maxOccurs="1" />
        <xs:element name="NodeList" type="NodeList" minOccurs="0"
maxOccurs="1" />
    </xs:sequence>
    <xs:attribute name="FunctionID" type="xs:int" use="required" />
    <xs:attribute name="Name" type="xs:string"/>

```

```

</xs:complexType>

<xs:complexType name="LevelList">
  <xs:sequence>
    <xs:element name="Level" type="Level" minOccurs="0"
maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="NodeList">
  <xs:sequence>
    <xs:element name="Node" type="Node" minOccurs="0"
maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="Level">
  <xs:attribute name="ID" type="xs:int" use="required">
    <!--

=====
          RESTRICTION 1:
          The following restriction may be imposed if we want to limit the
number of level IDs in the store
          hierarchy. The enumeration will contain the level IDs starting
from zero, and will correspond with
          the number of levels within the store hierarchy.

=====
    <xs:simpleType>
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="0"/>
        <xs:enumeration value="1"/>
        <xs:enumeration value="2"/>
        <xs:enumeration value="3"/>
      </xs:restriction>
    </xs:simpleType>
  -->
</xs:attribute>
<xs:attribute name="Name" type="xs:string">
  <!--

=====
          RESTRICTION 2:
          The following restriction may be imposed if we want to limit the
number of levels in the store
          hierarchy. The enumeration will contain the store hierarchy level
names, which should have a
          corresponding level ID in the attribute, above.

=====
    <xs:simpleType>
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="Level1"/>
        <xs:enumeration value="Level2"/>
        <xs:enumeration value="Level3"/>
        <xs:enumeration value="root"/>
      </xs:restriction>
    </xs:simpleType>
  -->

```

```

</xs:attribute>
<xs:attribute name="ParentID" type="xs:int">
  <xs:annotation><xs:documentation>
    If the parent id is missing, this is assumed to be the root.
  </xs:documentation></xs:annotation>
<!--

=====
      RESTRICTION 3:
      The following restriction may be imposed to tie a specific parent
level to the current node
      within the store hierarchy. Ensure that the IDs defined in
RESTRICTION 1 will correspond to the
      IDs defined in the enumeration of this restriction.

=====
      <xs:simpleType>
        <xs:restriction base="xs:NMTOKEN">
          <xs:enumeration value="0"/>
          <xs:enumeration value="1"/>
          <xs:enumeration value="2"/>
        </xs:restriction>
      </xs:simpleType>
    -->
  </xs:attribute>
</xs:complexType>

<xs:complexType name="Node">
  <xs:sequence>
    <xs:element name="RetailStoreId" type="RetailStoreId" minOccurs="0"
maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="ID" type="xs:int" use="required" />
  <xs:attribute name="Name" type="xs:string" />
  <xs:attribute name="Description" type="xs:string" />
  <xs:attribute name="LevelID" type="xs:int" use="required">
    <!--

=====
      RESTRICTION 4:
      The following restriction may be imposed if we want to limit the
number of levels within
      the store hierarchy. The number of levels should correspond with
the number of level
      IDs imposed by RESTRICTION 1.

=====
      <xs:simpleType>
        <xs:restriction base="xs:NMTOKEN">
          <xs:enumeration value="0"/>
          <xs:enumeration value="1"/>
          <xs:enumeration value="2"/>
          <xs:enumeration value="3"/>
        </xs:restriction>
      </xs:simpleType>
    -->
  </xs:attribute>
  <xs:attribute name="ParentNodeID" type="xs:int"/>
</xs:complexType>

```

```

<xs:simpleType name="RetailStoreId">
  <xs:annotation><xs:documentation>
    Store Id's can only be five characters long and preferably only
    numerals.
  </xs:documentation></xs:annotation>
  <xs:restriction base="xs:string">
    <xs:minLength value="1"></xs:minLength>
    <xs:maxLength value="5"></xs:maxLength>
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ChangeType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="ADD" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="FillType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="KillAndFill"/>
  </xs:restriction>
</xs:simpleType>

</xs:schema>

```

Tax Flat File Specification

When the data is downloaded from RMS via *POSDNLD_TAX.PC* batch program, the tax assignment details are included in the download file shown below.

All the inputs comes from the TAX_ASSIGNMENT table of RMS. Most columns that are NULL will default to blank (spaces).

Table B-14 Tax Flat File Specification

Record Name	Field Name	Field Type	Default Value	Description
File Header	File Type Record Descriptor	Char (5)	FHEAD	Identifies file record type.
	File Line Identifier	Number (10)	0000000001	ID of current line being created for output file.
	File Type Definition	Char (4)	POST	Identifies file as 'POS Download Tax'.
	File Create Date	Char (8)		Vdate, formatted to 'YYYYMMDD'.

Table B-14 (Cont.) Tax Flat File Specification

Record Name	Field Name	Field Type	Default Value	Description
File Detail	File Type Record Descriptor	Char (5)	FDETL	Identifies file record type.
	File Line Identifier	Number (10)	Sequential number. Created by program.	ID of current line being created for output file.
	Location Number	Number (10)	Store	Contains the store location that has been affected by the transaction.
	Tax Category	Number (4)	Tax_category	Contains the Tax Category
	Tax Authority	Number (4)	Tax_authority	Contains the Tax Authority
	VAT Region	Number (4)	vat_region	Contains the VAT Region for the Store
	Taxpayer Type	Number (4)	taxpayer_type	Contains the taxpayer type for the store
	VAT Code	Char (6)	vat_code	This field contains the VAT Code in the assignment
	VAT code description	Char (120)	vat_code_desc	This field contains the VAT Code description
	Applied on	Char (500)	applied_on	This field contains the Formula on the tax assignment
	Application Order	Number (6)	application_ord	This field identified in which order the formulas need to be calculated
File Trailer	VAT Rate	Number (20,10)	vat_rate	This field contains the VAT_Rate that is active as of the next business day.
	File Type Record Descriptor	Char (5)	FTAIL	Identifies file record type
	Line number	Number (10)		Sequential file line number (total # lines in file)
	Number of transactions	Number (10)		Number of transactions in file

