

Retek[®] Data Warehouse[™] 11.0

Operations Guide

Corporate Headquarters:

Retek Inc.
Retek on the Mall
950 Nicollet Mall
Minneapolis, MN 55403
USA

888.61.RETEK (toll free US)
Switchboard:
+1 612 587 5000

Fax:
+1 612 587 5100

European Headquarters:

Retek
110 Wigmore Street
London
W1U 3RW
United Kingdom

Switchboard:
+44 (0)20 7563 4600

Sales Enquiries:
+44 (0)20 7563 46 46

Fax:
+44 (0)20 7563 46 10

The software described in this documentation is furnished under a license agreement, is the confidential information of Retek Inc., and may be used only in accordance with the terms of the agreement.

No part of this documentation may be reproduced or transmitted in any form or by any means without the express written permission of Retek Inc., Retek on the Mall, 950 Nicollet Mall, Minneapolis, MN 55403, and the copyright notice may not be removed without the consent of Retek Inc.

Information in this documentation is subject to change without notice.

Retek provides product documentation in a read-only-format to ensure content integrity. Retek Customer Support cannot support documentation that has been changed without Retek authorization.

The functionality described herein applies to this version, as reflected on the title page of this document, and to no other versions of software, including without limitation subsequent releases of the same software component. The functionality described herein will change from time to time with the release of new versions of software and Retek reserves the right to make such modifications at its absolute discretion.

Retek[®] Data Warehouse[™] is a trademark of Retek Inc.

Retek and the Retek logo are registered trademarks of Retek Inc.

This unpublished work is protected by confidentiality agreement, and by trade secret, copyright, and other laws. In the event of publication, the following notice shall apply:

©2004 Retek Inc. All rights reserved.

All other product names mentioned are trademarks or registered trademarks of their respective owners and should be treated as such.

Printed in the United States of America.

Customer Support

Customer Support hours

Customer Support is available 7x24x365 via email, phone, and Web access.

Depending on the Support option chosen by a particular client (Standard, Plus, or Premium), the times that certain services are delivered may be restricted. Severity 1 (Critical) issues are addressed on a 7x24 basis and receive continuous attention until resolved, for all clients on active maintenance. Retek customers on active maintenance agreements may contact a global Customer Support representative in accordance with contract terms in one of the following ways.

Contact Method	Contact Information
----------------	---------------------

E-mail	support@retек.com
--------	-------------------

Internet (ROCS)	rocs.retek.com Retek's secure client Web site to update and view issues
-----------------	---

Phone	+1 612 587 5800
-------	-----------------

Toll free alternatives are also available in various regions of the world:

Australia	+1 800 555 923 (AU-Telstra) or +1 800 000 562 (AU-Optus)
France	0800 90 91 66
Hong Kong	800 96 4262
Korea	00 308 13 1342
United Kingdom	0800 917 2863
United States	+1 800 61 RETEK or 800 617 3835

Mail	Retek Customer Support Retek on the Mall 950 Nicollet Mall Minneapolis, MN 55403
------	---

When contacting Customer Support, please provide:

- Product version and program/program 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.

Contents

Chapter 1 – Introduction	1
What is RDW and data warehousing?	2
Technical architecture	3
Dimension processing	4
Fact processing	5
Process to update records in RDW	6
Where you can find more information	6
Chapter 2 – Dimension data concepts	7
An overview of RDW dimension processing	7
Dimensions	9
Major changes and lower-level dimensions	10
Minor changes and top-level dimensions	11
Actions during processing	11
Maintenance columns in the DM table	12
Keys and identifiers	12
Next_key_val	12
As-was vs. as-is	12
Pushdowns	13
An overview of RDW dimension processing flows	13
Data preparation for lower-level dimensions	13
Data preparation for lower-level dimensions flow description	16
Major and minor change capture for lower-level dimensions flow	17
Major and minor change capture for lower-level dimensions flow description	19
Processing for the regular top-level dimensions	20
Regular top-level dimension processing data flow description	22
Processing for special top-level dimensions	23
Special top-level dimension processing data flow description	24
Datamart table	24
Dimension datamart (DM) table	25

Chapter 3 – Fact data concepts 27

An overview of RDW fact processing	27
Fact functional areas	29
Fact table types: base and aggregate	30
Fact temp table usage.....	30
General fact processing.....	30
Detailed fact load description	32
Base fact loading process flow description	34
Fact aggregation.....	35
Positional fact aggregation	35
Standard fact aggregation	37
Fact aggregation from a base fact table flow diagram.....	37
Fact aggregation from a base fact table flow description	38
Fact aggregation from another aggregate fact table diagram	40
Fact aggregation from another aggregate fact table description.....	41
Derived datamarts.....	42

Chapter 4 – Compression and partitioning 45

Overview of compression	45
What compression does.....	45
The mechanics of compression	46
Compressed tables and CUR tables.....	47
Coping with major changes	48
Partitioning for the Oracle client only	49
Overview of partitioning strategies	49
Implementing RDW partitioning.....	50
Partitioning strategy and requirements for MicroStrategy 7	52
An example of setup and maintenance for partitioning RDW's compressed inventory tables using warehouse partitioning	54
How Oracle implements partitions.....	57
Summary	58

Chapter 5 – RDW program overview 59

Program features	59
Program return code	59
Restart and recovery	59
RDW restart and recovery	59
Message logging.....	60
Program error file	61
Schema files	62
Resource files	62
Command line parameters.....	63
Partitioning	63
Temporary directory	63
The first time RDW batch is run.....	63
Typical mt_prime.ksh run	64
Typical run and debugging situations	65
RDW dimension load	65
RDW fact load.....	66

Chapter 6 – RDW interfaces 69

Retek Merchandising System	71
Dimension data.....	71
Fact data	71
Retek Invoice Matching (ReIM).....	72
Retek Price Management (RPM)	72
Retek Sales Audit (ReSA)	72
Retek Merchandise Financial Planning	73
Retek Customer Order Management	73
Dimension data.....	73
Fact data	73
Client-supplied data	74

Chapter 7 – Program flow diagrams 75

Batch scheduling	75
Setting up the batch schedule.....	75
LANGUAGE variable	75
rdw_config.env settings	76
Merchandise Financial Planning to RDW scheduling	76
Data from undefined sources	77
RDW batch schedule for DB2 clients only	77
Program flow diagrams	77
Legend: RDW dimension programs	78
Legend: RDW fact programs	85

Chapter 8 – Program reference lists 95

Dimension programs	95
Fact programs.....	108
Maintenance programs.....	124
Program type and operation type descriptions.....	132
Dimension types	132
Fact types.....	140
Maintenance types.....	153

Appendix A – Application programming interface (API) flat file specifications 155

API format	155
File layout.....	155
General business rules and standards common to all APIs	156
Flat file specifications	158

Chapter 1 – Introduction

Retek Data Warehouse (RDW) version contains enhanced localization functionality and works in conjunction with the Retek Extract Transform and Load (RETL) 11.2.1 framework. This architecture optimizes a high performance data processing tool that lets database batch processes take advantage of parallel processing capabilities. In addition, RDW can be extended beyond its traditional reliance upon Oracle to IBM's DB2 Universal Database (UDB) and NCR's Teradata.

With the implementation of RETL, the RDW client benefits from the following capabilities:

- Database Independence: Allows RDW to be deployed on different database platforms
- Parallel computing technology:
 - Promotes the flexibility of a stand-alone solution
 - Lets database batch processes take full advantage of parallel processing capabilities
 - Increases scalability, leveraging parallel processing of both the system and database server (reads, writes, performs transformations and aggregations)
- Expanded use of Application Programming Interfaces (API): Allows for easier customization
- Elimination of table triggers: Reduces the burden on the source system
- Extensible Markup Language (XML) scripts: Facilitate the framework's ability to process fact and dimension data by using valid operators
- Streamlined ETL Code: Provides for less data storage, easier implementation, and reduced maintenance requirements through decreased code volume and complexity

What is RDW and data warehousing?

A data warehouse is a physical place, a database, where you can place data from a transactional system, such as Retek Merchandising System (RMS), for the purpose of querying that data. In order to work with RDW, you start by populating it with existing data from source systems such as RMS, Retek Sales Audit (ReSA), and Retek Merchandise Financial Planning.

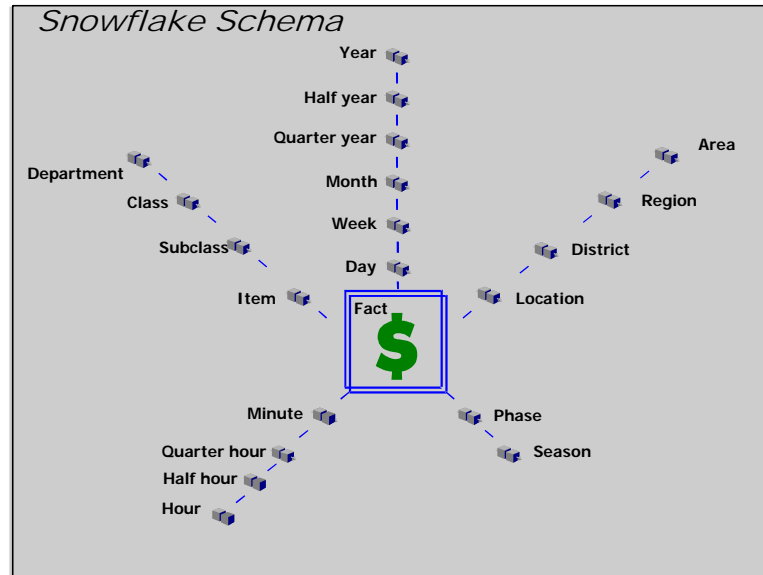
RDW uses very sophisticated techniques to populate the data warehouse. Explained in greater detail throughout this guide, these techniques include taking the data provided by source systems (such as RMS) and then rapidly transforming that data and loading it into the data warehouse. Techniques used to load data into the warehouse vary depending upon whether the data consists of ‘facts’ or ‘dimensions’.

Understanding the differences between fact and dimension data depends first upon understanding data processing in a data warehouse. RDW uses an online analytical processing (OLAP) application that serves as an interface to your data, giving it meaning through pre-designed and custom queries and reports. The data warehouse itself supports these queries by structuring data in a useful schema. Note that the word ‘schema’ in this context is an industry-standard term that refers to the way in which data is modeled and organized throughout a data warehouse and should not be confused with the ‘schema files’ that are described later in this document. (More information about schema files can be found in the latest RETL Programmer’s Guide.)

At the center of this schema is fact data. Facts are the transactions that occur in your data warehouse’s source systems, such as RMS. You might want to look at sales transaction facts, or inventory stock count facts at stores or warehouses, or inventory movement facts.

Facts have little meaning by themselves because they are usually just values, for example: 6 sales at a store, 15 items left at a warehouse, or 300 items transferred. What gives fact data true meaning in RDW is the intersection of dimensions in which facts exist. In other words, 6 sales on Wednesday at store B, or 15 dishwashers in stock last Monday at the Chicago warehouse, or 300 blouses transferred during the last week in February from the St. Louis warehouse to the Denver warehouse. Dimension data, therefore, exists in the data warehouse to serve as reference data to facts.

The schema of a data warehouse illustrates its data elements and their inter-relationships. The following graphic describes the schema used in RDW:



Snowflake schema in RDW

RDW's schema, the 'snowflake schema', starts out as a star with a fact in the middle surrounded by rays pointing out from the center. These points are the dimension data that give meaning to the fact by serving as points of reference.

RDW contains far greater volumes of fact data than it does dimension data. Besides being more abundant than dimensions, facts change constantly as new data enters the database. Dimension data, on the other hand, changes much less frequently. New stores need to be added into the data warehouse much less frequently than new sales transactions (fact data) that need to be processed daily. Because of the different natures of fact and dimension data, RDW employs different techniques to load and manipulate the data.

The dimension and fact processing sections located later in this chapter illustrate the differences in these two processes, both of which contribute to the success of RDW as your data warehouse. A more detailed description of dimension and fact processing concepts continues throughout the next two chapters.

Technical architecture

The primary goal of the RETL architecture is to take advantage of enhanced parallel processing capabilities and at the same time provide a database independent solution that runs streamlined code. The RETL framework runs and parses through the valid operators composed in XML scripts.

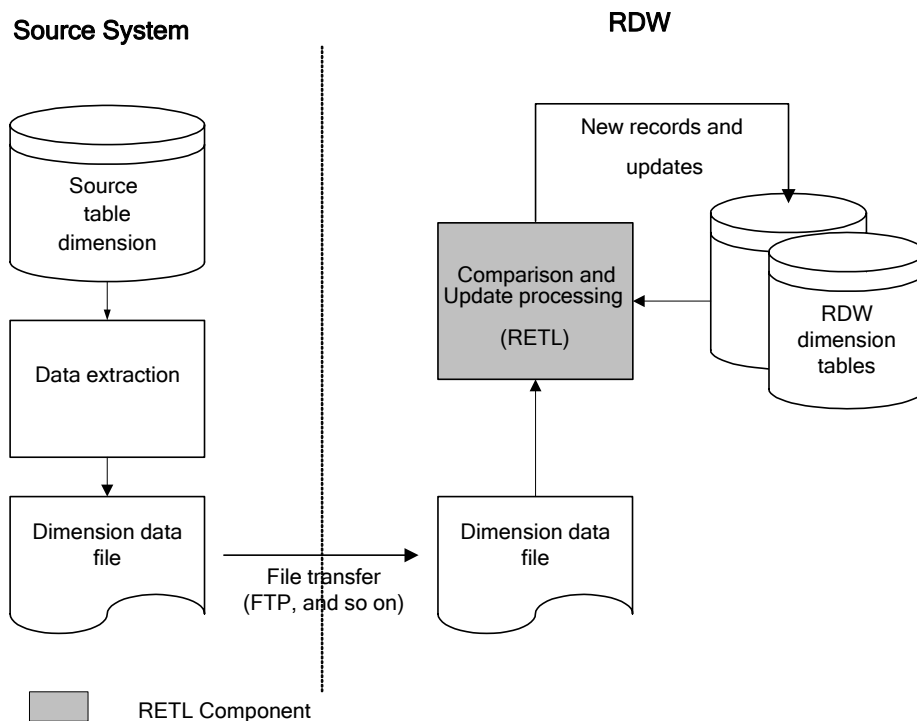
In this section, three features of RDW are described: dimension processing, fact processing, and the process to update records.

Dimension processing

The following diagram illustrates the dimension processing architecture that is employed in RDW. The process involves comparing a dimension data file that contains the current snapshot of the applicable source system table with the historical data in RDW. This comparison eliminates the need to capture frequent dimension changes as they occur in the source system over the course of the day. The comparison is performed on the RETL framework and written back directly to the datamart tables in RDW. The source dimension data file can be created for retailers that have the Retek source applications by running the data extraction programs that are packaged with the applications. Retailers without these source applications must provide the data files from their source system in the format recognized by RDW (see “Appendix A – Application Programming Interface (API) flat file specifications”).



Note: Data extraction programs are available for retailers with Retek source applications (for example, RMS, RCOM, ReSA, Merchandise Financial Planning, ReIM, and RPM). These programs are packaged with the applications.



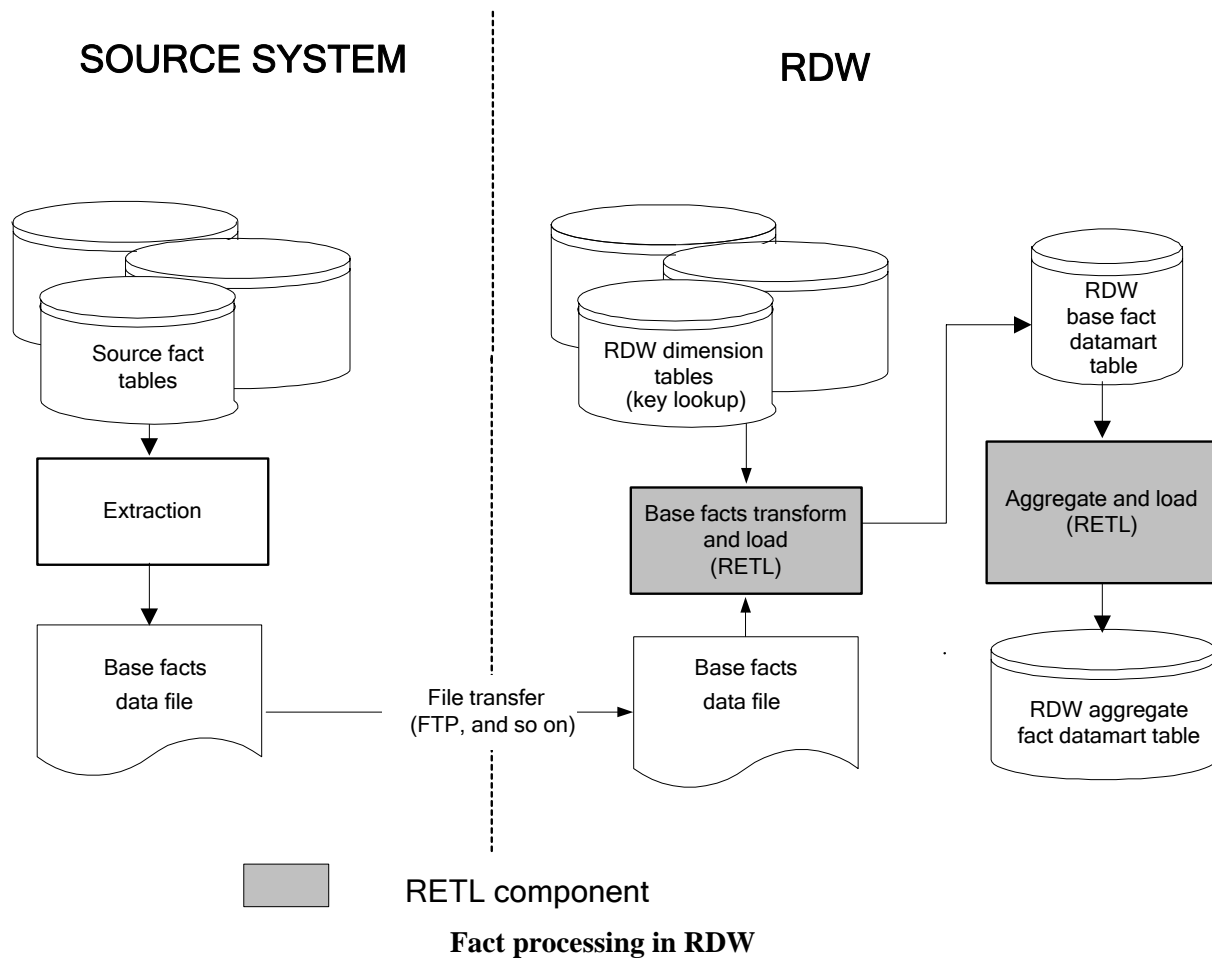
Dimension processing in RDW

Fact processing

Fact data provided via a base fact data file is transformed, aggregated, and loaded directly into RDW datamart tables without the need of staging tables. The source fact data file can be created for retailers that have Retek source applications by running the data extraction programs that are packaged with these applications. Retailers without the Retek source applications must provide data files from their source system in the format recognized by RDW (see “Appendix A – Application Programming Interface (API) flat file specifications”).



Note: Data extraction and transformation programs are available for retailers with Retek source applications (for example, RMS, RCOM, ReSA, Merchandise Financial Planning, ReIM, and RPM).



Process to update records in RDW

Because RETL does not currently support a database update operator, the actual updates into the database are accomplished through one of two processes, depending upon whether a normal update or an incremental update is occurring. A normal update is one that uses incoming records to replace old records in the target table. An incremental update (applicable to fact processing only) is one that sums the incoming records with the old records in the target table and replaces those old records with the new summed records.



Note: The temporary tables that are mentioned throughout this operations guide are always dropped by the batch code every day, after the various batch processes that use the temporary tables complete.

Normal update description

- 1 The dataset (containing the new records) is written into a temporary table.
- 2 This temporary table is used to determine which of the old update records in the target table should be deleted.
- 3 The old records are deleted from the target table.
- 4 The new records are inserted into the target table.

Incremental update description (applicable to fact processing only)

- 1 The dataset (containing the new records) is written to a temporary table.
- 2 The records to be updated are read from the target table and a second temporary table (temporary table 2) is created.
- 3 The temporary table 2 is used to determine which of the old update records in the target table should be deleted, and those records are deleted.
- 4 The records in the temporary table and in temporary table 2 are combined to form a new dataset.
- 5 The new dataset is grouped by the primary keys of the target table to sum up the required fact fields.
- 6 The resulting dataset is written to the target table (that is, the records are inserted into the target table).

Where you can find more information

You can find more information about RDW in these resources:

- RDW 11.0 Data Model
- RDW 11.0 Installation Guides
- The latest RETL Programmer's Guide
- RDW 11.0 User Guide
- RDW online help

Chapter 2 – Dimension data concepts

This chapter describes how RDW processes dimension data from the source system or systems. This chapter presents the following dimension data concepts in RDW:

- An overview of dimension data processing
- The dimensions in RDW
- Detailed dimension processing flows

An overview of RDW dimension processing

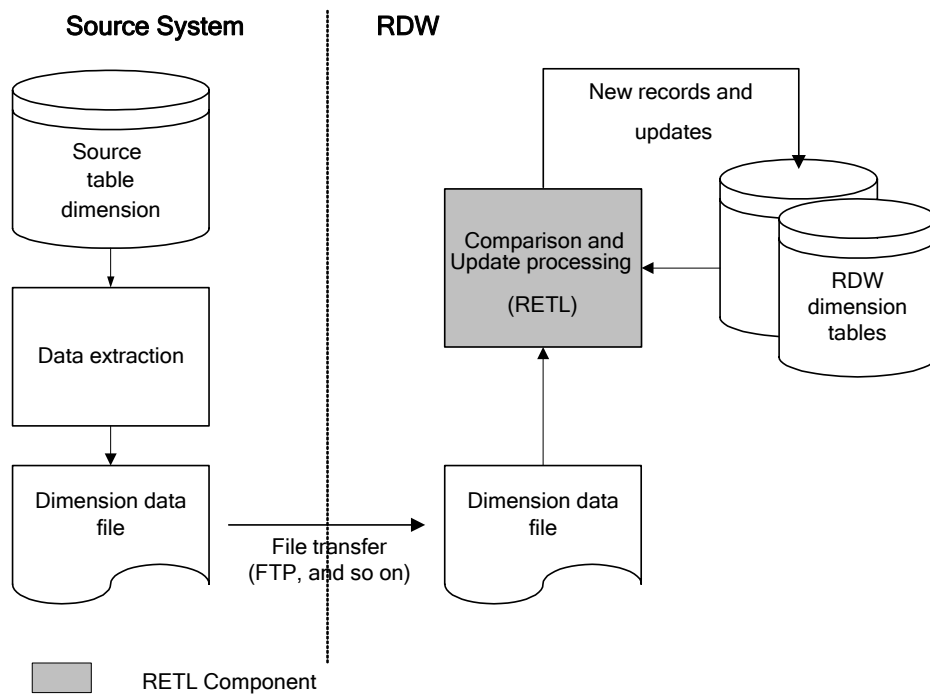
The following description and “Dimension processing in RDW” diagram offer an overview of RDW’s dimension process.

Note that there is an exception to this extraction process (see the section, “Processing for special top-level dimensions”, later in this chapter. The data file is transferred to the RDW server using a common data transfer process such as FTP.

The process involves comparing a dimension data file that contains the current snapshot of the applicable source system table with the historical data in RDW. This comparison eliminates the need to capture frequent dimension changes as they occur in the source system over the course of the day. The comparison is performed on the RETL framework and written back directly to the datamart tables in RDW. The source dimension data file can be created for retailers that have the Retek source applications by running the data extraction programs that are packaged with the applications. Retailers without these source applications must provide the data files from their source system in the format recognized by RDW (see “Appendix A – Application Programming Interface (API) flat file specifications”).



Note: Data extraction programs are available for retailers with Retek source applications (for example, RMS, RCOM, ReSA, Merchandise Financial Planning, ReIM, and RPM). These programs are packaged with the applications.



Dimension processing in RDW

Dimensions

RDW consists of the dimensions shown in this table. Product and organization are bold because they can be subject to (what is called in RDW) a ‘major change’. A further discussion of this concept follows.

RDW Dimensions	
Organization and Product dimensions can be reclassified (major-changed).	
Company	Competitor
Currency Code	Customer Account
Customer and Customer Demographic	Customer and Product Clustering
Customer Geographic	Customer Order (call center, carrier, carrier service, customer ship to)
Employee	Item-Location Trait Cross-dimension
Item-Supplier-Location Cross-dimension	Market Data
Media	Organization
Plan Season	Product
Product Season	Promotion
Reason	Regionality
Register	ReSA Total Type
Retail Type	Sub-transaction Type
Supplier	Tender Type
Time (time calendar, time of day, time like for like)	Voucher Age Band

Major changes and lower-level dimensions

A major change occurs whenever an entity changes its place in the product hierarchy (group, department, and item can be reclassified) or in the organization hierarchy (area, region, district, and location can be reclassified). This type of reclassification alters the relationship among entities in a hierarchy. Of the dimensions, only product and organization can undergo a major change, and they are known as lower-level dimensions. Another way to think of these is as ‘dimensions with major changeable lower levels’. Because product and organization are aggregating dimensions, a major change results in an altered data aggregation within their hierarchy.

The history of an entity before and after the major change can be tracked and compared. For example, suppose an item is moved from one subclass to another within its product hierarchy of department and class. While there are many good reasons for a retailer to move, or reclassify, an item in this way—perhaps there is a need to track that item in relation to different items in the system—RDW still needs to track sales for that item from its new location in the product hierarchy...both before and after the change. (See the sections, “Pushdowns” and “As-was vs. as-is”, later in this chapter.) Looking at the diagram, “Dimension processing in RDW” (located at the beginning of this chapter), you can see the box labeled “Compare and update processing”. Major change processing occurs at this point. RDW handles major changes by assigning the reclassified item, to use the same example, a new surrogate key. The surrogate key, along with the dimension’s identifier, lets RDW track the dimension and all transactions related to it at any point in time.

Minor changes and top-level dimensions

A minor change means that an attribute of an entity is changed, but its position in the hierarchy remains the same.

The dimensions that can *only* undergo minor changes are known as top-level dimensions and consist of every dimension except organization and product. The levels of the top-level dimensions cannot be reclassified; they are static. Note that product and organization dimensions *can undergo* minor changes, but minor changes are not significant enough to alter their hierarchies.

One example of a minor change is the modification of a description field in a dimension. For example, a description of a subclass is changed from “Humorous Cards” to “Funny Cards”. This type of change does not alter the relationship of subclass to any other level of the hierarchy above or below it. The record is simply updated to reflect the description change; a new surrogate key does not need to be inserted. Minor change dimension processing in RDW is less complex than major change processing.

Actions during processing

During the actual processing of data, there are four kinds of actions that can happen to a dimensional entity in the RDW:

- **Insert:** When an entity is created, it is inserted into the system.
- **Major Change:** When a major change occurs, an entity is effectively closed and re-inserted, so that its history before and after the change can be tracked and compared. (See the passages, “Pushdowns” and “As-was vs. as-is”, later in this chapter.)
- **Minor Change:** When an entity undergoes a minor change, the attribute of the entity is changed, but its position in the hierarchy remains the same.
- **Close:** When an entity is no longer active, it is considered to be closed. Although closing an entity in a transactional system often involves deleting it from the system entirely, in an analytical system like RDW, the entity’s record is retained so that its history can continue to be reported. One exception in RDW is dimensional matrices, where only the current relationship between two source system identifiers (and their surrogate keys) is kept (for example, `item_key` and `itemlst_key` on the `PROD_ITEMLST_MTX_DM` table). Note the two following exceptions to this rule:
 - `Pack_item` relationships on `PROD_PACK_ITEM_MTX_DM`, where deleted `pack_item` relationships, closed items, and reclassified items are all kept on the table.
 - Comparable (`comp`) store relationships on the table, `ORG_LOC_WK_MTX_DM`, where both closed and reclassified locations are maintained.

Maintenance columns in the DM table

- `dm_recd_last_updt_dt`: The last date on which this record was either inserted, updated, or closed.
- `dm_recd_load_dt`: The date on which this record was loaded/created.
- `dm_recd_close_dt`: The last date on which this record could be considered active. Closes occur either because of a record being deleted in the source system, or because a record had a major change applied to it. If the record is an active dimensional record, it will have a default value of '4444-04-04' as a `dm_recd_close_dt`.
- `dm_recd_curr_flag`: Indicates whether a record can be considered active. Valid values are 'Y'es and 'N'o.

Keys and identifiers

Most dimensional entities in the RDW have both keys (typically referred to as 'surrogate keys' or 'pseudokeys') and identifiers (typically abbreviated 'idnt'). The term 'identifier' in the RDW refers to the identifier given to the entity when it was created in the source system. However, in the RDW, this identifier cannot always be used to uniquely identify an entity. An entity may undergo a major change, where it is closed and reloaded in order to mark the change in hierarchy, so that history can be tracked before and after the change. It may also be deleted in the source system, and its identifier reused later. Both of these situations result in multiple records in the RDW tables for the same entity. In order to distinguish between different states of the same entity, or different entities with the same identifier, the RDW must use some other value to uniquely mark it. A surrogate key is a unique value used to identify an entity in the RDW. A new key is attached to an entity whenever it is inserted into a datamart dimension table.

Next_key_val

Each datamart dimension table which needs a surrogate key has a record on the table `MAINT_DIM_KEY_DM`. This record holds the next valid surrogate key for the dimension. The dimension's load program queries this record at the beginning of its run, and, at the end of its run, updates the record with the next valid key for the next run. Note that there are some cases in which the identifiers in the source system are unique, and they will not change over time. If there is no need in RDW to keep track of the changes, RDW does not always create surrogate keys in the applicable dimension tables (`ORG_LOC_TRAIT_DM`, for example).

As-was vs. as-is

One of the primary types of analysis in the RDW is drilling, that is, seeing a particular report at a given level, and then being able to see the same report at a lower level to examine data at a finer level of granularity. This type of analysis makes well-defined hierarchies extremely important in the RDW. Drill paths must be clear, and facts must add up between levels of aggregation. This requirement explains why changes in an entity's place in the hierarchy are considered major.

One of the effects of a major change is that the presence of two surrogate keys makes it possible to compare an entity's performance before and after it undergoes a major change. Fact aggregate tables are also left in a state where the data ties out, because all history was summed up under the entity's old key, while all future data will be summed up under its new key. This is referred to as *as-was reporting*, because history is seen as part of the hierarchy it was in. In order to achieve *as-is reporting*, in which history is shown as if it had occurred under the new hierarchy, fact aggregate tables would either have to be eliminated (resulting in poor report performance) or would have to be rebuilt to account for the hierarchical changes. *RDW only supports as-was reporting*.

Pushdowns

In order to optimize performance, each datamart dimension table holds the keys and identifiers of its parent in the hierarchy, its parent's parent, and so on. Because of this structure, when an entity at a higher level undergoes a major change, all of its descendents (held within the lower levels of the hierarchy) must undergo the major change with it. The same rule applies for closes. Each lower-level dimension program joins with that dimension's immediate parent table to get parent keys for incoming data to compare with the keys in the dimension table to decide if there is a major change. For instance, if a group changes to another division, the group key is changed. The incoming department data joins with the group dimension table to get the group key for that department and group combination. If the department's group key is different than the group key in the department dimension table, a major change is recognized. The pushdown effect is seen after each lower-level dimension program runs individually.

An overview of RDW dimension processing flows

The remainder of this chapter illustrates the flow of dimension data from source tables to RDW datamart tables. The processing described begins with a dimension text file provided from the source system. That file is read by dimension and maintenance RDW libraries, which load the data to the dimensional datamart table. Each dimension processing program will have a record (or entry) on the maintenance table `PROGRAM_CONTROL_DM`, with values populated in the `operation_type` and `program_type` columns. See Chapter 8, "Program reference lists", for more details.

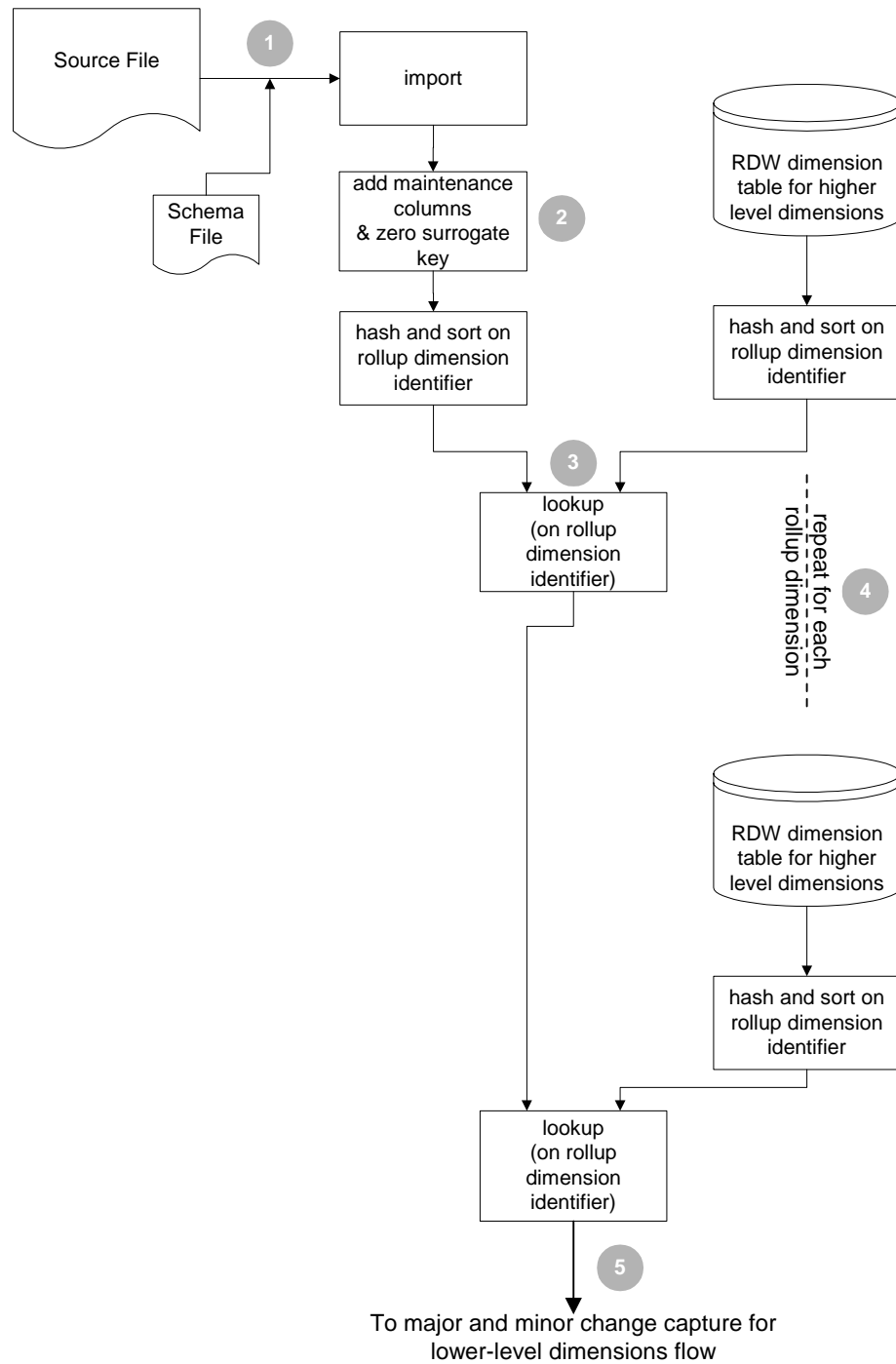
Data preparation for lower-level dimensions

The flow diagram in this section—"Data preparation for lower-level dimensions flow"—is used to produce the source data that is used in step one (1) of the flow that follows, "Major and minor change capture for lower-level dimensions flow." In other words, this flow is not a separate process, but is the predecessor of the flow that immediately follows. Together, the two flows represent a single process that updates a lower-level dimension.

This flow only applies to dimensions that have a parent dimension table above them. That is, they are not the highest-level dimensions in the hierarchy. The source data stream in the change compare dataset must match the RDW table structure, and it must have all the higher-level keys to be able to detect major changes and to have the necessary fields to produce insert records.

Because the dimension data in the source system, RMS for example, is typically normalized, it contains only the idnt of the immediate rollup dimensions, not of any higher-level dimensions. To get all the idnts and keys of all higher-level dimensions (denormalized for performance in RDW), the incoming data is joined with all the immediate rollup dimension tables from RDW. To ensure that the most recent information is being used (and thus to account for major changes in higher level dimensions), the order in which the dimension updating process is applied is to start with the highest-level dimensions and to work down the hierarchy until the base level dimension is processed. Thus, the higher-level RDW tables that are used in the joins will have already been refreshed with the incoming data for those dimensions.

The diagram below shows this flow. Explanations of each numbered item on the diagram follow it.



Data preparation for lower-level dimensions flow

Data preparation for lower-level dimensions flow description

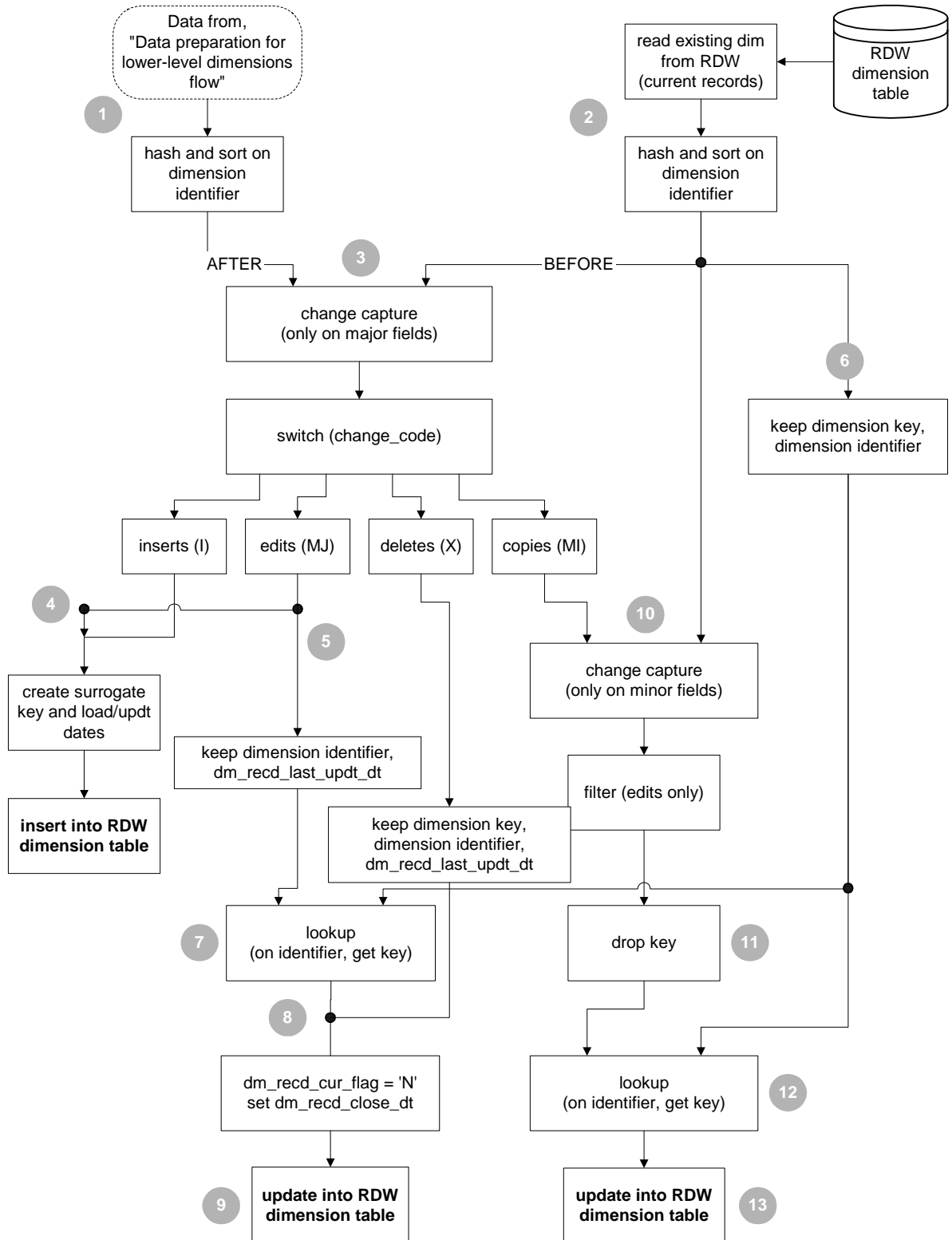
- 1 The current dimension data file is transferred to RDW and loaded into an RETL dataset using an IMPORT operator and predefined schema file.
- 2 The GENERATOR operator adds the following maintenance columns to the dataset (see the section, “Maintenance columns in the DM table”, earlier in this chapter):
 - dm_recd_load_dt
 - dm_recd_last_updt_dt
 - dm_recd_close_dt
 - dm_recd_curr_flag

Note that although the columns are added to the dataset, the processing that occurs within the “Major and minor change capture for lower-level dimensions flow” described later in this chapter determines which columns are actually kept. In addition, a blank or default surrogate key is added, to enable the schema to match the target table in RDW.

- 3 The dataset is joined with one of the immediate rollup tables from RDW. For example, the LOC dimension dataset is joined with the ORG_DISTT_DM table to get the surrogate keys for district and all the dimensions above district, because these keys are redundantly stored in RDW.
- 4 This join with the dimension table above is repeated for every immediate rollup of a dimension. Thus, in the example above, region is not used for one of these joins, because it is not an immediate rollup of location, but is a rollup of district. However, when processing the item dimension, subclass would be joined to incoming item data, because it is an immediate rollup of item.
- 5 The final data is then the input for the next dataflow diagram, “Major and minor change capture for lower-level dimensions data flow”.

Major and minor change capture for lower-level dimensions flow

The diagram in this section describes the general RDW major and minor change capture for lower-level dimensions flow. Explanations of each numbered item on the diagram follow it.



Major and minor change capture for lower-level dimensions flow

Major and minor change capture for lower-level dimensions flow description

Note that the following numbers correspond to the numbers shown in the flow diagram above:

- 1 Data from the source system is already transformed to match the existing dimension table from RDW in all respects except that the surrogate key for the current dimension is not available (set to zero). Although the `dm_recd_load_dt` and other dimension maintenance columns are in the schema, whether each one is kept depends upon the type of processing to occur (for example, insert, edit, delete, and so on).
- 2 The data is read from the RDW table that stores the current dimension's information, filtered to contain only the current records (rows where `dm_recd_curr_flag="Y"`).
- 3 The CHANGECAPTURE operator compares the two incoming datasets and adds a 'change_code' field to the output, which indicates one of the following:
 - inserts (a record exists in the AFTER dataset, but not in the BEFORE)
 - deletes (the record does not exist in the AFTER dataset, but does exist in the BEFORE)
 - edits (a record exists in both datasets but with different values)
 - copies (a record exists in both datasets, and all the minor changeable fields are the same)

This CHANGECAPTURE operator looks only at fields that would cause a 'major' change, and ignores all other fields for the sake of comparison. The delete stream passes the records from the BEFORE dataset, whereas all other streams pass the AFTER dataset unchanged.

- 4 Inserts and edits have a new surrogate key generated for them, have the load and update dates set to the current date, and are inserted into the RDW dimension table. Because the incoming dataset from step one (1) has all the information necessary to fill in the RDW dimension table, these records can be directly inserted with no further joins. Edits result in an insertion and an update because the CHANGECAPTURE operator is detecting major changes, which result in the creation of a new RDW record, and the closing out of the old record.
- 5 Because major changes require closing out the old record, the edit stream also goes to a part of the flow that closes out old records. Closing out a record involves changing the value of the `dm_recd_curr_flag`, `dm_recd_close_dt`, and `dm_recd_last_updt_dt` fields, but no other fields. Because the CHANGECAPTURE operator passes all the fields from the AFTER dataset, all fields are removed except the `idnt` field, which is used to get the old surrogate keys. Thus, the resulting schema of the stream that is used to update RDW only contains fields that are to be updated, and the key. ('`Idnt`' is also there but is guaranteed to be the same because the compare is for the same `idnts`).
- 6 The RDW dataset is stripped to only the dimension `idnts` and keys, which are used as a lookup table to reattach the surrogate keys to datasets downstream. This step is intended to avoid field name conflicts and to stop downstream datasets from getting old values when undesired.
- 7 This lookup gets the old surrogate key for the current dimension for all updated/inserted records.
- 8 Because the remaining steps are to set the `dm_recd_curr_flag` to "N" and update `dm_recd_last_updt_dt` and `dm_recd_close_dt`, deletes and edits, at this point, can be considered together. The same set of fields—only the fields that are necessary to update the records—has been preserved for the delete stream.

- 9 The data is updated in RDW. Because RETL cannot update directly, this step involves a separate process. (See the passage, “Process to update records in RDW”, in Chapter 1.)
- 10 Records deemed as copies imply that no major change has occurred. However, it is possible for a minor change to have occurred. To prevent updating records where no change at all has occurred, this step compares the records again with the current RDW dataset, but this time, the comparison is executed on all minor fields. Only the records considered edits (that is, minor changes) are further processed.
- 11 The surrogate key field is dropped, to allow the actual surrogate key to be re-fetched in the next step (because the major CHANGECAPTURE operator will have lost the surrogate keys).
- 12 The original surrogate key is re-fetched using a join on the idnt field.
- 13 This stream of data is updated into the RDW tables using the standard process for updating records. (See the passage, “Process to update records in RDW” in Chapter 1.) Note that this stream contains many more fields than in step (9), because we effect minor changes. This logic implies that these streams cannot be combined.

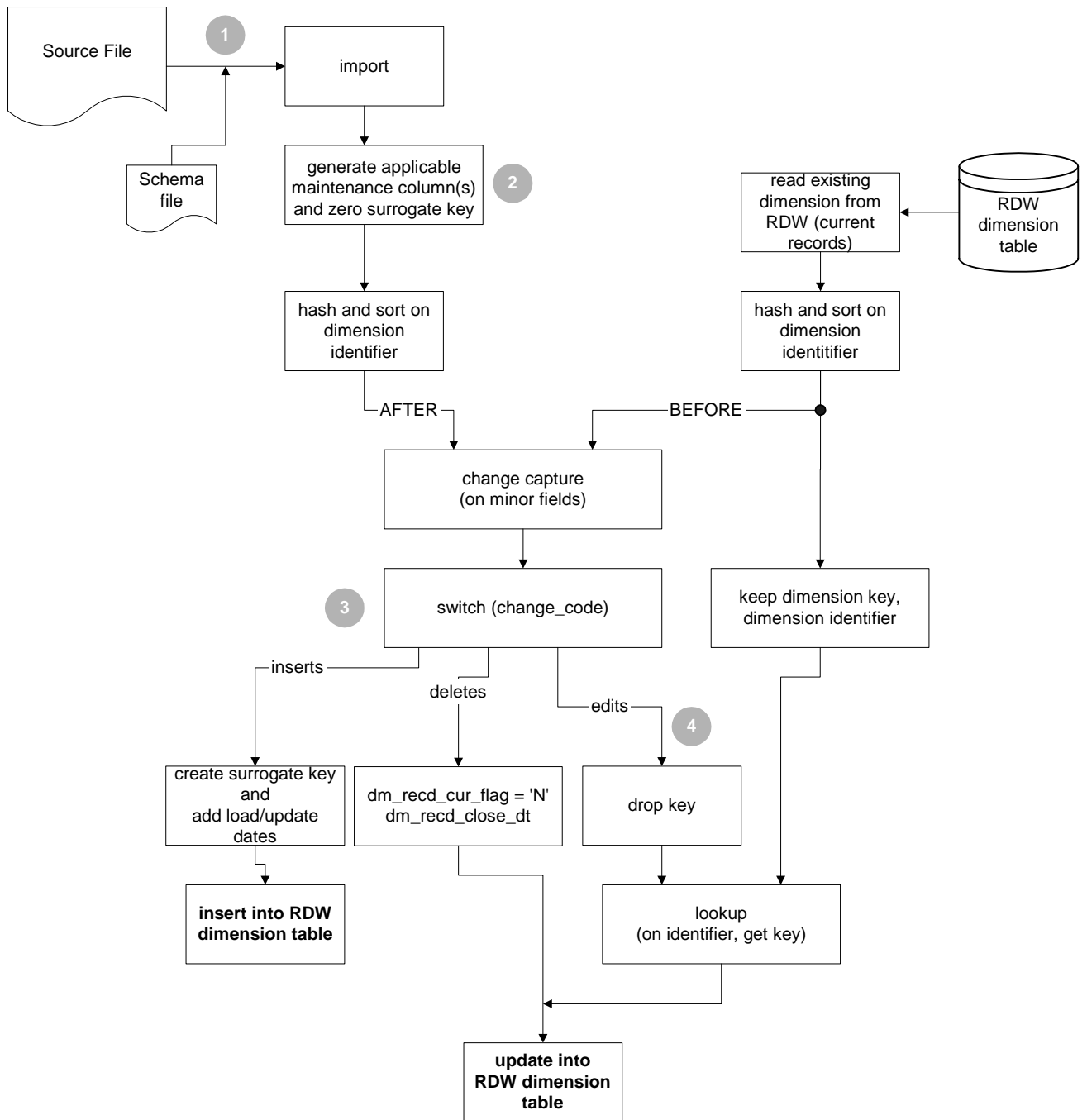
Processing for the regular top-level dimensions

The flow diagram in this section—“Regular top-level dimension processing data flow”—describes the processing of the highest levels in each dimension hierarchy. That is, this section addresses standalone non-hierarchical dimensions, such as currency, along with the highest level of a dimension hierarchy, such as a promotion event. None of the lookups described in the lower-level dimension processing section are required; thus, they do not appear in this flow.



Note: This process flow assumes that the data file from the source system contains the snapshot data of the whole dimension. However, in some exceptional cases (such as the Customer dimension), the source system provides only new or changed data in the dimension data file. In these cases, RDW requires the source system to mark each record in the data file as an insert/update/delete record and has a separate flow to process them accordingly. See the section, “Processing for special top-level dimensions”, later in this chapter.

This regular top-level dimension process has a very simple flow that imports dimension snapshot data from the file, compares it to the target data on minor changeable fields, and uses only the insert and minor change portions of the core change compare flow. The diagram on the next page shows this flow. Explanations of each numbered item on the diagram appear following the diagram.



Regular top-level dimension processing data flow

Regular top-level dimension processing data flow description

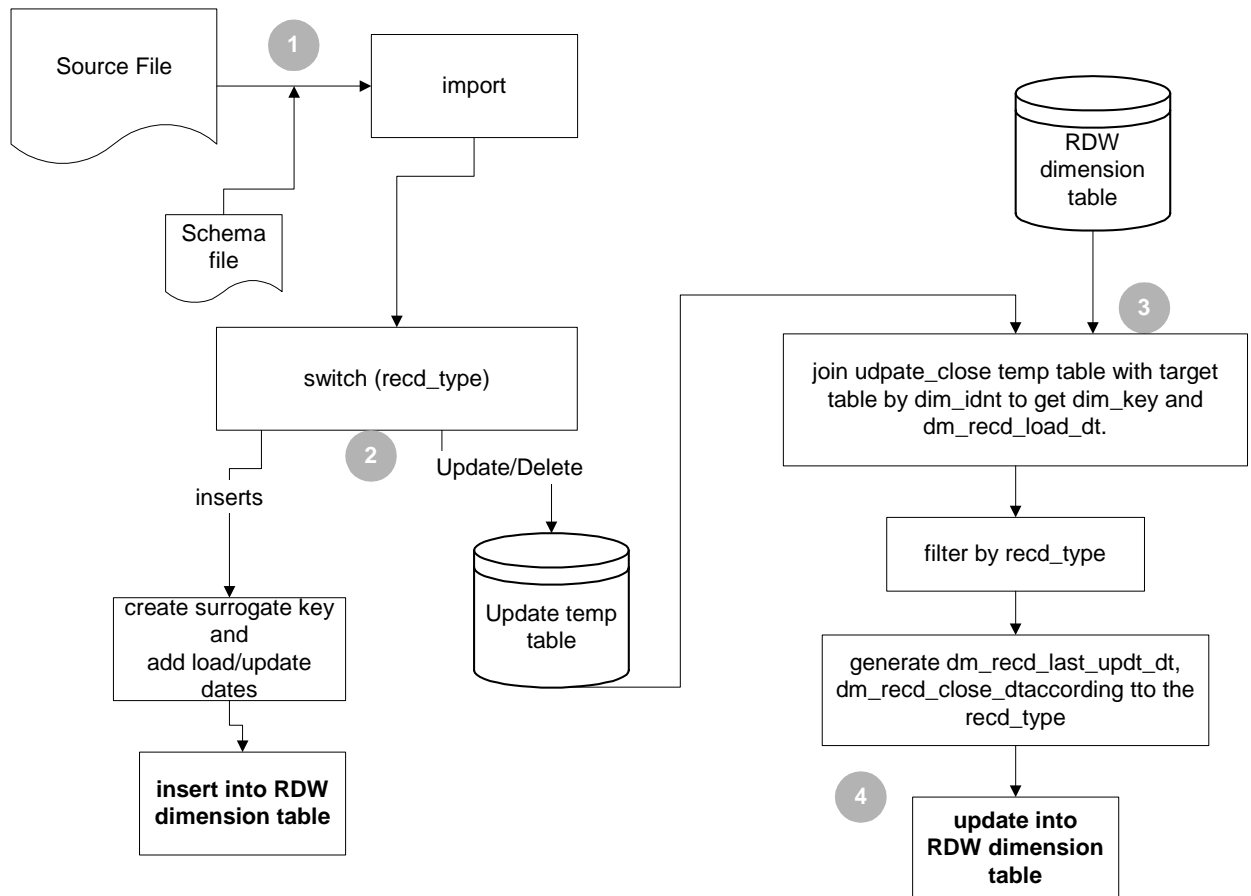
- 1 The current dimension data is transferred to RDW and loaded into an RETL dataset using an IMPORT operator and predefined schema file.
- 2 The GENERATOR operator adds the following maintenance columns to the dataset (see the section, “Maintenance columns in the DM table”, earlier in this chapter):
 - dm_recd_load_dt
 - dm_recd_last_updt_dt
 - dm_recd_close_dt
 - dm_recd_curr_flag

In addition, a blank or default surrogate key is added, to enable the schema to match the target table in RDW.

- 3 The CHANGECAPTURE operator in this case only compares against minor fields, because there are no major fields. Copies are discarded immediately.
- 4 Because changes can only be minor, there is no need to close out records. The lookup to reattach the old surrogate keys is still needed, but these records are then updated directly.

Processing for special top-level dimensions

The flow diagram in this section—“Special top-level dimension processing data flow”—describes the processing of exceptional cases where the source system provides only new or changed data in the dimension data file. Explanations of each numbered item on the diagram appear following the diagram.



Special top-level dimension processing data flow

Special top-level dimension processing data flow description

- 1 The current dimension data file contains only new or changed data. Each record has a `recd_type` of insert/update/delete. This file is transferred to RDW and loaded into an RETL dataset using an IMPORT operator and a predefined schema file.
- 2 The new records are attached to surrogate keys and inserted into the target table. The updated or deleted records are written into a temporary table.
- 3 A query joins the temp table and the target table to select out the needed columns from database, and the data is divided by `recd_type` for updated and closed records.
- 4 The applicable maintenance columns are attached to these different records. The result is used to update the target table.

Datamart table

The datamart (DM) table is the final repository in the data warehouse for dimensional entities. DM tables are visible from the front end. These tables are also used by fact loading programs to perform the following:

- Map identifiers to keys, which are then inserted into fact datamart tables.
- Determine hierarchical relationships for aggregation.

Note that these tables cannot be purged, unless the client wishes to manually roll-off or delete closed dimensional rows (for an item which no longer needs to be queried, for instance). Retek does *not* recommend that clients attempt such dimension data purging, and Retek provides no dimension purging code.

The table and the accompanying descriptions of the maintenance columns shown below illustrate how a record that reflects a change type is reflected in a DM table.

Dimension datamart (DM) table

	dm_recd_last_updt_dt	dm_recd_load_dt	dm_recd_close_dt	dm_recd_curr_flag
Inserted	Current processing date	Current processing date	4444-04-04	Y
Minor Changed	Current processing date	Original load date	4444-04-04	Y
Closed	Current processing date	Original load date	Current processing date	N
Major Changed Closed	Current processing date	Original load date	Current processing date	N
Major Changed Inserted	Current processing date	Current processing date+1	4444-04-04	Y

Chapter 3 – Fact data concepts

This chapter describes the following fact data concepts in RDW:

- An overview of RDW fact processing
- Fact functional areas
- Types of fact tables
- Fact temp table usage
- General fact processing
- Detailed fact load processing
- Fact aggregation processing
- Fact matrix table processing

An overview of RDW fact processing

The following description and “Overview of fact, extraction, load, and aggregation” diagram offer an overview of RDW’s fact process.

For retailers with Retek source applications (RMS, RCOM, ReSA, Merchandise Financial Planning, ReIM, and RPM), extraction programs can be used to extract fact data from these applications. Only changed and/or new facts should (and will be) extracted. Retailers without Retek source applications should provide the data files from their source system in the format recognized by RDW (see “Appendix A –Application Programming Interface (API) flat file specifications”).

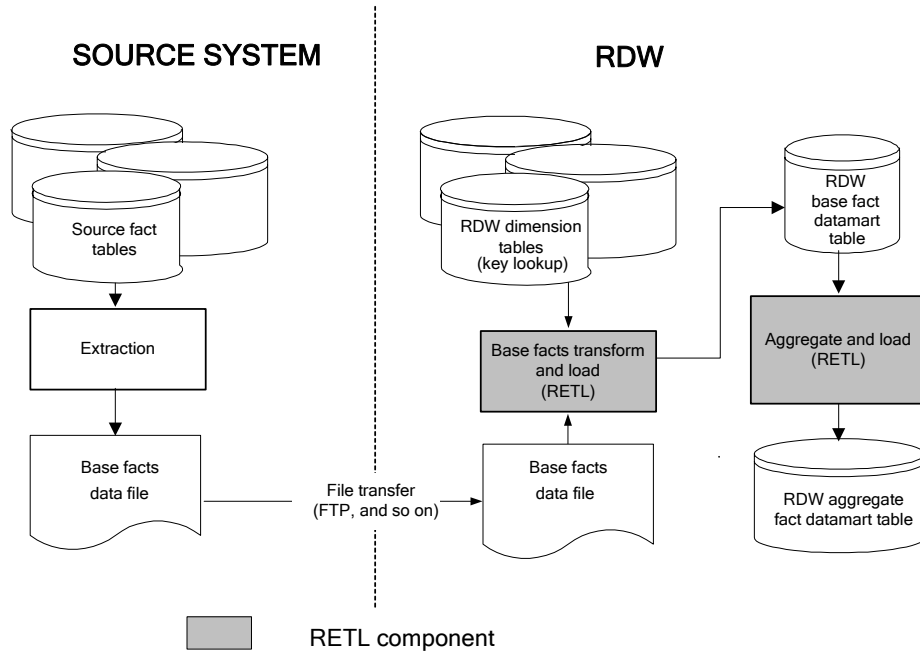
The data file is transferred to the RDW server using a common data transfer process such as FTP. In RDW, the data is pulled from the data file, and identifiers are mapped to the appropriate surrogate keys. (See Chapter 2, “Dimension data concepts”, for a discussion of surrogate keys.)

RDW base fact processing programs, which are Korn shell scripts containing RETL operators and RDW library calls, transform and load data to base fact datamart tables. The next step is the aggregation process, where the data is read from one base fact datamart table, aggregated, and then loaded into one fact aggregate datamart table.

As with dimension processing, see Chapter 8, “Program reference lists”, for program-specific information, such as PROGRAM_CONTROL_DM values, command-line parameters, and so on.



Note: Data extraction programs are available for retailers with Retek source applications (for example, RMS, RCOM, ReSA, Merchandise Financial Planning, ReIM, and RPM). These programs are packaged with the applications.



Overview of fact extraction, load, and aggregation

Fact functional areas

Fact data represent transaction values extracted from a source system such as Retek Merchandising System. RDW's fact functional areas are listed in the following table:

RDW Fact Functional Areas	
Competitor Pricing	Cost
Exchange Rates	Inventory Adjustments
Inventory Position	Inventory Receipts
Inventory Transfers	Loss Prevention
Sales Markdowns	Market Sales Data
Net Cost and Profit on Base Cost	Pack Sales
Planning (Merchandise Financial Planning)	Pricing
Inventory Return to Vendor	Sales Forecasts
Sales Productivity	Sales Transactions
Space Allocation	Stock Ledger
Store Traffic	Supplier Availability
Supplier Compliance	Supplier Contract
Supplier Invoice Cost	Transaction Tender
Unavailable Inventory	Vouchers
Customer Order	

Fact table types: base and aggregate

RDW contains two types of tables: base and aggregate.

- A 'base' fact table holds fact data for a given functional area at the lowest level of granularity. The process of populating a base fact table begins with the extraction of the data from the source system. The extraction results in a text file that is sent to RDW. In RDW, a RETL transformation and load process accepts the fact data file and updates the base table. In order to use RETL to load data files, the RDW fact API defines a schema file to describe the target table columns and data types for each base fact datamart table. RETL references the schema for loading source data files. Data on the base fact table is then aggregated.
- A fact 'aggregate' table holds fact data rolled up from the base table to a higher level of a dimensional hierarchy. RDW uses Korn shell scripts and RETL operators in order to aggregate data.
- Non-compressed fact data can be purged or rolled off whenever a client no longer wishes to query the data. Retek provides no purging routines because purging must be determined by client-specific business requirements. For more information concerning compressed fact tables, see Chapter 4.

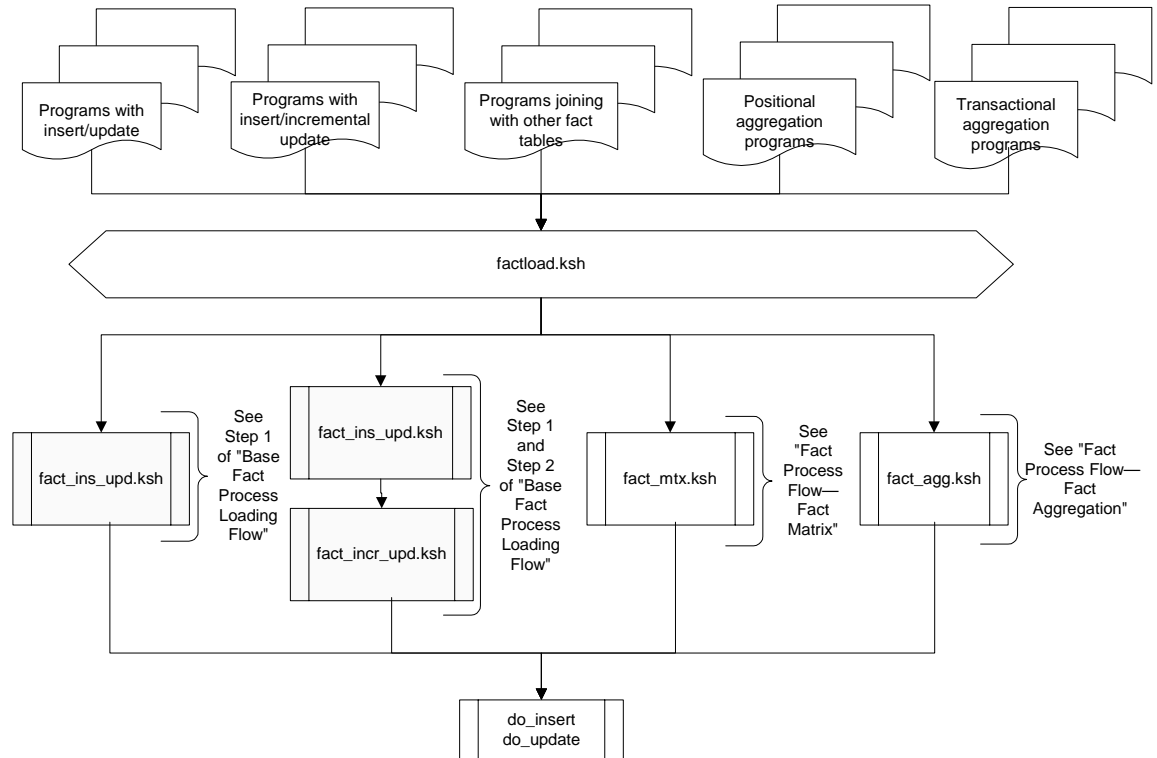
Fact temp table usage

- As noted in Chapter 1, temp tables are created and dropped by the code within the RDW batch programs in cases where records are being inserted/updated.
- For base fact programs, the temp tables that are used for inserting/updating are created by the programs itself. These base fact temp tables contain the current day's data. Thus, for aggregate fact programs, these base fact temp tables can be used to add on to the existing aggregate data when fact data is rolled up to a higher level of a dimension hierarchy.
- The last program to use a temp table within a datamart drops the temp table. There should be no remaining temp tables by the next business day's processing. Again, the RDW code handles the processing of temp tables, from their creation to their being dropped.

General fact processing

The following diagram illustrates the fact process flow in RDW. The flow proceeds from the fact programs (that require the use of sub-libraries) to the factload.ksh. This library interprets the needs of the programs in order to direct them to call the correct sub-library or sub-libraries. Factload.ksh thus plays the role of the 'library traffic cop'. Note that almost every fact program that uses sub-libraries must call factload.ksh so that it can be properly directed. Once the applicable sub-library has processed the program, the system can make the correct changes to RDW's fact tables. The very few standalone programs that do not use factload.ksh are not shown in the diagram.

The flow diagrams described later in this chapter illustrate specifically how and in what context data is processed within each applicable Korn shell sub-library. Thus, adjacent to each sub-library in the diagram below is a callout that refers to the specific process flow diagrams (and any applicable steps therein) that are described later in this chapter.



Fact process flow—general

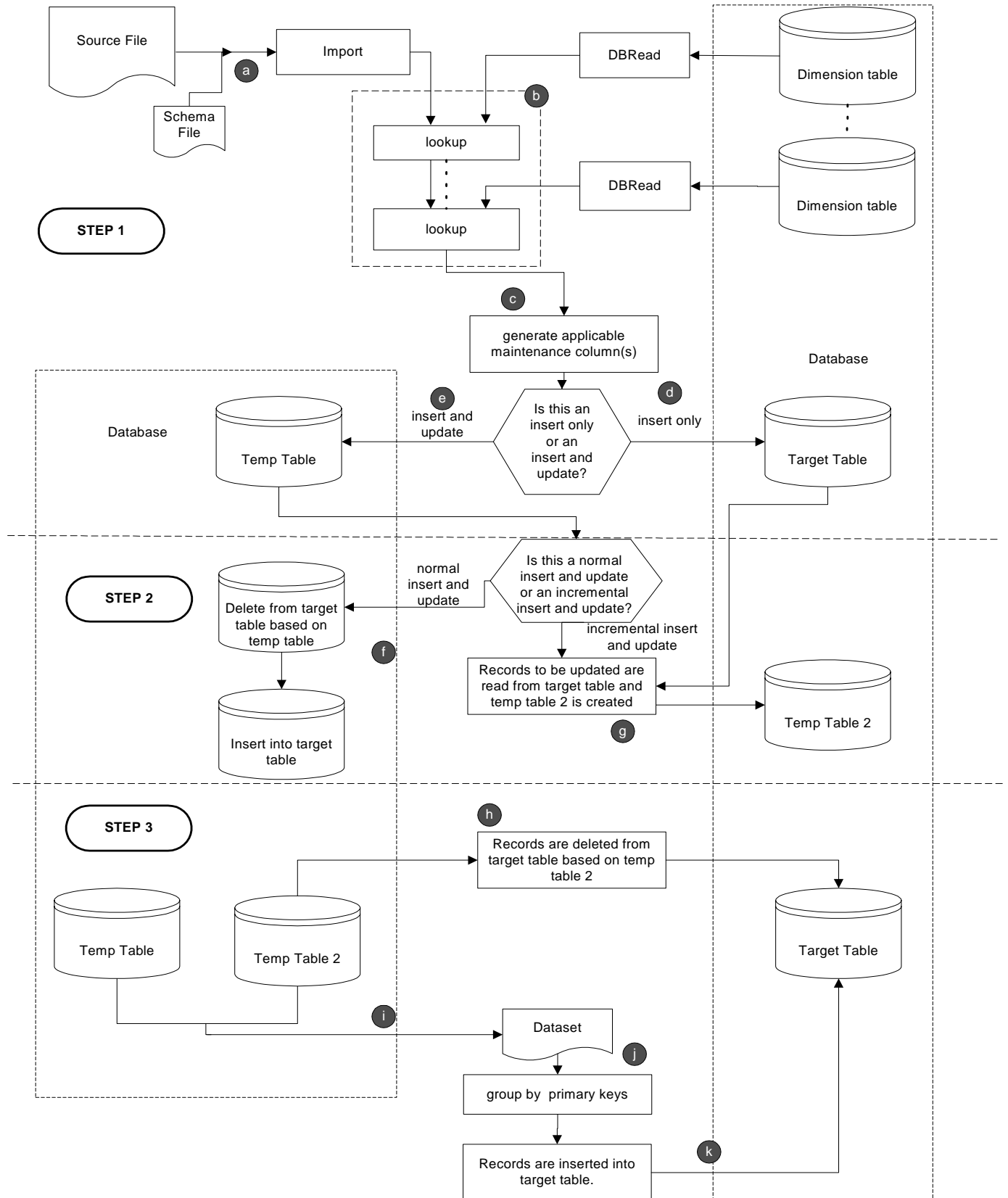
Detailed fact load description

This section describes the process of transforming and loading fact data. In order to use RETL to load data files, RDW uses a schema file to describe the source file fields and data types for each base fact datamart table.

The following diagram describes the general RETL fact process and represents most of the fact loading processes. However, note that the following issues are treated differently for each individual program:

- Some base facts use different dimension tables to do lookups.
- Some base fact loadings only have INSERTs with no UPDATEs (compressed datamarts, for example).
- Some base facts (such as merchandise sales) have incremental updates where the new value is a summation of an old and a new value.

Explanations of the numbered steps and lettered items follow the diagram.



Base fact loading process flow

Base fact loading process flow description

Step 1

- a The fact data file extracted from the source system is loaded into a RETL dataset using the IMPORT operator, based on the schema file that defines all the data fields and data types in the data file that, in turn, are based on the target table.
- b The DBREAD operator is used to read all joining dimension tables into RETL datasets as lookup tables for incoming data in order to get keys based on the identifiers. The number of dimension tables varies for each fact program. Program dimlkup.ksh generates the RETL code (including the DBREAD operator) that selects the data from the dimension tables and joins it with the incoming data. Should the need arise to customize this process, the client can change the variables within dimlkup.ksh.
- c A maintenance column is generated that acts as a date marker (that is, a time stamp). Essentially, this maintenance column records that fact that these rows have been altered on this day.
- d For insert-only fact programs (for example, exchange rate, cost, and so on), the resulting dataset can be appended into the target table directly. For these programs, this step is the end of the process.
- e For fact programs with insert and update records (for example, space allocation, net cost, and so on), the dataset (containing the new records) is written into a temporary table (base temp table).

Step 2

- f For all base fact programs with normal insert and updates, this temporary table is used to determine which of the old update records in the target table should be deleted. The old records are deleted from the target table. The new records are inserted into the target table. For these programs, this step signifies the end of the process.
- g For all base fact programs with incremental insert and updates, the records to be updated are read from the target table and a second temporary table (temporary table 2) is created.

Step 3

- h The temporary table 2 is used to determine which of the old update records in the target table should be deleted, and those records are deleted.
- i The records in the temporary table and in temporary table 2 are combined to be written to the target table (that is, the records are inserted into the target table) using an insert-select query. For base fact programs with incremental inserts and updates, this step signifies the end of the process.

Fact aggregation

After facts are loaded into the base datamart tables, the process of aggregation begins.

Aggregation refers to the process of taking data at a particular level of granularity, for example the item level, and summing it up to a higher level, such as the subclass level, in order to improve query performance. In order for the front end to accurately drill between levels, the names of fact columns must remain the same between the base level and all aggregate levels.

There are two primary types of aggregation in RDW: positional fact aggregation and standard fact aggregation. Positional aggregation updates a value to the current amount at the current time. Standard aggregation sums up all values to the current time. A third aggregation type called a ‘derived datamart’ also exists that supports some complex metrics.

Positional fact aggregation

Some fact tables in the RDW contain information about an entity’s position or status at a given point in time. Such data does not sum up in the same way that transactional data does. See the section “Standard fact aggregation” later in this chapter. For instance, the pricing datamart contains unit retail values for a given item at a given location. Even though new records are written to the table only when a price changes, a user must be able to query for any day and have the system return the correct value. However, storing positions for every item at every location for every day quickly becomes prohibitive from a data storage and load performance standpoint. In order to strike a balance between storage and performance, RDW makes use of a technique called compression to store and report on positional facts. See Chapter 4, “Compression and partitioning”, for more information about how compression works and where RDW uses it.

RDW contains four positional fact aggregation programs. They are listed in the table below.

Positional Fact Aggregation Programs	
Invilwdm	Compressed source and target table
Invblddm	Non-compressed source (cur table) and target table
Invblwdm	Non-compressed source and target table
Sfcblwdm	Non-compressed source and target table

Positional fact aggregation over time

Because data on positional fact tables reports on the state of an entity at a certain point in time, rather than the total activity of an entity, these facts cannot be simply summed over time. For instance, the question: “What was my total unit retail for this week?” is nonsensical. For this reason, aggregations of positional facts along the axis of time take end-of-period snapshots that answer the question: “What was my unit retail at the end of this week?”

With all aggregations along the time axis, aggregation programs run daily. For aggregations of positional facts within a period, this results in a period-to-date position, rather than an end-of-period position. Once the period is complete, the last run of that period results in the desired end-of-period position.

Decompressed aggregates

The compression of positional facts is complex. In order to simplify maintenance and to maximize performance, it is sometimes better to leave base-level facts in their raw compressed state, and to store higher-level aggregates (with less fine levels of granularity) in a decompressed state, in which positions for all entities are written everyday. Building these decompressed aggregates can be a significant task in itself because it involves finding the current positions for every entity at the lower level for the current point in time—even for those entities that may have last had a record some time ago. Fortunately, this task can be simplified by the use of a current position table (such as INV_IL_CUR_DM). A current position table is used, for example, when facts are aggregated from item-location-day to subclass-item-location-day. Less frequently, loads may also make use of a temporary table, which only contains today’s changes, to facilitate bulk processing of the data. For example, when facts are aggregated from item-location-day to item-location-week, the aggregation does not include the entire week’s data, only today’s changes.

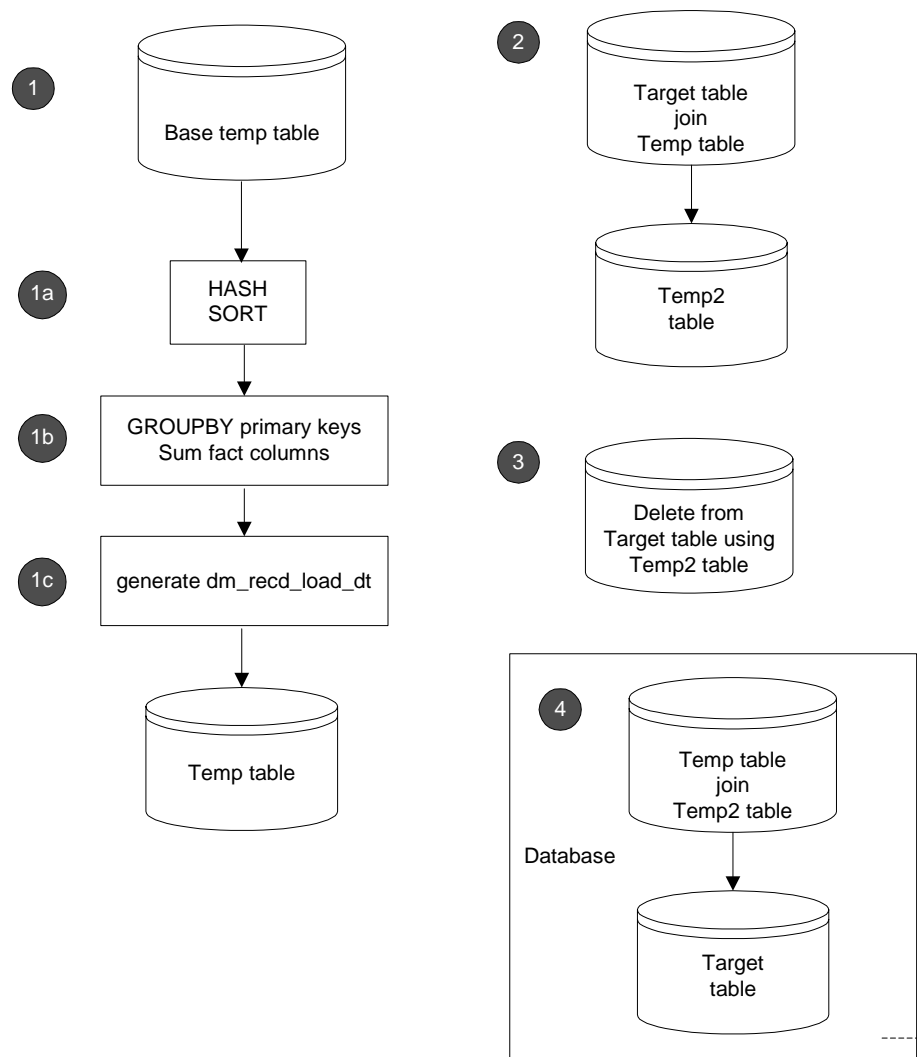
Standard fact aggregation

Most fact tables in RDW contain information about some sort of activity, or transaction, that has occurred. For instance, the merchandise sales tables contain total sales values for a given item at a given location on a given day. This is the simplest type of fact data in RDW. All the data is there and can be summed up along any dimensional axis for reporting purposes.

Fact aggregation from a base fact table flow diagram

Aggregation can be based on a base fact table. In such cases, because a base fact temp table contains the current day's data, its data can be used to add on to the existing aggregate data without summing up everything from base fact table.

The following diagram shows the process of standard fact aggregation process from a base fact table. Explanations of the numbered items follow the diagram.



Fact aggregation from a base fact table data flow

Fact aggregation from a base fact table flow description

The base temp table is the collection of changed and new data that needs to be re-aggregated.

For example:

Base Temp Table (today)				
Day	Location	Item	Week	Amount
4	A	B	1	5
4	A	D	1	70
4	A	F	1	30
4	A	F	1	20

- 1a The base temp table is read into a RETL dataset and hashed and sorted (by the HASH and SORT operators) in the order of its primary key.
- 1b Aggregation takes place on the primary key because of the work of the GROUPBY operator, which facilitates the summation of the fact columns. The aggregation produces:

Aggregation Temp Table				
Day	Location	Item	Week	Amount
4	A	B	1	5
4	A	D	1	70
4	A	F	1	50

- 1c A maintenance column is generated that acts as a date marker (that is, a time stamp). Essentially, this maintenance column records the fact that these rows have been altered on this day.
- 2 The target table is joined with the aggregation temp table to create temp2 table in order to select the rows from the target aggregate table that need to be re-aggregated because data has been changed on and/or inserted to the base temp table today.

For example:

Target Aggregate Table			
Location	Item	Week	Amount
A	B	1	20
A	C	1	30

is joined with:

Aggregation Temp Table				
Day	Location	Item	Week	Amount
4	A	B	1	5
4	A	D	1	70
4	A	F	1	50

to produce temp2 table:

Rows that Need to be Re-aggregated			
Location	Item	Week	Amount
A	B	1	20

- 3 The temporary table 2 is used to determine which of the old update records in the target table should be deleted, and those records are deleted.

For example, the aggregation table now holds:

Target Aggregate Table			
Location	Item	Week	Amount
A	C	1	30

- 4 The records in the temporary table and in temporary table 2 are combined in an insert-select query by grouping by the primary keys of the target table to sum up the required fact fields and insert into the target table.

For example:

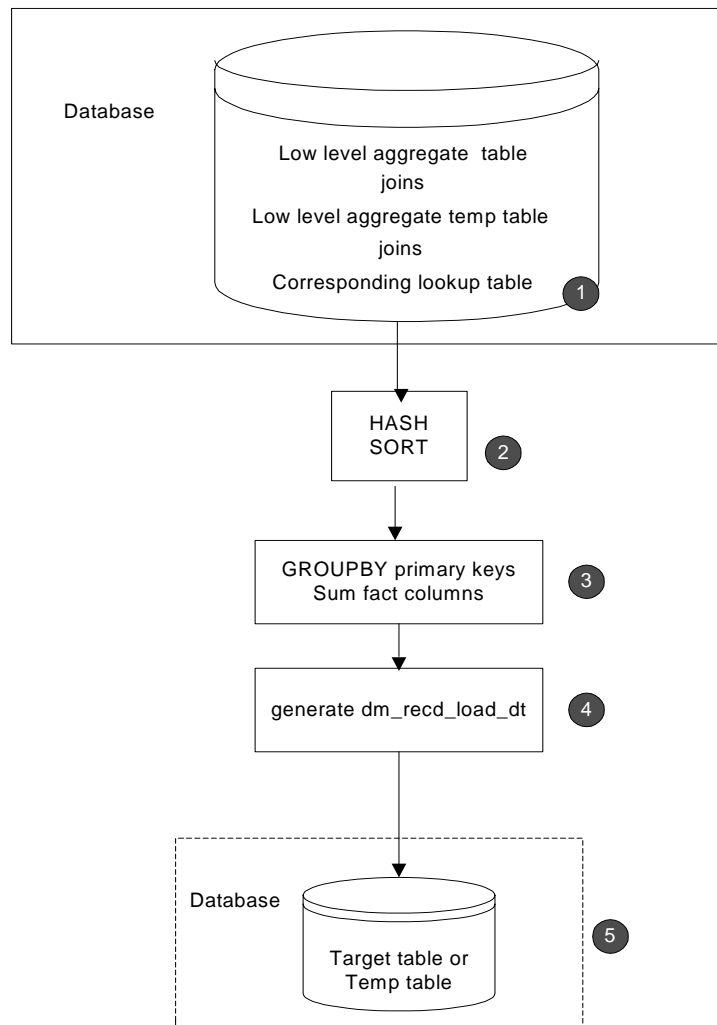
Data inserted into Aggregation Table			
Location	Item	Week	Amount
A	B	1	25
A	D	1	70
A	F	1	50

Data in Aggregation Table			
Location	Item	Week	Amount
A	B	1	25
A	D	1	70
A	F	1	50
A	C	1	30

Fact aggregation from another aggregate fact table diagram

An aggregate fact temp table contains not only current day's data, but also re-aggregated old data. Thus, if the aggregation is based upon another aggregate fact table, the aggregate fact temp table can only be used to find out what data needs to be re-aggregated and to sum up that data.

The following diagram shows the process of standard fact aggregation from another aggregate fact table. Explanations of the numbered items follow the diagram.



Fact aggregation from another aggregate fact table diagram

Fact aggregation from another aggregate fact table description

- 1 Data has been changed on and/or inserted to the lower level aggregate temp table today. Thus, the low level aggregate table is joined with the lower level aggregate temp table and corresponding lookup table in order to select the rows from the lower level aggregate table that need to be re-aggregated to the higher level aggregate table.

For example:

Lower Level Aggregate Table			
Location	Subclass	Day	Amount
A	B	1	20
A	C	1	30
A	C	2	10
A	C	3	25

is joined with:

Lower Level Aggregate Temp Table (Today)				
Day	Location	Subclass	Week	Amount
1	A	C	1	5
4	A	D	1	70
4	A	F	1	30

to produce:

Rows that Need to be Re-aggregated			
Location	Subclass	Week	Amount
A	C	1	5
A	C	2	10
A	C	3	25
A	D	1	70
A	F	1	30

- Each RETL dataset is hashed and sorted (by the HASH and SORT operators) in the order of its primary key so that the GROUPBY operator can be utilized.
- Aggregation takes place on the primary key because of the work of the GROUPBY operator, which facilitates the summation of the fact columns.

For example:

RETL Dataset after Aggregation			
Location	Subclass	Week	Amount
A	C	1	40
A	D	1	70
A	F	1	30

- A maintenance column is generated that acts as a date marker (that is, a time stamp). Essentially, this maintenance column records the fact that these rows have been altered on this day.
- The data is written to either:
 - the target table if the applicable programs contained only inserts.
 - a temp table if the applicable programs contained updates. The target table then undergoes a normal update process. If necessary, see the passage, “Process to update records in RDW”, in Chapter 1.

Derived datamarts

To support some complex metrics, it is sometimes necessary to build an aggregate table with facts that are more than simple sums of those lower levels of granularity. This is similar to standard aggregation in that data moves from one fact datamart table to another. However, because the fact column names are different, there is no straight drill path between the two levels. As a result, these higher-level DM tables are not really aggregates in the purest sense; rather, they are different datamarts derived from a lower level. Here is an example.

The Sales datamart contains profit calculated using an item’s weighted average unit cost. The Net Cost datamart holds various costs for an item, from a given supplier, that are used for a more detailed profit analysis. By combining data from these two functional areas—the Net Profit datamart is built. By deriving a datamart, the user can view profit analysis reports in the front end without the use of overly complex metrics. An additional benefit of deriving a datamart is that database performance improves.

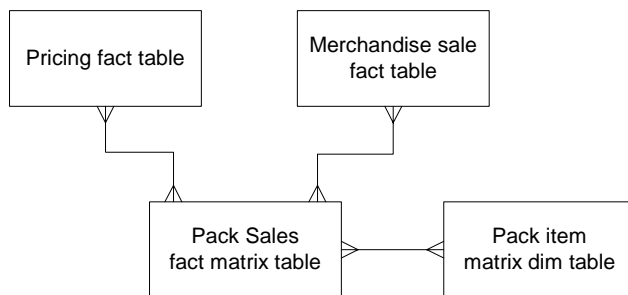
Derived datamarts in RDW include the following:

- Sales Transaction Summary
- Tender Transaction Summary
- Loss Prevention Transaction Summary
- Supplier Compliance Summary
- Net Profit
- Pack Sales / Pack Sales Markdowns
- Voucher Movement

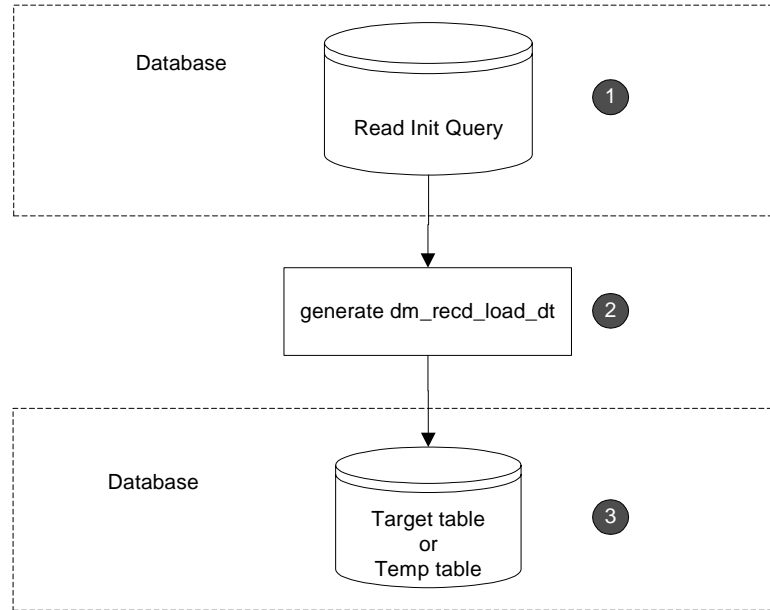
Fact matrix processing

A derived datamart can also be thought of as a ‘fact matrix’. As the diagram and table below illustrate, the matrix table, by having the same dimension key, resolves the relationship among fact and dimension tables that have, in terms of their cardinality, a many-to-many relationship.

For example:



Fact Matrix Table Example				
Item	Location	Day	Cost Amount	Sale Amount
1	1	2	5	10
1	2	1	10	20
2	1	1	5	10



Fact process flow—fact matrix

- 1 When the program calls the library (fact_mtx.ksh), the RETL database read operator (for example, ORAREAD), within a query, retrieves the data. The query can, if necessary, perform aggregation.
- 2 A maintenance column is generated that acts as a date marker (that is, a time stamp). Essentially, this maintenance column records the fact that these rows have been altered on this day.
- 3 The data is written to either:
 - the target fact matrix table, if the applicable programs contained only inserts.
 - a temp table if the applicable programs contained updates. The target fact matrix table then undergoes a normal update process. If necessary, see the passage, “Process to update records in RDW”, in Chapter 1.

Chapter 4 – Compression and partitioning

This chapter describes how RDW implements compression, and offers a discussion of Oracle partitioning.

Overview of compression

Although data warehouses are often very large, the amount of detail generated in some RDW tables is enormous even by usual standards. For example, a retailer with 500,000 items and 500 locations would generate 250,000,000 new rows each day. Storing this amount of uncompressed data is impractical from a disk storage perspective, both in the cost to store the rows and in the cost to perform backups and other database maintenance operations.

One approach that RDW uses to reduce the data volume is compression. This chapter describes:

- What compression does
- The mechanics of compression
- Which tables are currently compressed
- The Oracle features that are related to compression
- Strategies for implementing compressed tables for Oracle clients

What compression does

Compression refers to storing physical data that only reflects changes to the underlying data source, and filling in the gaps between actual data records through the use of database views. This method is engaged primarily for subject areas that are perpetual, such as inventory. For example, when querying sales data, a valid sale record either exists (a sale occurred) or a record does not exist (no sale occurred). However, when querying for on-hand inventory, even if no change occurred to the inventory on the date desired, a valid value is still required. One way to resolve this discrepancy is to store a record for every day for every valid item-location combination as mentioned above. Another method, compression, allows for the storage of only changes to the inventory position. The query is resolved by looking backward through time from the desired date (if no change record exists on that date) until an actual change record is found. This method returns the correct current data with the minimum requirements necessary for processing and storing data.

The mechanics of compression

The purpose of decompression views is to give the application the illusion that there is a record for each possible combination (for example, an item-location-day record for each permutation) when in fact there is not. Thus, the fact of whether a table is compressed or not should not be visible to the application that queries data from that table.

A compressed table is made up of two distinct parts: a 'seed' that consists of all existing combinations at a point in time (typically the first day or week of the table or partition) and the changed data since that time.

When resolving a query for a particular record, the decompression view provides the latest record for the requested item and location with the maximum day that is less than or equal to the requested day. A decompression view needs to encompass both the seed and all of the changed data since that seed.

To illustrate how the decompression views actually work, assume the following: I am interested in the inventory position of item 10 at location 10 on 1/23/02. The seed was done on 1/1/02. Changes were posted on 1/4/02, 1/15/02, and 1/30/02. The row that is presented to the application by the decompression view is the row on 1/15/02, because it is the greatest date that is less than or equal to the requested date. As a second example, assume that the inventory position of item 10, location 10, day 1/3/02 was desired. Because there was no change record less than or equal to the desired date, the seed record from 1/1/02 would be presented to the application.

Compression's performance is excellent when the user is querying for a single day (as in the example above). When querying over a group of days, however (for example, all of the inventory positions at a given location on a given day), the performance can be unacceptable. Even though the user is requesting a group of information back and in most cases the database can process groups of information efficiently, each individual row must be evaluated individually by the decompression view and cannot be processed as a group. To counteract the slow performance of these summary operations, Oracle clients may take advantage of compressed table partition seeding (see the passage, "Partitioning for the Oracle client only", later in this chapter).

This partition seeding utilizes CUR tables (also known as 'current' tables). An example is the INV_IL_CUR_DM table, which holds the current decompressed position for every item and location on the INV_ITEM_LD_DM table. This position can be used (by Oracle clients only) as a partition seed. This position is also utilized by base RDW code during major change fact seeding (see the passage, "factopendm.ksh", later in this chapter).

Compressed tables and CUR tables

The table below illustrates the compressed tables within RDW, along with their corresponding CUR tables.

Compressed Tables	CUR Tables
CMPTR_PRICING_ITEM_LD_DM	CMPTR_PRICING_IL_CUR_DM
COST_ITEM_SUPP_LD_DM	COST_ISL_CUR_DM
EXCHNG_RATE_CRNCY_DAY_DM	(No CUR table)
INV_ITEM_LD_DM	INV_IL_CUR_DM
INV_ITEM_LW_DM	(No CUR table)
INV_UNAVL_ITEM_LD_DM	INV_UNAVL_IL_CUR_DM
NET_COST_SUPP_ITEM_LD_DM	NET_COST_SIL_CUR_DM
ORG_LOC_WK_MTX_DM	(No CUR table)
PRICING_ITEM_LD_DM	PRICING_IL_CUR_DM
SPACE_ALLOC_DEPT_LD_DM	SPACE_ALLOC_DL_CUR_DM
SPACE_ALLOC_ITEM_LD_DM	SPACE_ALLOC_IL_CUR_DM
SUPP_AVAIL_ITEM_DAY_DM	SUPP_AVAIL_ITEM_CUR_DM
SUPP_CNTRCT_ITEM_DAY_DM	SUPP_CNTRCT_I_CUR_DM

Coping with major changes

Factclosedm.ksh

On a compressed fact table, a record is only posted to the table when there is a change in one of the fact attributes. If there is no activity, no record is posted. Decompression views then fill in the gaps between physically posted records so the front end can see a fact record for each item-location-day combination. However, when an item or location or department is closed or major-changed, any fact record with those dimensions becomes inactive. The decompression views need to be informed to stop filling in the gap after the last record was posted. To accomplish this instruction, factclosedm first queries the PROD_ITEM_RECLASS_DM, ORG_LOC_RECLASS_DM, and PROD_DEPT_RECLASS_DM tables (see the section, “factopendm.ksh”, later in this chapter) to determine what compressed item-location facts need to be closed today. Factclosedm then inserts a ‘stop record’ that has a DM_RECDD_STATUS_CDE = ‘X’. The decompression view fills in records up to the day that a status code of ‘X’ is posted. The close record is inserted for tomorrow’s DAY_IDNT, indicating that the fact record is no longer valid beginning tomorrow, when the newly seeded record (from factopendm.ksh) becomes active. In the case of the one compressed week table, INV_ITEM_LW_DM, factclosedm inserts close records for next week’s WK_IDNT.

Factopendm.ksh

RDW Data Compression tables require seeding when a major change in the product and/or organization dimension causes new surrogate keys to be created for items or locations. Seeding the compressed tables is required because the new key represents a new hierarchy relationship. If the new key is not represented on the compressed table, the compression view does not pick up any data from the day the old dimensions were closed to the day a record with the new dimensions is posted to the compressed fact tables. This missed data causes inaccuracy in query results and incorrect data aggregation.

There are two steps to this seeding process. First, the programs prditmdm.ksh, prddepdm.ksh, and orglocdm.ksh run as part of the dimension loading process to populate temporary tables PROD_ITEM_RECLASS_DM, PROD_DEPT_RECLASS_DM, and ORG_LOC_RECLASS_DM. Next, the program factopendm.ksh looks at the tables for reclassified items, departments, or locations. It seeds all of the compressed tables with records that contain the reclassified ITEM_KEYS, DEPT_KEYS and/or LOC_KEYS.

Partitioning for the Oracle client only

Overview of partitioning strategies

Currently, RDW base code ships with no tables partitioned. Because RDW is database independent and partitioning only applies to Oracle clients, this section describes optional partitioning strategies for compressed datamarts that clients using Oracle may wish to pursue.

As described earlier, ‘decompression views’ provide a virtual view of a fully populated table even though the underlying table is actually only sparsely populated. For large compressed tables, especially the INV tables, splitting them into table partitions can provide the following benefits:

- Partitions are smaller and therefore easier to manage
- Management operations on multiple partitions can occur in parallel
- Partition maintenance operations (such as index rebuilds) are faster than full table operations
- Partition availability is higher than table availability (for example, if I am recovering a particular partition, users may access all other partitions of the table at the same time)
- The optimizer can prune queries to access data in only the partition of interest, not the entire table (for example, if I am interested only in February’s data, I do not need to look at any of the table’s data outside of the February partition)
- Partitions are separate database objects, and can be managed accordingly (for example, if December sales are frequently accessed throughout the year whereas other months are not, the December sales partition could be located in a special tablespace that allows for faster disk access)
- In some situations, Oracle can create parallel operations on partitions that it cannot on tables; an example is joining between two different tables if they are partitioned on the same key (this feature is called a ‘parallel partition-wise join’)

The general guideline is that table partitions should be used on very large tables. Tables larger than 20 GB would be potential candidates for partitioning. For optimal performance on inventory tables, partitioning is highly recommended.

Indexes, as well as tables, can be partitioned. ‘Index partitions’ can be global (one index over the table, regardless of whether the table is partitioned or not) or local (there is a 1-to-1 correspondence between index partitions and table partitions). In general, when tables are partitioned, local indexes should be preferred to global indexes for the following reasons:

- Maintenance operations involve only one index partition instead of the entire index (for example, if the oldest table partition is aged out, a local index partition can be dropped along with its corresponding table partition, whereas an entire global index would need to be rebuilt after it becomes unusable when a table partition is dropped)
- The optimizer can generate better query access plans that use only an individual partition
- When multiple index partitions are accessed, the optimizer may choose to use multiple parallel processes rather than just one

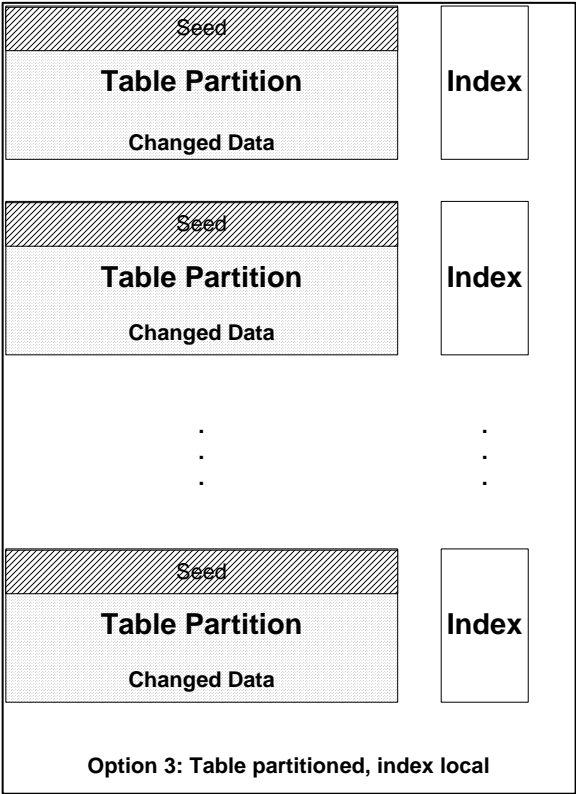
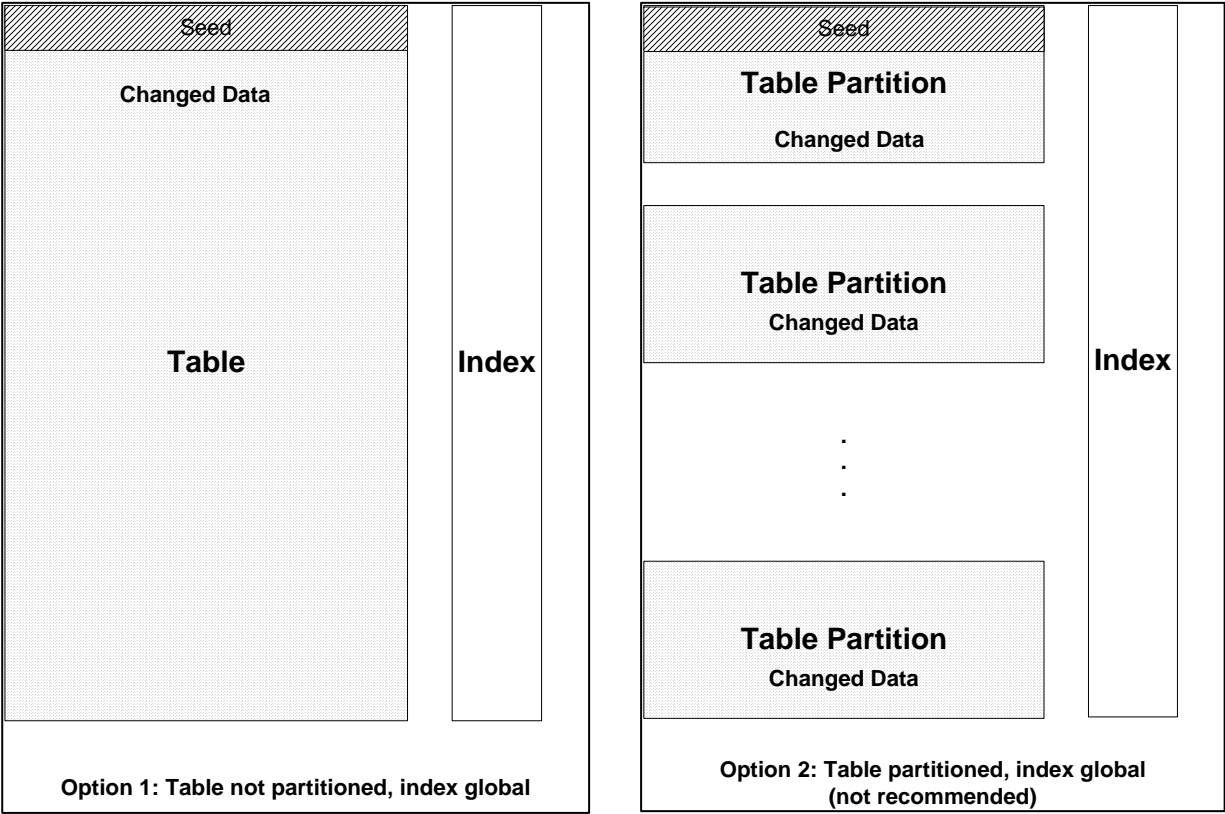
Implementing RDW partitioning

For those clients who choose to partition a compressed table, the diagrams on the following page illustrate some of the possibilities for table and index layout.

In general, option 3 is the preferred solution for large compressed tables (typically, the INV_ITEM_LD_DM and INV_ITEM_LW_DM tables). It uses table partitions and local indexes, thus minimizing the impact of index maintenance and the deletion of old table partitions.

Option 1 is viable for smaller compressed tables (typically, all other compressed tables besides INV). The disadvantage is that there is no way to delete historical data (that is, the table continues to grow).

Option 2 is not recommended. It combines the disadvantage of not allowing for historical data to be purged with the disadvantage of having a global index on a partitioned table.



Partitioning strategy and requirements for MicroStrategy 7

Partitioning can also be managed at the application level. This strategy can be accomplished in addition to or independent of the partitioning that can be done at the database level.

MicroStrategy partitioning allows a large fact table to be split into several smaller tables. For example, instead of creating just one decompression view for a compressed table, you can create several smaller decompression views for the compressed table. The appropriate partitioning strategy can both significantly improve query response time and decrease the time required to load tables. These advantages must be weighed against the increased database maintenance that is required in a partitioned environment.

Partitioning can be effective for large tables that allow splitting along the time dimension. For example, a sales fact table might be partitioned by year in an environment in which the majority of queries retrieve data for the current year. In such a case, performance would increase because the majority of queries would be run against a smaller table. Although time is frequently used to partition tables, MicroStrategy 7 allows partitioning along any dimension.

MicroStrategy 7 supports the following two types of partitioning:

- Warehouse partition mapping manages partitioning using mapping tables. These must be created and managed by the DBA.
- Metadata partition mapping manages partitioning from the MicroStrategy 7 application. This method eliminates much of the maintenance at the database level. This method must be carried out by the MicroStrategy administrator, using the the Metadata Partition Mapping Editor.

Warehouse partitioning

Warehouse partitioning requires a partition mapping table and partition base tables. When the partition mapping table is added to a MicroStrategy project, the partition base tables are also automatically added. You do not add the original fact table to the MicroStrategy project.

Partition base tables (PBT)

Partition base tables are smaller physical tables (or views) that the database administrator creates in the data warehouse. Each table (or view) contains a subset of the data in the original fact table. The database administrator is responsible for keeping the partitions consistent and up to date.

Partition mapping table (PMT)

The DBA must create a table that maps the new tables according to the attribute used to create the partition. The PMT table must have the following structure:

ATTRIBUTE_ID	PBTNAME

The ATTRIBUTE_ID column contains the values of the attributes on which the table is partitioned. The PBTNAME (Partition Base Table Name) column contains the names of the partitions.

The PMT table for a partition by year would resemble the following:

YR_ID	PBTNAME
1997	Y1997_Sales
1998	Y1998_Sales
1999	Y1999_Sales
2000	Y2000_Sales
2001	Y2001_Sales

Multiple attributes may be used to create partitions. For example, you might partition by year and region. In this case, the PMT would contain the year and region identifiers and the corresponding PBT names.

Metadata partitioning

In this method, partition base tables are mapped in the project metadata, eliminating the need for the PMT. These objects, sometimes referred to as ‘data slices’ are filters that define the contents of the partition base tables. These objects are created with the Metadata Partition Mapping Editor within MicroStrategy Desktop.

An example of setup and maintenance for partitioning RDW's compressed inventory tables using warehouse partitioning

- 1 Make the following determinations, among others:
 - Your partitioning strategy
 - The time period your partitions will use
 - The 'values less than' boundaries according to your calendar
 - How many partitions are to be used
 - The partition naming standard
- 2 On the database, create the partitions and indexes for the tables you want to partition.
- 3 Verify you have populated the Time Calendar Dimension. See the RDW Database Installation Guide for details.
- 4 Create one decompression view for each database partition you created for the Inventory Position tables (that is, INV_ITEM_LD_DM and INV_ITEM_LW_DM). You should also perform this action for other compressed tables you partition.
- 5 Create the partition mapping tables (PMT) according to the structure specified earlier.
- 6 Re-run the standard grant and synonym scripts. See the RDW Database Installation Guide for details.
- 7 Populate the Partition Mapping Tables PMT_INV_ITEM_LD_DM and PMT_INV_ITEM_LW_DM as well as any other compressed PMT tables that you have created.

Perform step numbers 4, 5, 6, and 7 whenever any of the following events occur:

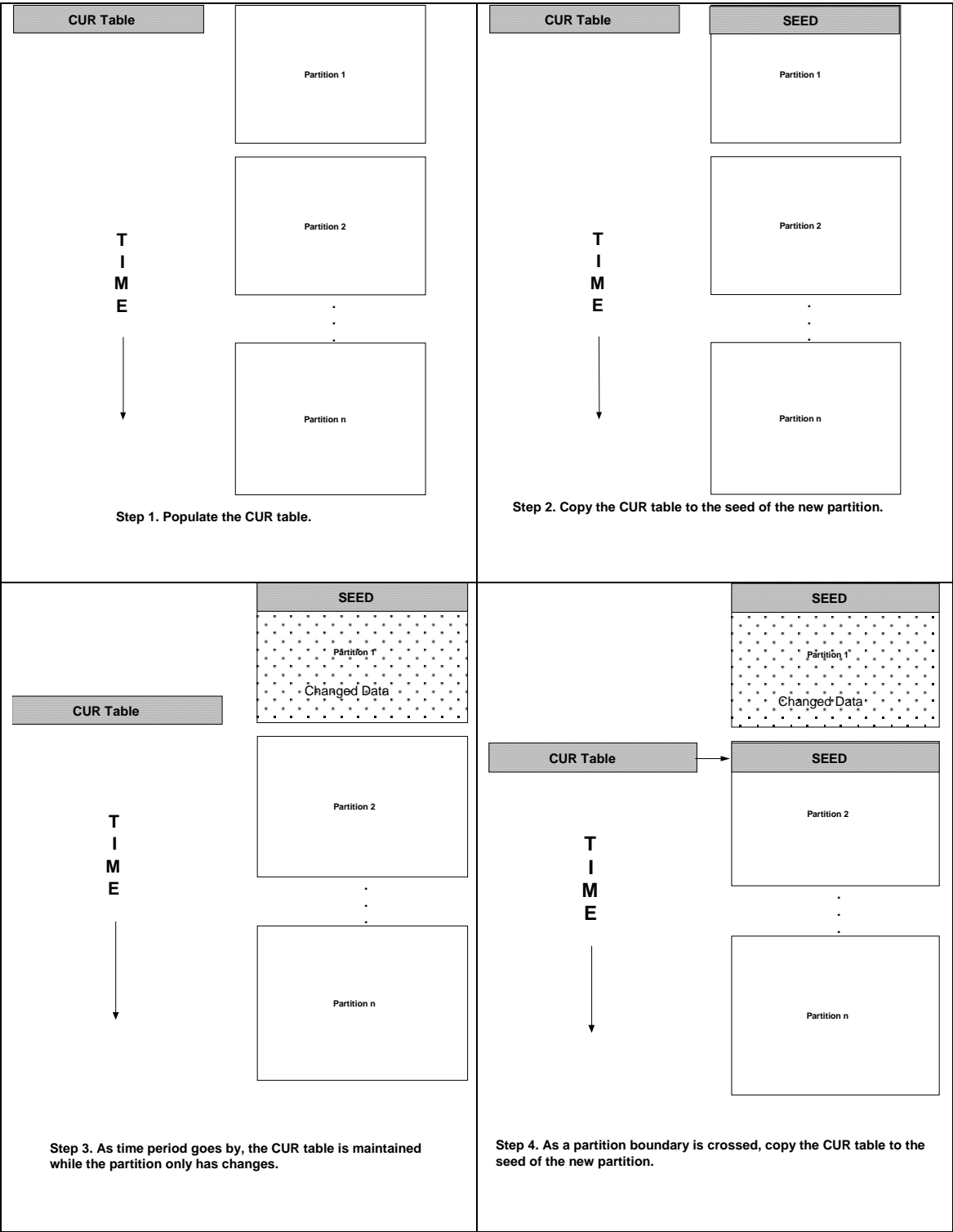
- Records are added to or deleted from the Time Calendar tables TIME_DAY_DM or TIME_WK_DM
- Partitions are added to the Inventory Position tables INV_ITEM_LD_DM or INV_ITEM_LW_DM
- Partitions are removed from the Inventory Position tables INV_ITEM_LD_DM or INV_ITEM_LW_DM

Implementing partitioning for compressed inventory tables

Once the tables (including partitions) and indexes have been created, the data must be loaded. For tables that have a corresponding CUR table (such as INV_ITEM_LD_DM and INV_IL_CUR_DM), the recommended steps are described in the following text and shown in the diagram on the following page:

- 1 Populate the INV_IL_CUR_DM table with data. (This population is accomplished the first time that invliddm.ksh runs and an inv position record is processed.)
- 2 Copy this INV_IL_CUR_DM table as the seed to the first partition. (This step is performed automatically by the orapartseed.ksh program. See the chapter, “Compression and partitioning”, for more information about seeding.)
- 3 At this point, only the changed records are added to the INV_ITEM_LD_DM table, whereas the INV_IL_CUR_DM table is a full, uncompressed version that holds the current inventory position as of the last time period.
- 4 When a partition boundary is crossed, the INV_IL_CUR_DM table is copied as the seed to the new partition, via the orapartseed.ksh program.

If you have questions about how to implement partitioning with compression or would like assistance implementing partitioning, contact Retek Customer Support or Retek Services.



Implementing partitioning for compressed inventory tables

How Oracle implements partitions

Oracle 8.0 introduced range partitions. These partitions are split by a range of values on the partition key. Examples include partitions by month, partitions by department number, and partitions by item range. Oracle 8.1 expanded the partitioning options to include hash partitions (spreading the rows across a fixed number of partitions by applying a hash function to the partition key), as well as composite partitioning (a combination of range partitioning and hash partitioning). Retek recommends that clients partition their tables using range partitioning. Retek also recommends that the partition key be the date field in the primary key to allow partitions to be aged out when no longer needed.

Retek's general guideline is that partitioning should be considered for a table that will be 20 GB or larger. Because there is an administrative trade-off between having more partitions to manage and obtaining the benefits of partitioning, partitioning tables smaller than 20 GB should only be considered when specific circumstances dictate. In addition, individual partitions should be kept to 10 GB or less.

The actual physical layout of partitions varies from site to site. A general approach is to put each partition into its own tablespace. This has several advantages:

- Maintenance operations, as well as tablespace recovery, can occur on a partition while other partitions are unaffected
- If manual performance tuning of the data files is being done, tablespaces and their files can be moved around to achieve optimal performance
- If partitions are no longer being updated, their tablespaces can be changed to READ ONLY, which greatly reduces backup requirements

Oracle partitions are ordered from low values to high values. The partition key value for a partition is a non-inclusive upper bound (high value) for that partition. For example, if the SLS_ITEM_LD_DM table is partitioned by month, the high value for the January, 2000, partition is 01-Feb-2000. A low value can always be inserted into the lowest partition. However, a high value may not be able to be inserted depending on the high value of the highest partition. For instance, if the highest partition has a high value of 01-Feb-2000, and the insertion of a record is attempted with a date of 01-Feb-2000, the row cannot be inserted into the table (remember that the high value of 01-Feb-2000, is a *non-inclusive* upper bound). For this reason, Oracle allows a special high value partition with a key of MAXVALUE. Retek recommends that all partitioned tables include a dummy partition with a MAXVALUE high value.

There are special considerations for the partitioning of RDW compressed tables. The following is a brief description of the different partition maintenance commands. Please see the current Oracle documentation set for more details:

- **ADD PARTITION:** adds a new partition to the high end of a partitioned table; because Retek recommends having a MAXVALUE partition, and this is the highest partition, the ADD PARTITION functionality can be achieved by performing a SPLIT of the MAXVALUE partition instead
- **DROP PARTITION:** drops the partition; this is the typical method to delete the oldest partitions (those with the lowest values) as they age to maintain a rolling window of data
- **EXCHANGE PARTITION:** converts a non-partitioned table into a partitioned table or converts a partitioned table into a non-partitioned table
- **MERGE PARTITION:** merges two adjacent partitions into one
- **MOVE PARTITION:** moves a partition to another segment; this is used to defragment a partition or to change its storage characteristics
- **SPLIT PARTITION:** splits an existing partition by adding a new partition at its low end
- **TRUNCATE PARTITION:** removes all rows from the partition

Oracle automatically maintains local index partitions in a 1-to-1 correspondence with their underlying table partitions. Any table partition operations, such as ADD PARTITION, also affect its appropriate index partitions.

Summary

Partitions are useful for breaking up large tables into smaller, more manageable pieces. Retek offers the following recommendations for partitions:

- Consider partitioning tables that will be greater than 20 GB in size or those for which a special need exists that partitioning could benefit
- Keep individual partitions under 10 GB in size
- Use the date as the partition key for range partitioning
- When tables are partitioned, make their indexes local
- Consider putting each partition in its own tablespace
- After updates on a partition cease, consider changing its tablespace to READ ONLY to reduce backup requirements

If partitioning compressed tables, be sure to address their special requirements.

Chapter 5 – RDW program overview

This chapter summarizes RDW's RETL programs. More information about the RETL tool is available in the latest RETL Programmer's Guide.

This chapter references the directory structure set up during RDW product installation. Descriptions of these directories are available in the RDW Database Installation Guide.

Program features

RDW's RETL programs include the following features:

- Program return code
- Restart and recovery
- Message logging
- Program error file
- Schema files
- Resource files
- Command line parameters
- Partitioning

Program return code

RDW's RETL programs use one return code to indicate successful completion. If the program successfully runs, a zero (0) is returned. If the program fails, a non-zero is returned.

Restart and recovery

RETL processes all records as a set, as opposed to one record at a time. The restart and recovery process within RDW serves the following two purposes:

- 1 It prevents the loss of data due to program or database failure.
- 2 It increases performance when restarting after a program or database failure by limiting the amount of reprocessing that needs to occur.

RDW restart and recovery

Datamart load programs that use a single RETL flow do not require the use of restart and recovery. If the load process fails for any reason, the program can be re-run from the beginning.

More complicated programs that require the use of multiple RETL flows have the potential risk of losing data during a failure. Each individual step is committed once it is successful. Thus, if a later step fails, a rollback is impossible. These programs utilize a bookmark method for restart and recovery. This method allows the program to be restarted at the point of last success and run to completion. The bookmark restart/recovery method incorporates the use of a bookmark flag to indicate which step of the process should be run next. The bookmark flag is written to and read from a bookmark file for each step in the process.



Note: If the fix for the problem causing the failure requires changing data in the source table or file, then the bookmark file must be removed and the process must be re-run from the beginning in order to extract the changed data.

Program control tables

The PROGRAM_CONTROL_DM table is used by RDW to determine how to process records for a program. See Chapter 8, “Program reference lists”, for program-specific details about the contents of the PROGRAM_CONTROL_DM table.

The PROGRAM_CONTROL_DM table holds record-keeping information about current program processes. See the RDW Data Model for table and column descriptions.

Bookmark file

The name and directory of the restart and recovery bookmark file is set in the configuration file, `rdw_config.env`. The directory defaults to `${MMHOME}/rfx/bookmark`. The naming convention for the bookmark file itself defaults to the following:

- The program name
- The first filename, if one is specified on the command line
- ‘bkm’
- The business virtual date for which the program was run

For example, the bookmark flag for the `prditmdm` program would be written to the following file for the batch run of January 5, 2001:

```
${MMHOME}/rfx/bookmark/prditmdm.bkm.20010105
```

Message logging

Message logs are written daily in a format described in this section.

Daily log file

Every RETL program writes a message to the daily log file when it starts and when it finishes. The name and directory of the daily log file is set in the configuration file, `rdw_config.env`. The directory defaults to `${MMHOME}/log`. All log files are encoded UTF-8.

The naming convention of the daily log file defaults to the following:

- The business virtual date for which the programs are run
- ‘.log’

For example, the location and the name of the log file for the business virtual date of January 5, 2001 would be the following:

```
${MMHOME}/log/20010105.log
```

Format

As the following examples illustrate, every message written to a log file has the name of the program, a timestamp, and either an informational or error message:

```
cusdemogdm 13:20:01: Program Starting...
cusdemogdm 13:20:05: Build update and insert data.
cusdemogdm 13:20:13: Analyze table rdwlldev.cust_demog_dm_upd
cusdemogdm 13:20:14: Insert/Update target table.
cusdemogdm 13:20:23: Analyze table rdwlldm.cust_demog_dm
cusdemogdm 13:20:27: Program Completed...
```

If a program finishes unsuccessfully, an error file is usually written that indicates where the problem occurred in the process. There are some error messages written to the log file, such as ‘No output file specified’, that require no further explanation written to the error file.

Program error file

In addition to the daily log file, each program also writes its own detail flow and error messages. Rather than clutter the daily log file with these messages, each program writes out its errors to a separate error file unique to each execution.

The name and directory of the program error file is set in the configuration file (rdw_config.env). The directory defaults to \${MMHOME}/error. All errors and *all routine processing messages* for a given program on a given day go into this error file (for example, it will contain both the stderr and stdout from the call to RETL). All error files are encoded UTF-8.

The naming convention for the program’s error file defaults to the following:

- The program name
- The first filename, if one is specified on the command line
- The business virtual date for which the program was run

For example, all errors and detail log information for the `slsildmdm` program would be placed in the following file for the batch run of January 5, 2001:

```
${MMHOME}/error/slsildmdm.slsildmdm.txt.20010105
```



Note: When running RETL in an Oracle or a DB2 environment, operators can ignore a warning that appears in the error file. The property, ‘primaryindex’, is only created for Teradata environments at this time. An example follows:

```
WARNING: orawrite:0 : unknown property primaryindex
```

Schema files

RETL uses schema files to define an incoming or outgoing dataset. The schema file defines each column's data structure, which is then used within RETL to format/handle the data. For more information about schema files, see the latest RETL Programmer's Guide. For the following reasons, schema file names are hard-coded within each program:

- The schema files should not change on a day-to-day basis.
- If you are a retailer that has a Retek source application (for example, RMS, RCOM, ReSA, Merchandise Financial Planning, ReIM, and RPM) and you choose to use the application's extract modules packed with the application, schema files used in data extractions must be the same as the schema files used by the RDW datamart loading module.
- For retailers without other Retek applications, the provided text file must be formatted to the schema used by the RDW loading program. To determine how data must be formatted to match RDW schemas, see "Appendix A – Application Programming Interface (API) flat file specifications".

Resource files

RDW uses resource files so that the same RETL programs can run in various language environments. For each language, there is one resource file.

Resource files contain hard-coded strings that are used by RDW time load programs. The name and directory of the resource file is set in the configuration file, `rdw_config.env`. The default directory is `${MMHOME}/rfx/include`.

The naming convention for the resource file follows the two-letter ISO code standard abbreviation for languages (for example, `en` for English, `fr` for French, `ja` for Japanese, `es` for Spanish, and so on).

Command line parameters

In order for each RETL program in RDW to run, the input/output data file paths and names may need to be passed in at the Unix command line.

RDW

Datamart base fact load programs require passing in the `input_file_path` and `input_file_name`. Datamart dimension load programs do not require passing in any parameters because the input path/filename defaults to the following:

```
$DATA_DIR/(program name).txt.
```

See Chapter 8, “Program reference lists”, for a detailed listing of all programs and their command line parameters.

Partitioning

RETL partitioning divides the data into multiple segments, or partitions, based upon the number of logical partitions defined in RETL. See the latest RETL Programmer’s Guide for a recommendation as to how to properly set the number of partitions. Each processor is responsible for a portion of a dataset, rather than the entire dataset. As a result of this partition load method, the processing of the entire dataset is much faster than in a single-processor environment.

Partitioning operators work closely with parallel operators, so that the detailed operations of partitioning and parallelism are hidden from the application user. See the latest RETL Programmer’s Guide for details on setting the number of partitions and determining which operators default to serial and which to parallel.

Temporary directory

RDW utilizes a temporary directory to aid in RDW’s RETL program processing. This directory is used for various reasons by a few RDW programs that utilize RETL. The path/location of the temporary directory (`TEMP_DIR`) is set in RDW’s environment configuration script, `rdw_config.env`.

RETL also utilizes a temporary directory to aid in its own processing. The path/location of this temporary directory (`TEMPDIR`) is set in the RETL configuration file, `rfx.conf`.

Although both RDW and RETL have separate reasons for using the temporary directories mentioned above, the client may choose to use a single temporary directory that would be shared by RDW and RETL. Retek recommends, however, that the client use separate directories to alleviate confusion.

For more information on how RETL uses the temporary directory, see the latest RETL Programmer’s Guide.

The first time RDW batch is run

To ensure that the correct current load data is entered in `dm_recd_load_dt`, the following must be considered:

- Ensure that RETL is configured per the RETL documentation and the `rffx -v` command is run at the UNIX prompt to confirm that you are pointing to the proper RETL executable version.
- Verify that the RETL executable is in the path of your UNIX session by typing: `%which rffx`.
- RDW installation is successful and default data and time are properly installed.
- Chapters 1-4 of this Operations Guide are read and understood by the batch operator to understand the relationship between time tables and columns that are populated in dimension and facts.
- Batch dependencies are understood. See “Chapter 7 – Program flow diagrams” for more information about the RDW program flow and dependencies.
- `Curr_load_dt` is changed to the day before the first dimension/fact is loaded (for example, if you plan to load data and have all the items on the first day of history have a `dm_recd_load_dt` of ‘20000101’, then the `curr_load_dt` should be updated to ‘19991231’).
- Run `mt_prime.ksh` before the dimension and fact modules to update the `curr_load_dt` in `maint_load_dt_dm` table to the intended dimension/fact load date (for example, the `curr_load_dt` before `mt_prime.ksh` is ‘19991231’, and then after `mt_prime.ksh` the `curr_load_dt` is updated to ‘20000101’).

Typical `mt_prime.ksh` run

To run `mt_prime.ksh`, follow these steps:



Note: A *program prerequisite* is that the date entered in `curr_load_dt` must exist in the `TIME_DAY_DM` table.

- 1 Change directories to `${MMHOME}/rffx/src`.
- 2 At a Unix prompt enter:
`%mt_prime.ksh`

If the program runs successfully, the following results:

- 1 **Log file:** Today’s log file, `19991231.log`, contains the messages “Program started ...” and “Program completed successfully” for `mt_prime`.
- 2 **Data:**
 - a The text files in `${MMHOME}/rffx/etc` are updated.
 - b The `PROGRAM_STATUS_DM` table is updated. The status is updated to ‘ready’ for programs with a ‘complete’ status.
 - c `Curr_load_dt` in the `MAINT_LOAD_DT_DM` table is updated to the previous `curr_load_dt` plus one.
- 3 **Error file:** The program’s error file, `mt_prime. 19991231`, contains the standard RETL flow (ending with “All threads complete” and “Flow ran successfully”) and no additional error messages.

Typical run and debugging situations

The following examples illustrate typical run and debugging situations for each type of program within RDW. The log, error, and so on file names referenced below assume that the program is run on the business virtual date of March 9, 2001. See the previously described naming conventions for the location of each file.

RDW dimension load

To run prdpimdm.ksh:

- 1 Change directories to `${MMHOME}/rfx/src`.
- 2 At a Unix prompt, enter:
`%prdpimdm.ksh`

If the program runs successfully, the following results:

- 1 **Log file:** Today's log file, 20010309.log, contains "Program starting ...", various informational messages, and "Program completed successfully..." messages for prdpimdm.
- 2 **Data:** The records from the source file `${MMHOME}/data/prdpimdm.txt` are loaded into the target table.
- 3 **Error file:** The program's error file, prdpimdm.20010309, contains the program's standard RETL flow (with "All threads complete" and "Flow ran successfully"), standard database output for dropping/updating tables, and no additional error messages.
- 4 **Program status control:** The PROGRAM_STATUS_DM table is updated to 'completed' where program_name = prdpimdm and file_name = `${MMHOME}/data/prdpimdm.txt`.
- 5 **Reject file:** Reject files are not created for RDW programs.
- 6 **Bookmark file:** The bookmark file, prdpimdm.bkm.20010309, does not exist.

If the program does *not* run successfully, the following results:

- 1 **Log file:** Today's log file, 20010309.log, does not have the "Program completed successfully..." message for prdpimdm.
- 2 **Data:** Some of the records from the source file `${MMHOME}/data/prdpimdm.txt` may be loaded into the target table.
- 3 **Error file:** The program's error file, prdpimdm.20010309, contains the program's RETL flow and any additional error messages.
- 4 **Program status control:** The PROGRAM_STATUS_DM table is updated to 'error' where program_name = prdpimdm and file_name = `${MMHOME}/data/prdpimdm.txt`.
- 5 **Reject file:** Reject files are not created for RDW programs.
- 6 **Bookmark file:** The bookmark file, prdpimdm.bkm.20010309, may exist. No bookmark file is created if the program did not make it past the first unit of work within the program or the program does not use restart and recovery.

To re-run the program, perform the following actions:

- 1 Determine and fix the problem causing the error.
- 2 If you wish to re-run the program from the beginning, remove the program's bookmark file.

- 3 Update the PROGRAM_STATUS_DM table to 'ready' where program_name = prdpimdm and file_name = \${MMHOME}/data/prdpimdm.txt.
- 4 Change directories to \${MMHOME}/rfx/src. At a Unix prompt, enter:
`%prdpimdm.ksh`

RDW fact load

To run vchreschddm.ksh:

- 1 Change directories to \${MMHOME}/rfx/src.
- 2 At a Unix prompt, enter:
`%vchreschddm.ksh ${MMHOME}/data/vchreschddm.txt`

If the program runs successfully, the following results:

- 1 **Log file:** Today's log file, 20010309.log, contains "Program starting...", various informational messages, and "Program completed successfully" messages for vchreschddm.
- 2 **Data:** The records from the source file \${MMHOME}/data/vchreschddm.txt are loaded into the target table.
- 3 **Error file:** The program's error file, vchreschddm.vchreschddm.txt.20010309, contains the program's standard RETL flow (with "All threads complete" and "Flow ran successfully"), standard database output for updating tables, and no additional error messages.
- 4 **Program status control:** The PROGRAM_STATUS_DM table is updated to 'completed' where program_name = vchreschddm and file_name = \${MMHOME}/data/vchreschddm.txt.
- 5 **Reject file:** Reject files are not created for RDW programs.
- 6 **Bookmark file:** The bookmark file, vchreschddm.vchreschddm.txt.bkm.20010309, may exist. No bookmark file is created if the program did not make it past the first unit of work within the program or if the program does not use restart and recovery.

If the program does *not* run successfully, the following results:

- 1 **Log file:** Today's log file does not contain the "Program completed successfully" message for vchreschddm.
- 2 **Data:** Some of the records from the source file \${MMHOME}/data/vchreschddm.txt may be loaded into the target table.
- 3 **Error file:** The program's error file, vchreschddm.vchreschddm.txt.20010309, contains the program's RETL flow, and any additional error messages.
- 4 **Program status control:** The PROGRAM_STATUS_DM table is updated to 'error' where program_name = vchreschddm and file_name = \${MMHOME}/data/vchreschddm.txt.
- 5 **Reject file:** Reject files are not created for RDW programs.
- 6 **Bookmark file:** The bookmark file, vchreschddm.vchreschddm.txt.bkm.20010309, may exist. No bookmark file is created if the program did not make it past the first unit of work within the program or if the program does not use restart and recovery.

To re-run the program:

- 1 Determine and fix the problem causing the error.

- 2 If you wish to re-run the program from the beginning, remove the program's bookmark file.
- 3 Update the PROGRAM_STATUS_DM table to 'ready' where program_name = vchreschddm and file_name = \${MMHOME}/data/vchreschddm.txt.
- 4 Change directories to \${MMHOME}/rfx/src. At a Unix prompt, enter:

```
%vchreschddm.ksh ${MMHOME}/data/vchreschddm.txt
```


Chapter 6 – RDW interfaces

This chapter provides a functional summary of data interfaces with RDW.

RDW receives information from various source systems. For the majority of RDW functionality, there are Retek applications that can serve as these source systems. These Retek applications are the following:

- Retek Merchandising System (RMS)
- Retek Sales Audit (ReSA)
- Retek Customer Order Management (RCOM)
- Retek Invoice Matching (ReIM)
- Retek Price Management (RPM)
- Retek Financial Merchandise Planning

Packaged with these Retek applications is the code to extract data from these systems and provide that data in a text file format which RDW can recognize and then load into RDW tables via RDW RETL programs.

Retailers who have not purchased Retek applications, or have purchased them and simply chosen not to use them as an RDW data source, can use their own source systems and code to provide RDW with the identical text files that the Retek source applications provide. APIs/schema files associated with this method are illustrated in “Appendix A – Application Programming Interface (API) flat file specifications”.

The following interfaces are described in this section:

- Retek Merchandising System (RMS)
 - Dimension data
 - Fact data
- Retek Invoice Matching (ReIM)
- Retek Price Management (RPM)
- Retek Sales Audit (ReSA)
- Retek Merchandise Financial Planning
- Retek Customer Order Management (RCOM)
 - Dimension data
 - Fact data
- Client-supplied data:
 - Customer account dimension
 - Customer geographic dimension
 - Customer and product cluster dimensions
 - Plan Season dimension
 - Market data facts and dimensions
 - Space allocation facts
 - Store traffic facts
 - Two of six supplier compliance facts text files
 - Quality control
 - Missed scheduled deliveries
 - Ad-hoc media transformations
- Loaded at install: Voucher age dimension and Time like for like transformations

All data comes into RDW as text files. See “Appendix A – Application Programming Interface (API) flat file specifications”, for a complete list of RDW’s API specifications and their business requirements.

Retek Merchandising System

The Retek Merchandising System (RMS) can be the principle source of RDW's dimension and fact data. RMS is the retail client's central transactional processing system. RMS data feeds to RDW can be broken down into dimensions and facts. General descriptions of each are contained in this section.

Dimension data

RMS can be the sole source of organization and product dimension data, and it supplies the majority of other dimension data as well.

The dimension data process extracts current dimension data from RMS using RETL scripts that are packaged with RMS. The extracted data is outputted to text files. After these text files are moved to the RDW server, RETL then compares the data in the text file with the historical dimension data in RDW, and thereafter inserts/updates the dimension changes back into RDW. This comparison eliminates the need to capture dimension changes as they occur in the source system during the day.

RMS supplied dimensions include the following: Company, Competitor, Currency Code, Employee, Item-Location Trait Cross, Item-Supplier-Location Cross, Organization, Product (including attributes, such as differentiators), Product Season, Reason, Regionality, Sub-Transaction Type, Supplier, Tender Type, and Total Type.

RMS can be the source of two of the three types of time that RDW supports: fiscal 454 time and fiscal 454 time with a Gregorian time calendar. Retailers can supply the other RDW-supported time functionality: 13 period time. For more information on how time is loaded to RDW, see the RDW Database Installation Guide.

Fact data

RDW's fact data process extracts fact data from RMS using RETL scripts that are packaged with RMS. The extracted data is outputted to text files. After these text files are moved to the RDW server, RETL takes the data in the text files and performs the appropriate transformation, inserts and updates to the fact datamart tables.

RMS supplied facts include the following: Competitor Pricing, Cost, Exchange Rates, Inventory Adjustments, Inventory Position, Inventory Receipts, Inventory Transfers, Markdowns, Net Cost, Pricing, Profit on Base Cost, Return to Vendor, Sales Forecasts, Stock Ledger, Supplier Availability, Supplier Compliance, Supplier Contract, Unavailable Inventory, and Currency Exchange Rates.

A note about local currency and facts

Many RDW clients conduct business in a multi-currency environment. While querying sales facts, for instance, a client may want to see the values in the common local currency of a group of stores in one country, or see values aggregated across all their stores from a number of countries. In order to provide clients both accuracy and flexibility in storing currency values, both local and primary currency values are stored in most of the RDW fact tables. A client using multiple currencies would have any fact that is stored by loc_key held in that location's local currency for that day, side-by-side with a column for that fact converted to primary currency. A client using only one currency would have only the primary currency column populated, leaving the local column null. This currency storage strategy is accomplished by either RETL fact extraction code or legacy fact interfaces, which generate text files that include both local and primary currency values for loading into the datamart tables.

Retek Invoice Matching (ReIM)

ReIM is a solution that provides all of the data necessary to support the invoice verification function, minimizing interface development and maintenance costs. ReIM can serve as the source of invoice cost fact data. ReIM extracts data using code packaged with ReIM. ReIM then writes one text file for RDW called sincilddm.txt. After this text file is moved to the RDW server, it is made available for RETL. RETL uses the data from the text file to perform the applicable inserts and updates to the invoice cost fact datamart table.

Retek Price Management (RPM)

RPM is a solution that suggests and assists with pricing decisions. RPM can serve as the source of promotion dimensional data. RPM extracts data using code packaged with RPM. RPM then writes three text files for RDW. After these text files are moved to the RDW server, they are made available for RETL to use the data. RETL uses the data from the text file to perform applicable inserts and updates to the promotion dimension datamart table.

Retek Sales Audit (ReSA)

ReSA is a flow through application that accepts 'raw' point of sale information and provides 'clean' data to downstream applications, such as the RDW.

Retek Sales Audit writes four flat (ASCII text) files for RDW—one each for transaction item data (file type RDWT), transaction tender data (RDWF), store total data (RDWS), and cashier or register over or short data (RDWC). Each of these files is then made available for processing by RETL scripts to extract data. These scripts are packaged with RMS. A part of the fact processing, these RETL scripts run within the RMS batch-processing schedule. The extracted data is outputted to text files. On the RDW servers, RDW RETL programs take the data in the text files and perform the appropriate inserts and updates to the fact datamart tables.

ReSA supplied facts include the following: Sales and Return Transactions (including Pack Sales), Sales Productivity, Loss Prevention Transactions, and Loss Prevention Totals (Tender Transactions, Cashier Over or Short, User-Defined Totals).

Register dimension data is derived from ReSA RDWF (tender transaction) file, by way of the ttldmdm.ksh fact DM script.

In addition to the four flat files described previously, ReSA serves as a source for voucher fact data for RDW. Three programs extract fact data for voucher movement, escheated vouchers, and outstanding vouchers. See Chapter 8 for more details about these programs.

Retek Merchandise Financial Planning

Retek Merchandise Financial Planning can serve as the source of planning fact data such as planned sales to a retailer. The application extracts data using code packaged within it. The application then writes the following two text files for RDW:

- ploblwdm.txt for original planning
- plcblwdm.txt for current planning

After these text files are moved to the RDW server, they are made available for RETL to use the data from the text files to perform the appropriate inserts and updates to the planning fact datamart tables. Data from Merchandise Financial Planning does not need to be loaded daily; it can be loaded periodically.

Retek Customer Order Management

Retek Customer Order Management (RCOM) is a centralized solution across all channels, that manages customer interactions, purchases, history on the Web, call center/catalog, kiosk, and the stores using one common infrastructure with a single view of inventory.

Dimension data

RCOM can serve as the source of customer and customer demographic information, customer request, customer order management codes, media, selling item, customer service representative (CSR), call center, ship-to-addresses, carrier and carrier service dimension data.

RCOM extracts data using RETL scripts that are packaged with RCOM. RCOM then writes a text file for each dimension to RDW. After the text files are moved to the RDW server, RETL scripts load the data in the text into the database. In some cases, only changed dimension data is provided by RCOM, along with a flag to indicate whether the record is to be inserted, updated, or closed. In most other cases, RCOM provides the full snapshot of dimension data. The RETL scripts then compare the data in the text file with the historical data in RDW and make the applicable updates/inserts into the dimension table as necessary in RDW.

Fact data

RCOM can serve as the source of customer order facts (header, line, value added service line, promotion), media/item/selling item facts, and service and catalog requests.

RCOM extracts data using RETL scripts that are packaged with RCOM. RCOM then writes a text file for each set of facts to RDW. After the text files are moved to the RDW server, RETL scripts either insert or update the fact information in RDW.

RCOM sends shipped customer order line information to ReSA. ReSA then sends those order line transactions to RDW via the RDWT sales flat file.

Client-supplied data

The RDW provides programs and tables for the areas of functionality described in this section. However, there currently is no Retek source system available to provide data for these functional areas. Clients need to supply the data via text files. These text files will be used as the inputs to process and load data into RDW datamart tables. To see the location of client-supplied data in the RDW program schedule, see Chapter 7. For more business content information regarding the following functional areas of client-supplied data, see “Appendix A – Application Programming Interface (API) flat file specifications”.

- Customer account dimension
- Customer geographic dimension
- Customer and product cluster dimension
- Plan season dimension
- Market data facts and dimensions
- Space allocation facts
- Store traffic facts
- Two of six supplier compliance facts text files
 - Quality control
 - Missed schedule deliveries
- Ad-hoc media dimension

The tables representing the following areas of functionality are loaded once at installation: voucher age dimension and time like for like transformations. See the RDW Database Installation Guide for more information.

Chapter 7 – Program flow diagrams

This chapter presents flow diagrams for all RDW dimension and fact data processing beginning with the data files provided by the source systems. Included are descriptions of the source system's data file that is required to be present, along with the RDW program or process that interfaces with the source data file. After initial interface processing of the data file, the diagrams illustrate the flow of the data into the respective datamarts.

Before setting up an RDW program schedule, familiarize yourself with the functional and technical constraints associated with each program. Read through the RDW Database Installation Guide and “Chapter 8 – Program reference lists”, of this operations guide for more details.

Batch scheduling

The following explains the order constraints of the RDW batch schedule. This section includes:

- Overall batch schedule, including schedule timing and when to run programs—daily, weekly, ad hoc, and so on
- Functional interdependencies, including functional constraints, such as that fact programs must run after dimension programs

Setting up the batch schedule



Note: The number of programs that can be run in parallel at any given time is dependent upon the client's hardware capacity.

The following discussion applies to Oracle and Teradata RDW clients. DB2 clients must refer to the “Batch schedule for DB2 clients” section later in this chapter. On a typical batch production run, the pre-batch maintenance programs must always run first. What runs next, as long as the client follows the batch dependencies in the flow diagrams, is up to the client.

For example, product dimension programs, such as `prditmdm.ksh` and `prditmldm.ksh`, can be run in parallel after their respective pre-dependencies. Fact programs, such as `prcilddm.ksh`, can run in parallel with other, unrelated fact programs provided their respective pre-dependencies (including dimension predecessors) complete successfully first.

The batch flows on the following pages are best read from top to bottom. Such a review of the RDW batch schedule allows clients to both set up program dependencies and to optimize their batch window through the concurrent running of unrelated programs.

LANGUAGE variable

For `rdw_config.env`, the `LANGUAGE` variable refers to the language that RDW expects to be used for writing descriptions to the dimension tables, the language used in the front end, and so on. Currently, RDW supports English, French, Japanese, and Spanish. The valid `LANGUAGE` values include the following:

- `en=English`
- `fr=French`

- ja=Japanese
- es=Spanish

rdw_config.env settings

The RDW Database Installation Guide refers to two important RETL environment variables that the client must set in the rdw_config.env file: LOAD_TYPE and SCHEDULE_TYPE.



Note: Clients must weigh the performance benefit of these settings before running their batch schedule.

- LOAD_TYPE refers to the load method that RETL uses to load data to the database, and is only used with Oracle or DB2 DBMS.
 - **LOAD_TYPE=conventional:** loads the data using the conventional SQL-loader method for Oracle, or the DB2LOADER utility for DB2.
 - **LOAD_TYPE=direct:** loads the data using the direct SQL_loader method for Oracle, or the Autoloader utility for DB2. Note that there is one exception to this rule: For DB2 clients, even when LOAD_TYPE is set to direct, all dimension programs (except dimension matrix programs) continue to use the DB2LOADER utility, not the Autoloader.
- The SCHEDULE_TYPE is only used for DB2 clients, and only affects DBMS loading where LOAD_TYPE=direct. If LOAD_TYPE=conventional, SCHEDULE_TYPE is ignored. Valid values for SCHEDULE_TYPE are sequential or parallel.
 - When SCHEDULE_TYPE is set to sequential, the following assumptions apply:
 - There is one tablespace for all dimension tables
 - There is one tablespace for all dimension matrix tables
 - There is one tablespace for each fact table
 - There are three user data tablespaces for temp tables

DB2 tablespaces are set up in this manner per the RDW base install. Even though all dimension programs (except dimension matrix) can be scheduled to run parallel, all dimension matrix programs and all fact programs must be run one at a time.

- When SCHEDULE_TYPE is set to parallel, the dimension matrix programs and fact programs can be running in parallel, but a tablespace must be created for each dimension matrix table and each fact temp table. This step requires the slight customization of RDW install scripts/procedures, and some potential customization of the RDW RETL code. Contact Retek Customer Care for assistance with this type of custom work.

Merchandise Financial Planning to RDW scheduling

Original and current plan data from Retek Merchandise Financial Planning are only loaded into RDW periodically. See Chapter 6, “RDW Interfaces”, for more details on the flow of data from Merchandise Financial Planning to RDW.

Data from undefined sources

There are no pre-defined sources for some functional areas such as Geographic Dimension, Space Allocation, and Store Traffic fact data. User-defined processes must populate the text files for these areas before their respective loading programs run.

RDW batch schedule for DB2 clients only

Because of DB2's unique loading requirements, RDW uses both the db2write and autoload utilities. The db2write utilities are used to write smaller sets of data. To help enhance load performance speed, the autoload utilities are used for larger sets of data.

Note that the use of autoloader has important ramifications with regard to the client's ability to read or write in parallel. When autoloader is used, the utilities lock the entire table space. Because any tables that reside in the locked tablespace become inaccessible, sequential processing becomes mandatory.

Note that in the base setup of RDW, programs are set up and scheduled to run in the following ways:

- Dimension programs use db2write utilities and can be run in parallel.
- Dimension matrix programs use autoload utilities and must run in sequential order.
- As shipped in base, all fact programs use autoload utilities and must run in sequential order (see the section, "config.env settings" earlier in this chapter). Even though the fact programs are running in sequential order, some programs use multiple temp tables for reading/writing. Those temp tables need to sit in separate tablespaces.



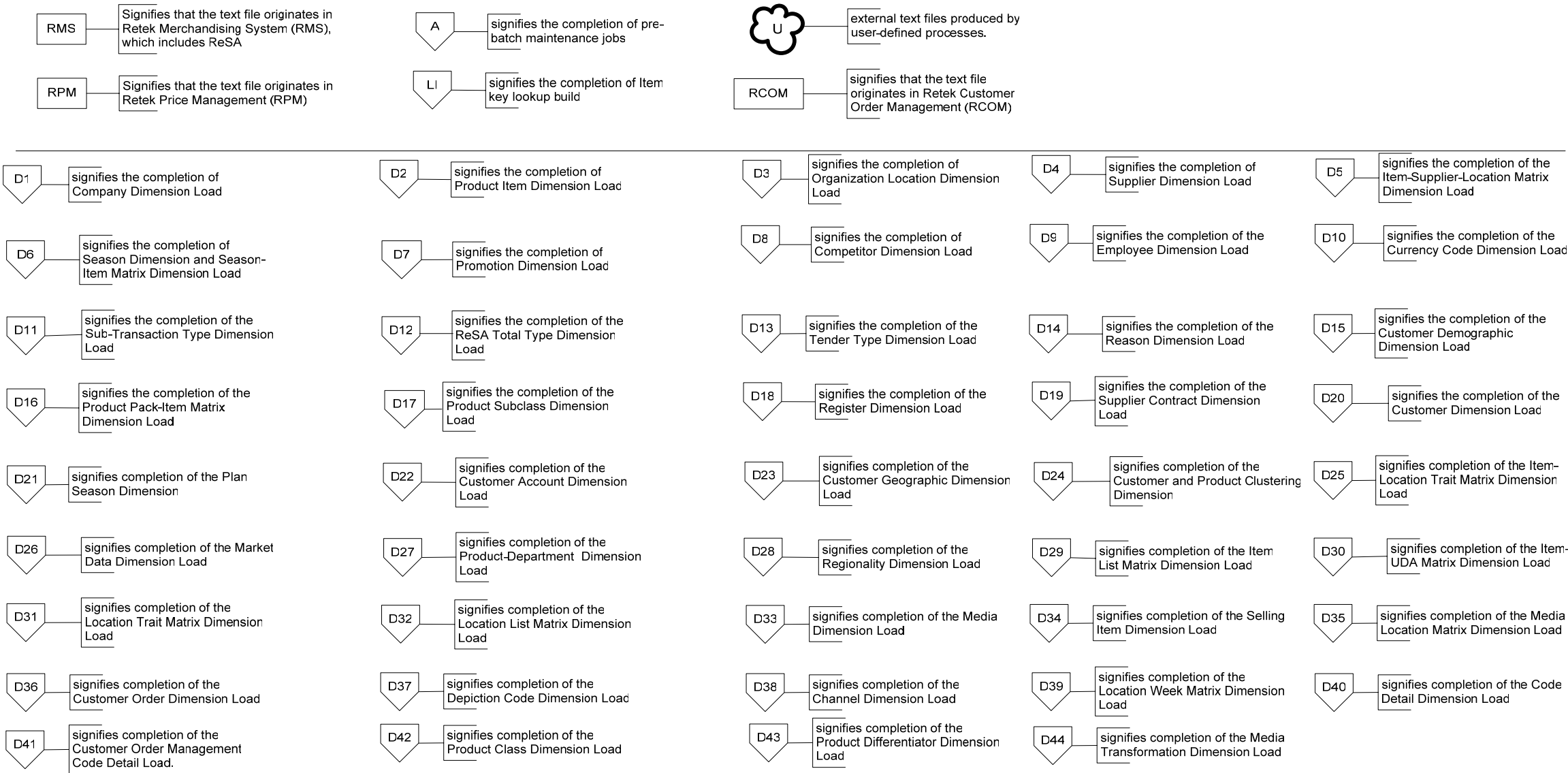
Note: If a client wants to run different fact datamarts in parallel, it must configure its user tablespaces to its specific processing needs, as well as modify the base code to write to the correct user tablespace.

For more information about the db2write and the autoload utilities, refer to DB2 documentation.

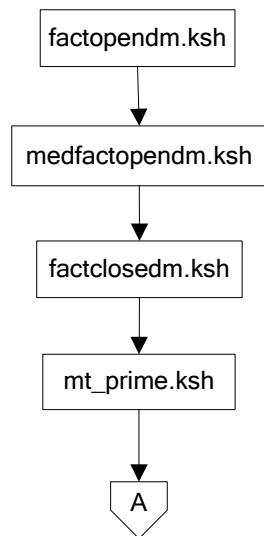
Program flow diagrams

Diagrams of RDW's program flows begin on the next page.

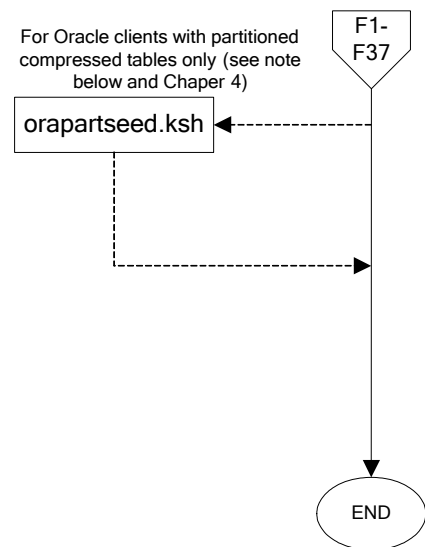
Legend: RDW dimension programs



Pre-batch
maintenance

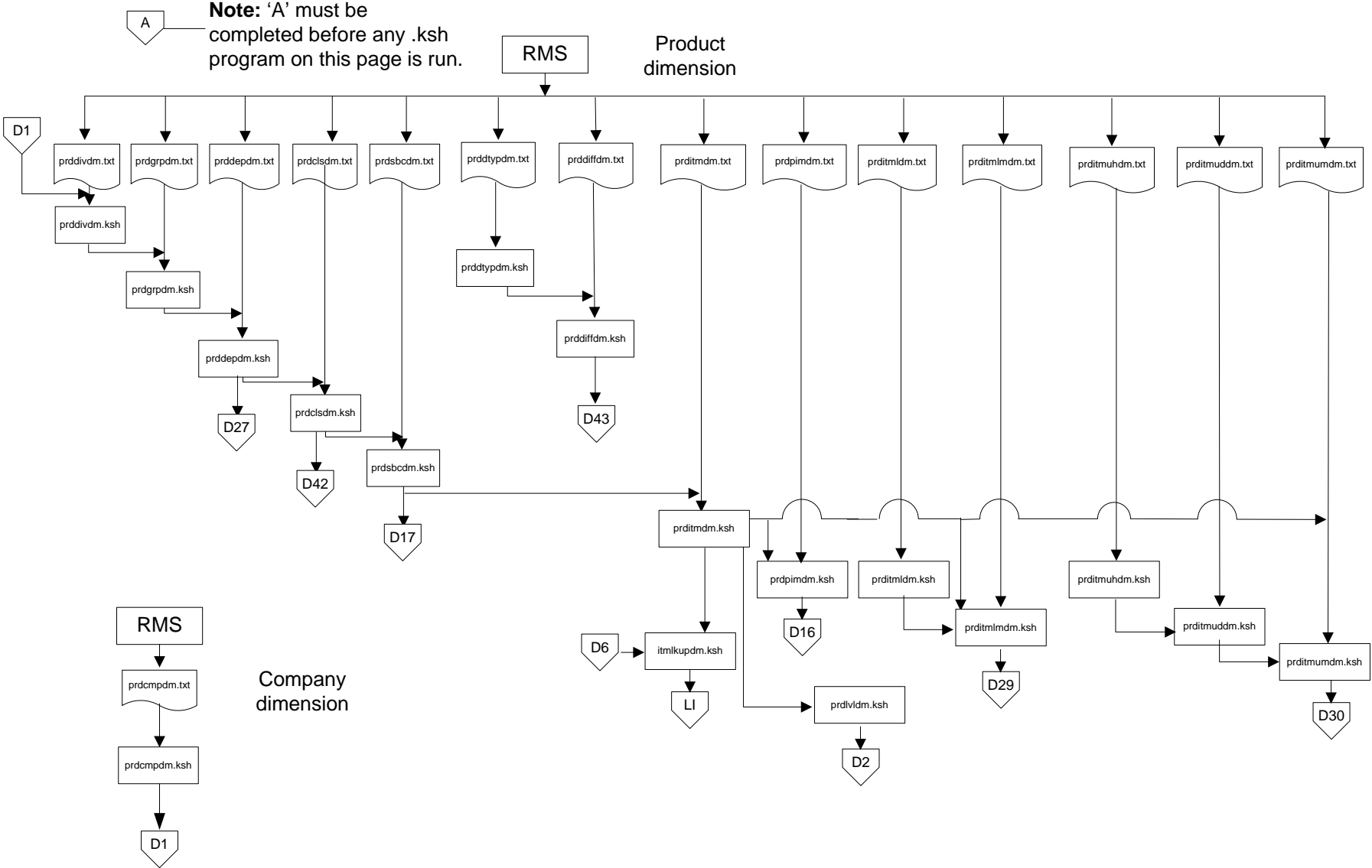


Post-batch
maintenance

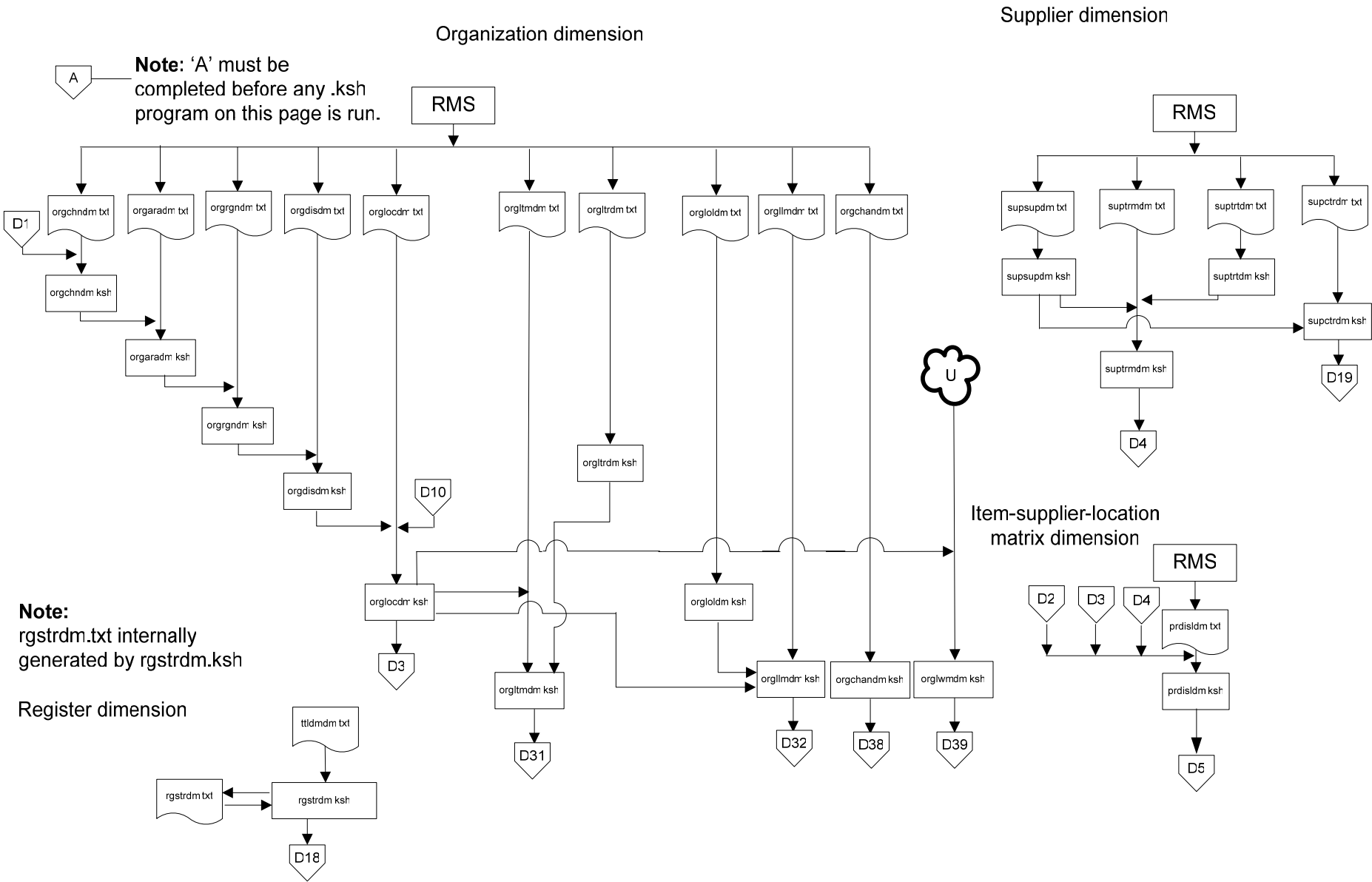


Note:
Orapartseed.ksh is an optional program used by Oracle clients only. The program affects compressed, partitioned datamart tables. See the chapter, "Compression and partitioning," for a more detailed explanation of seeding.

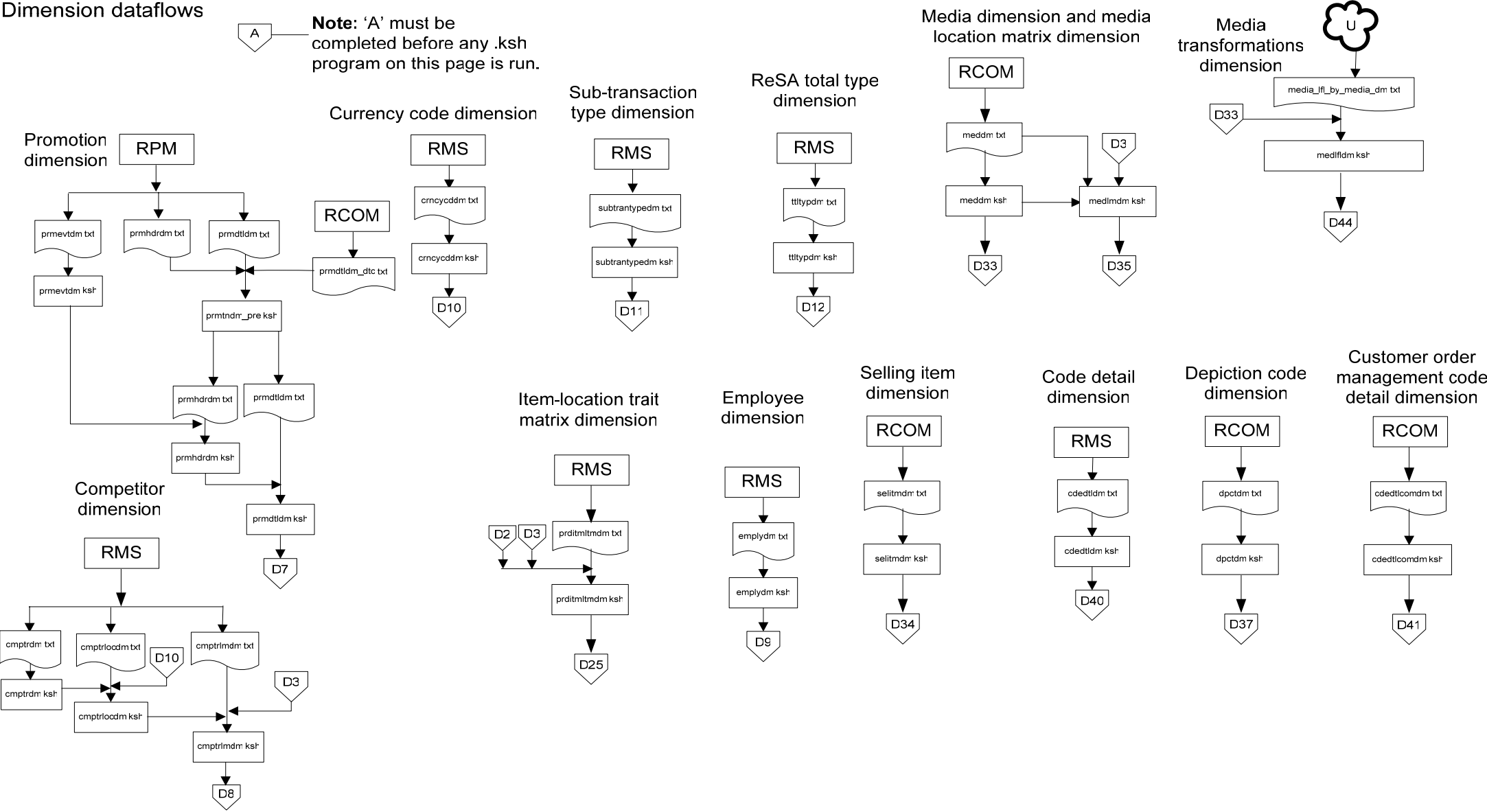
Dimension dataflows



Dimension dataflows

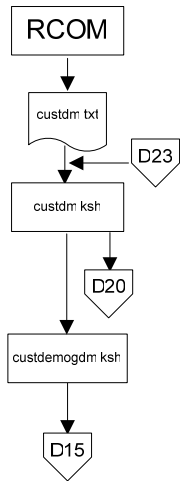


Dimension dataflows

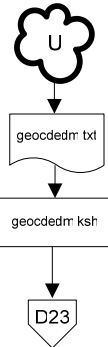


Dimension dataflows

Customer and customer demographic dimension

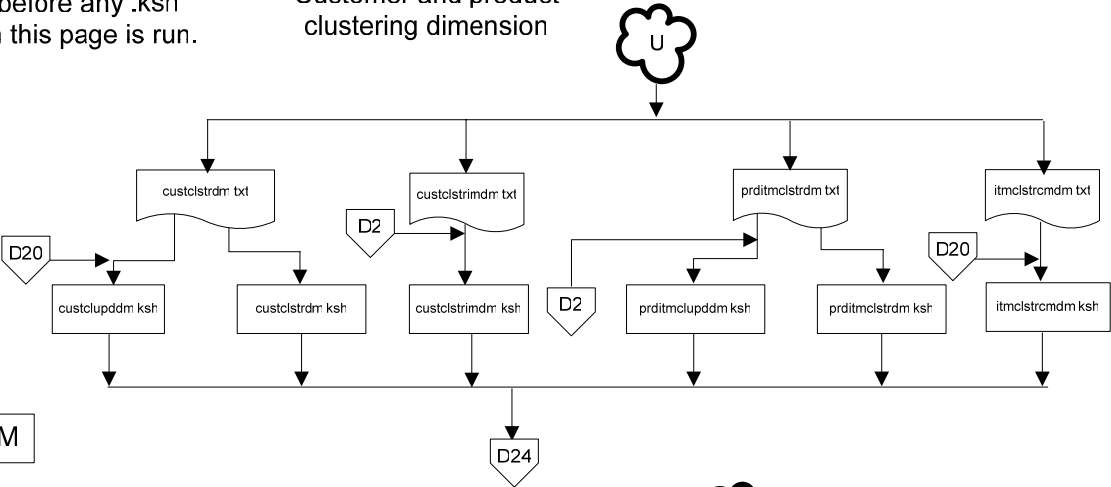


Customer geographic dimension

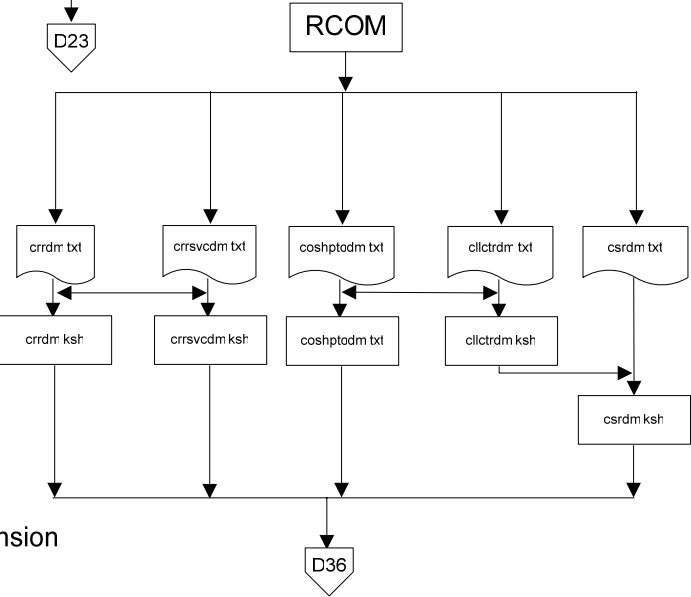


Note: 'A' must be completed before any .ksh program on this page is run.

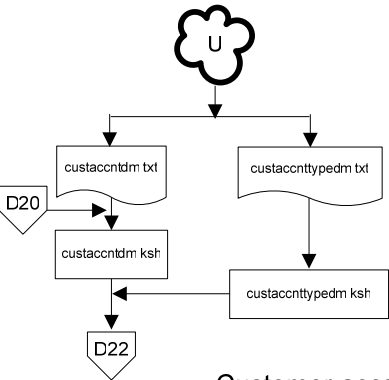
Customer and product clustering dimension




Customer order dimension

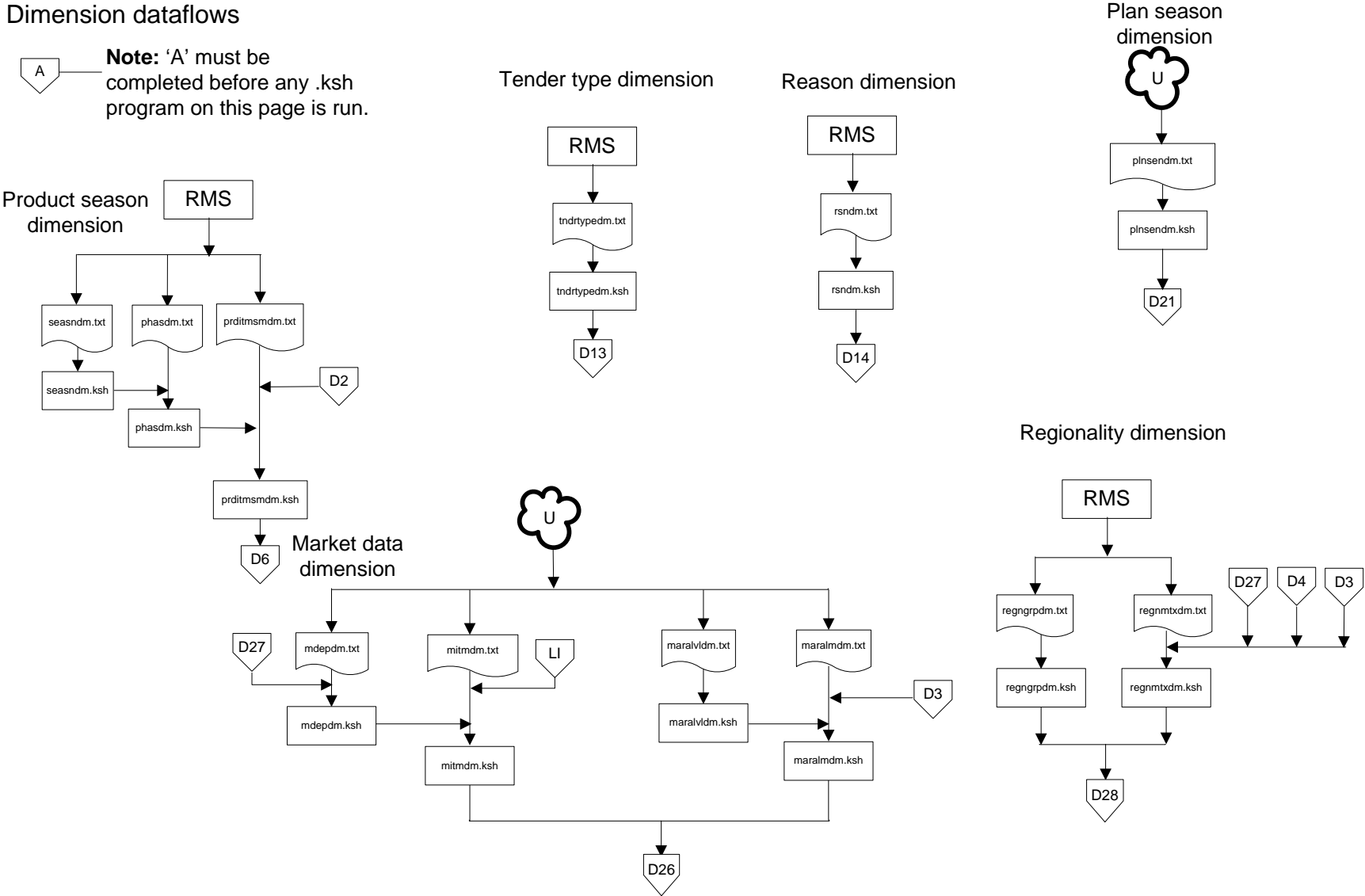


Customer account dimension



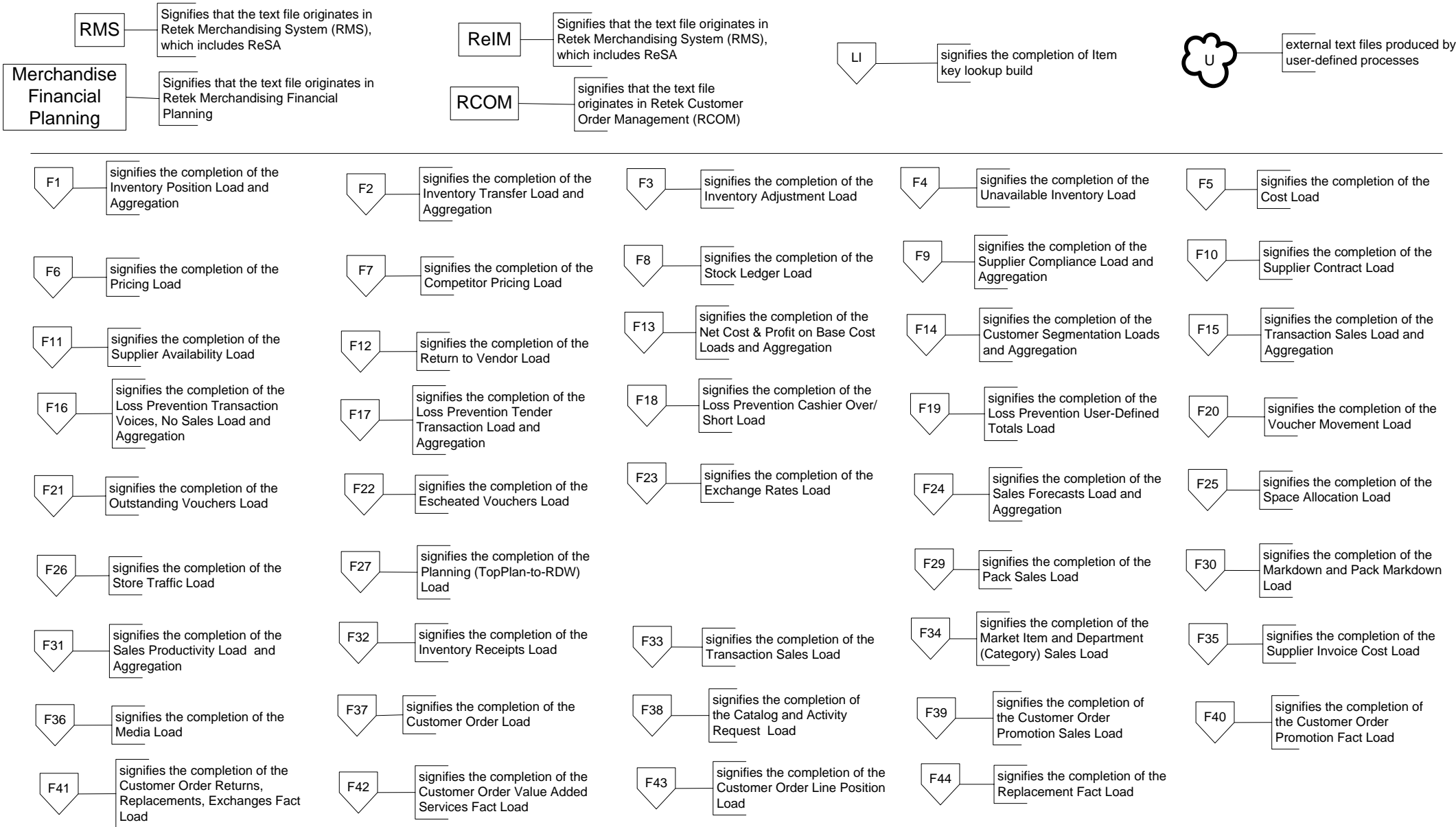
Dimension dataflows

 **Note:** 'A' must be completed before any .ksh program on this page is run.

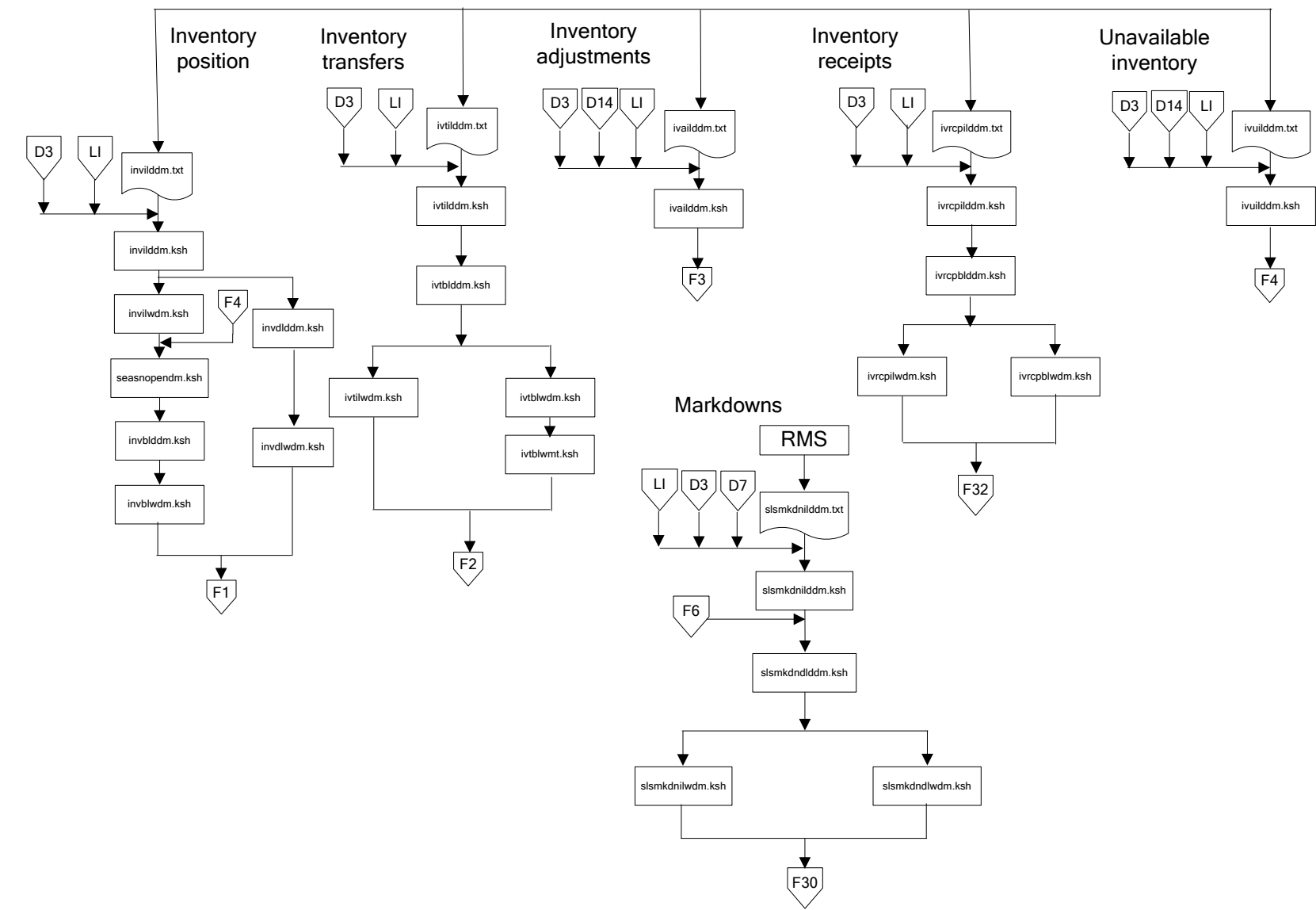


Legend: RDW fact programs

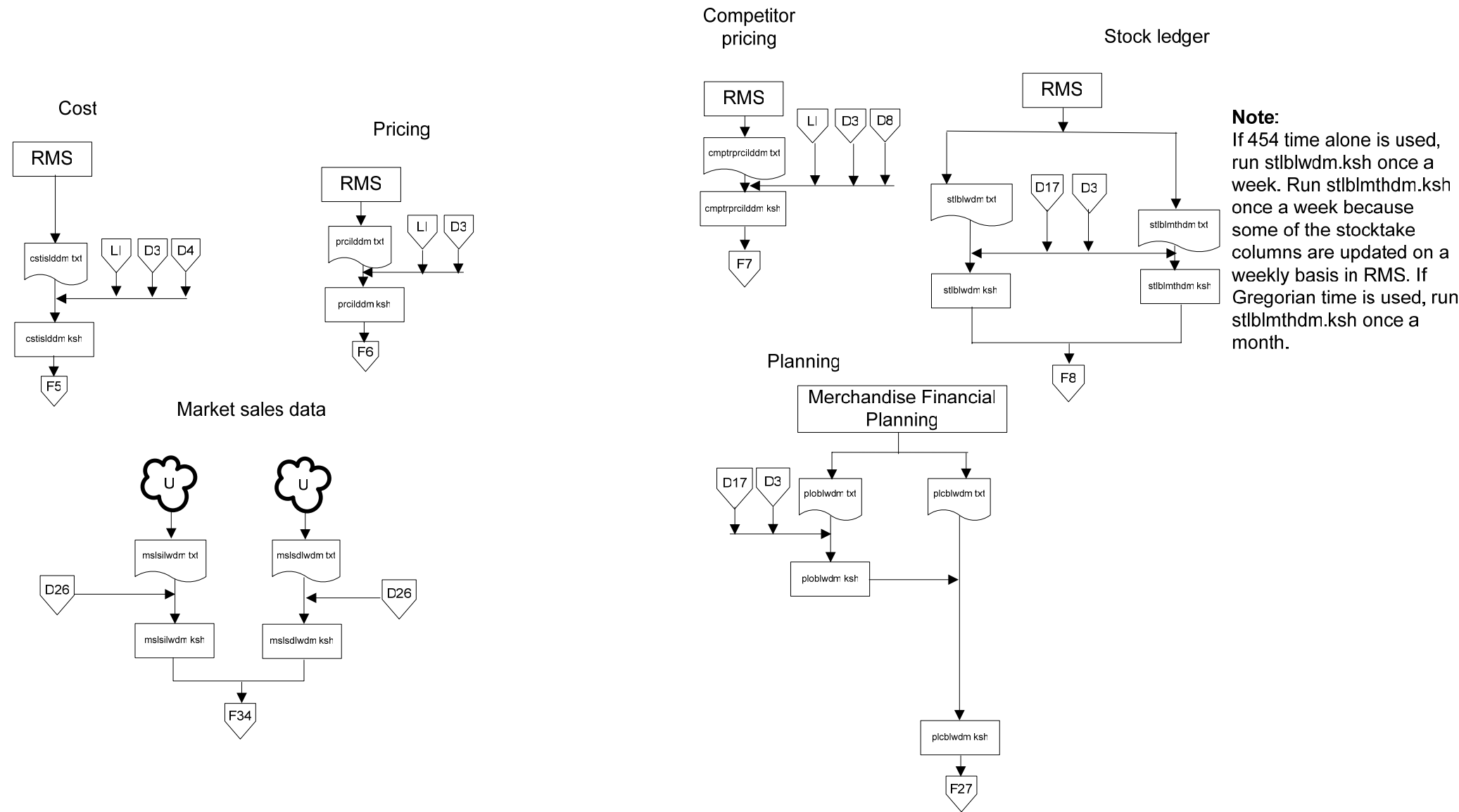
The legend for RDW’s fact program flows resides on the next page.



Fact dataflows

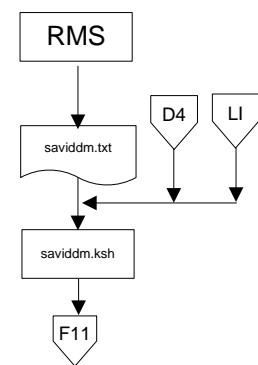


Fact dataflows

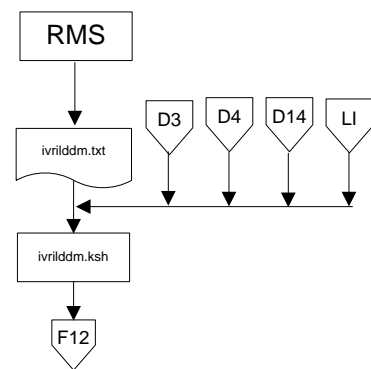


Fact dataflows

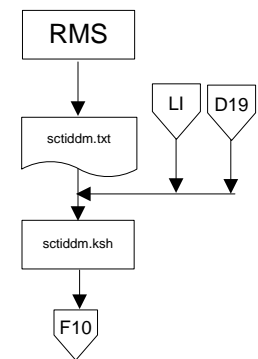
Supplier availability



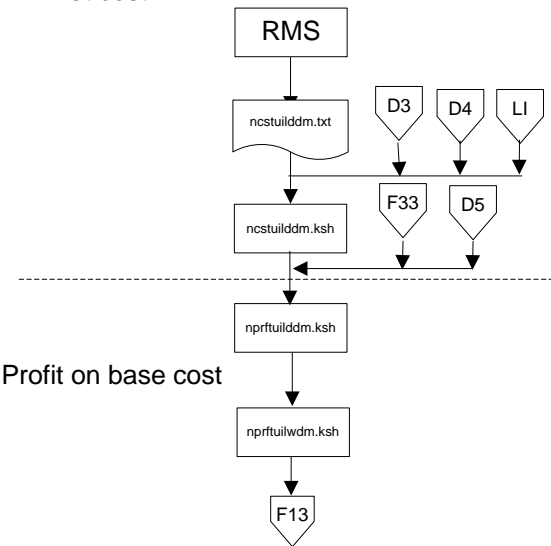
Return to vendor



Supplier contract

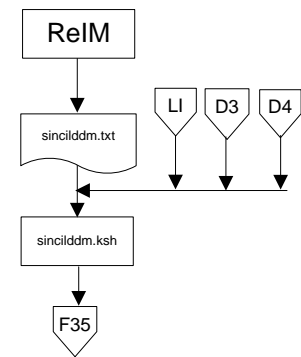


Net cost



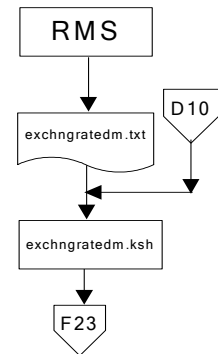
Profit on base cost

Supplier invoice cost

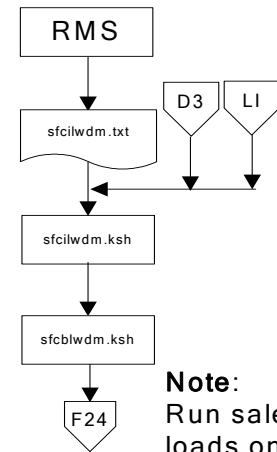


Fact dataflows

Exchange rates

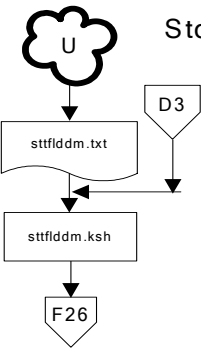


Sales forecasts

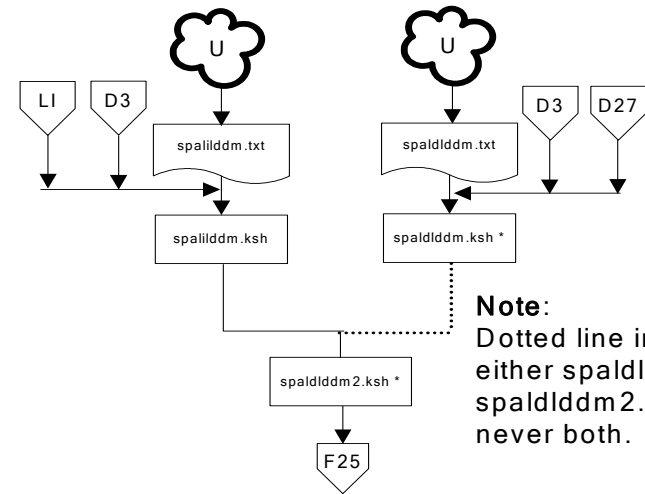


Note:
Run sales forecast fact loads once weekly.

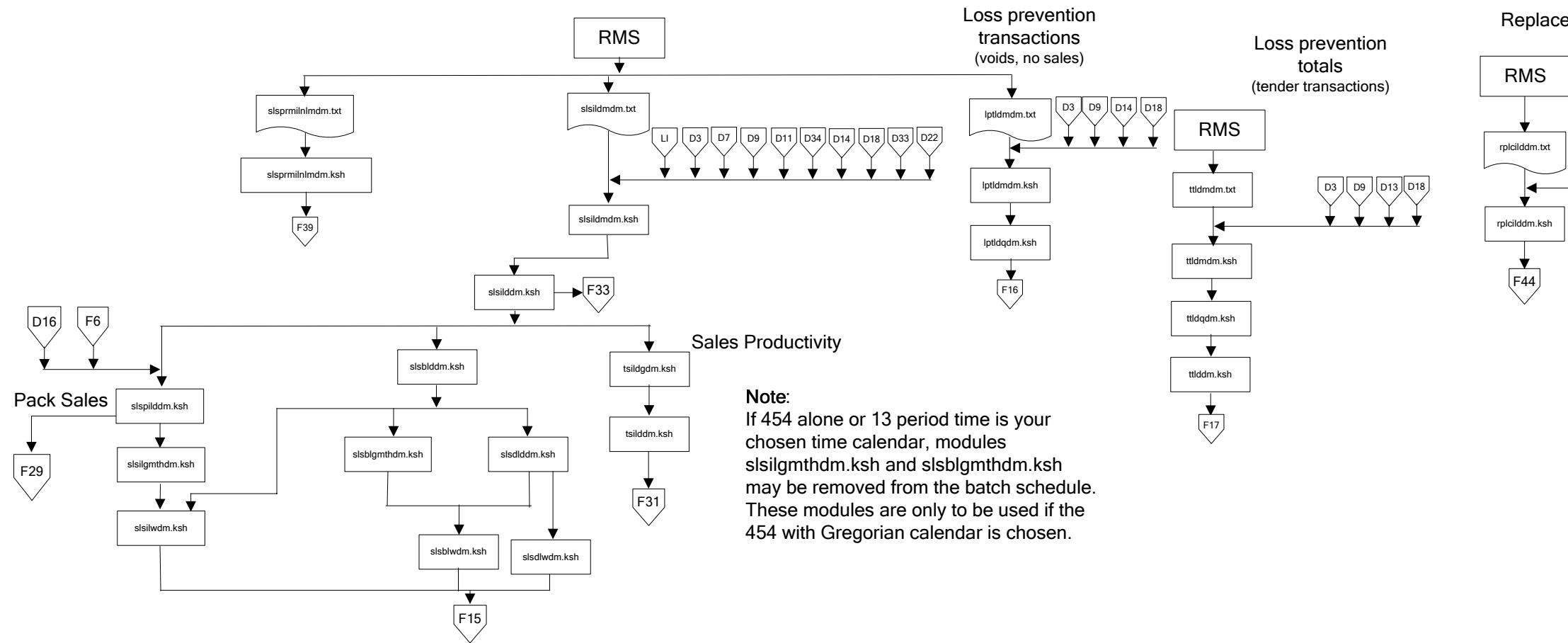
Store traffic



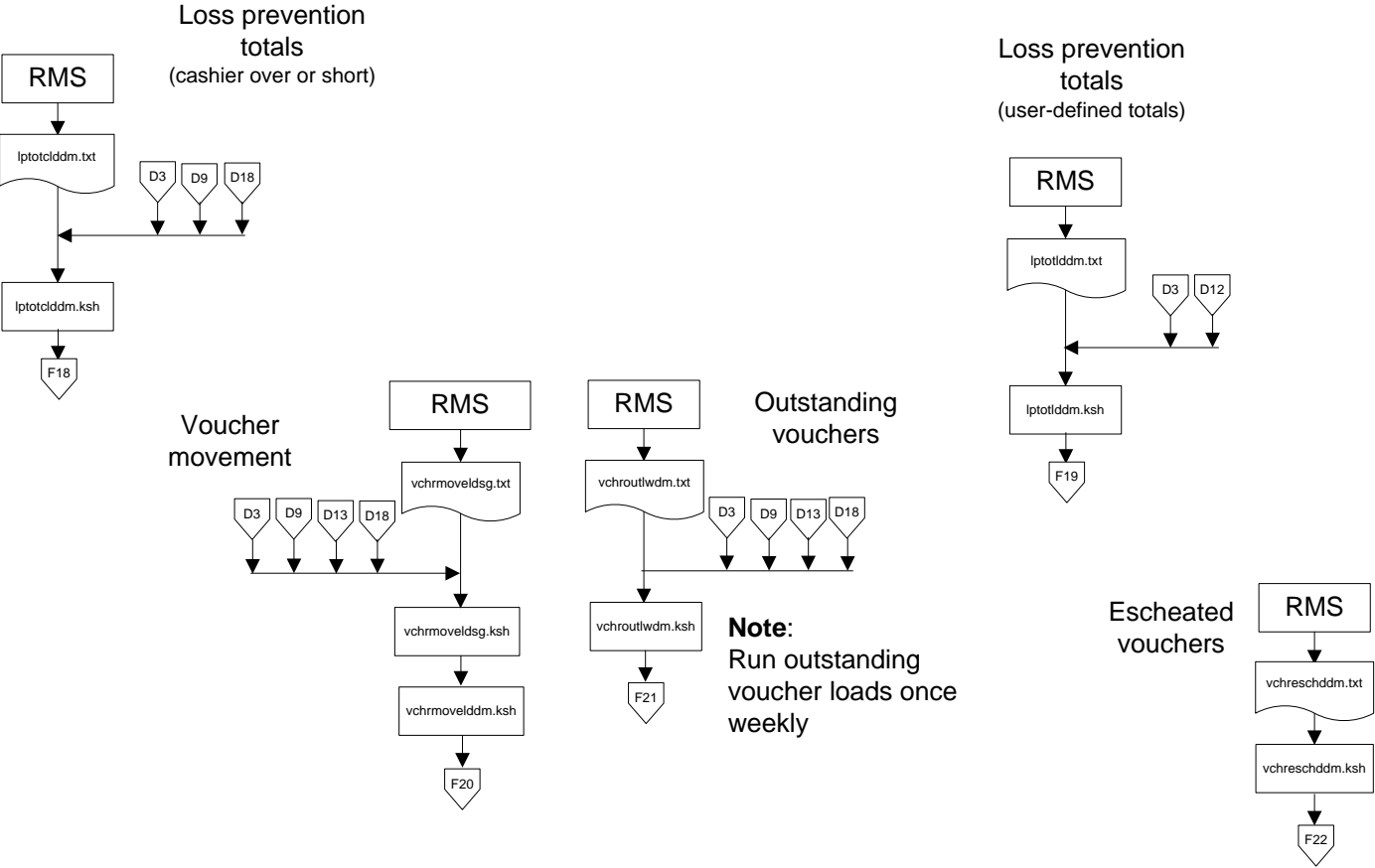
Space allocation



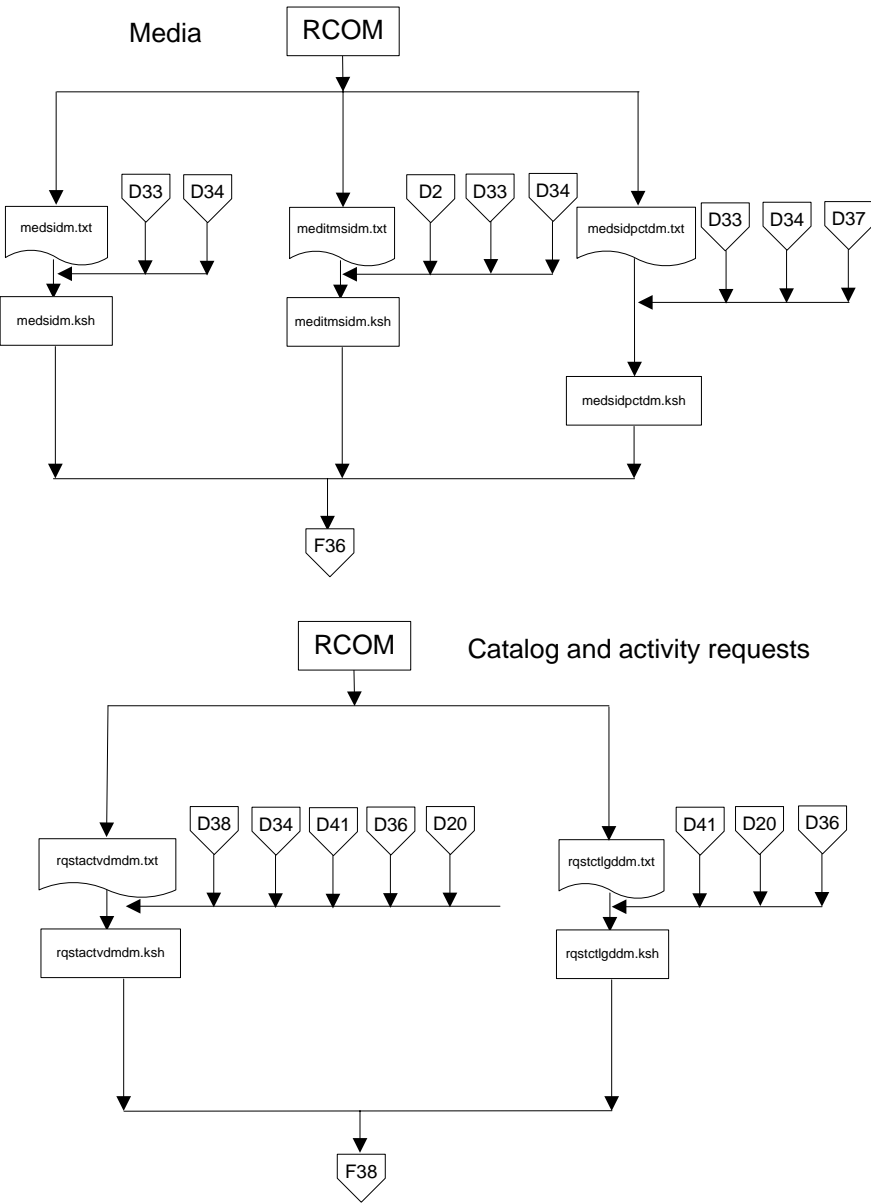
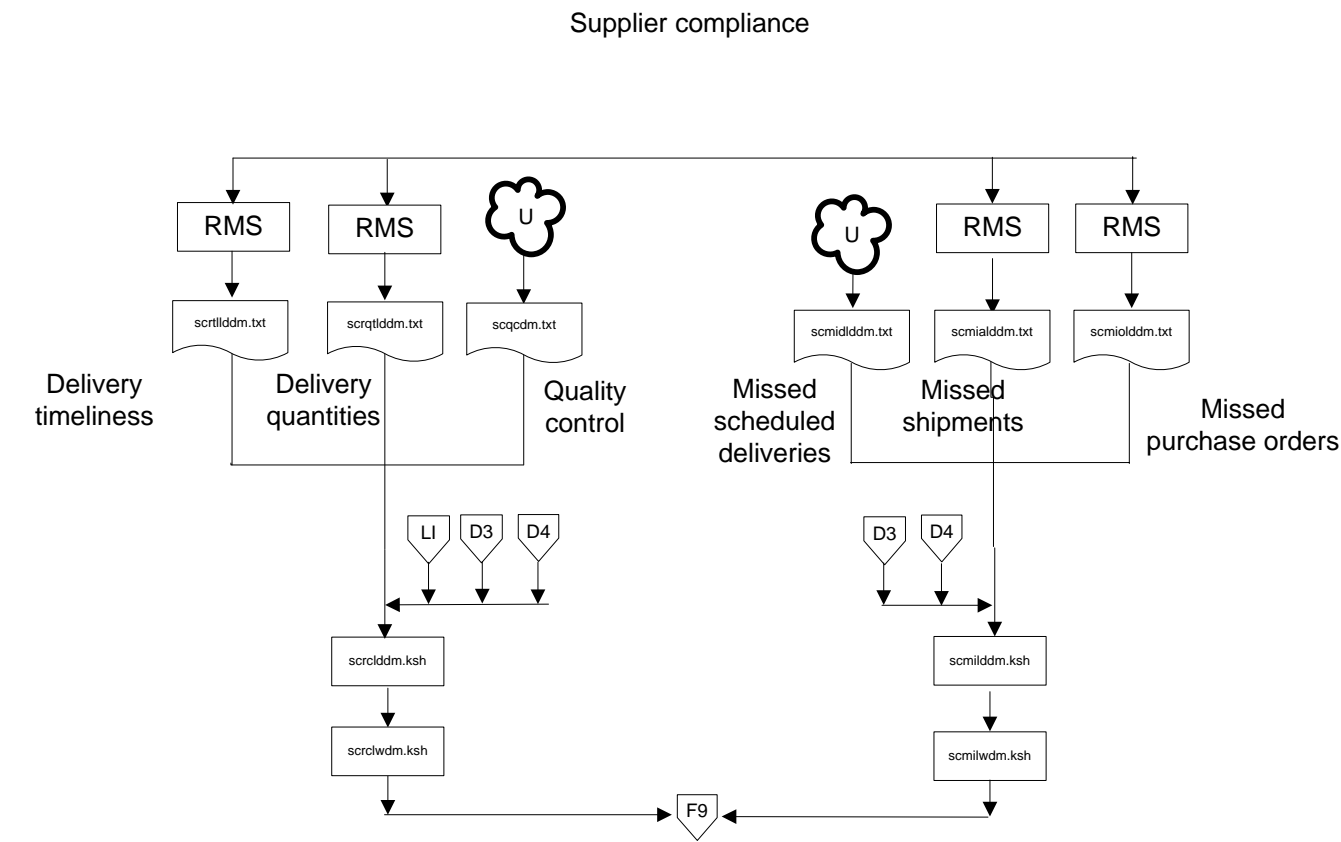
Note:
Dotted line indicates that either spallddm.ksh or spallddm2.ksh runs, never both.

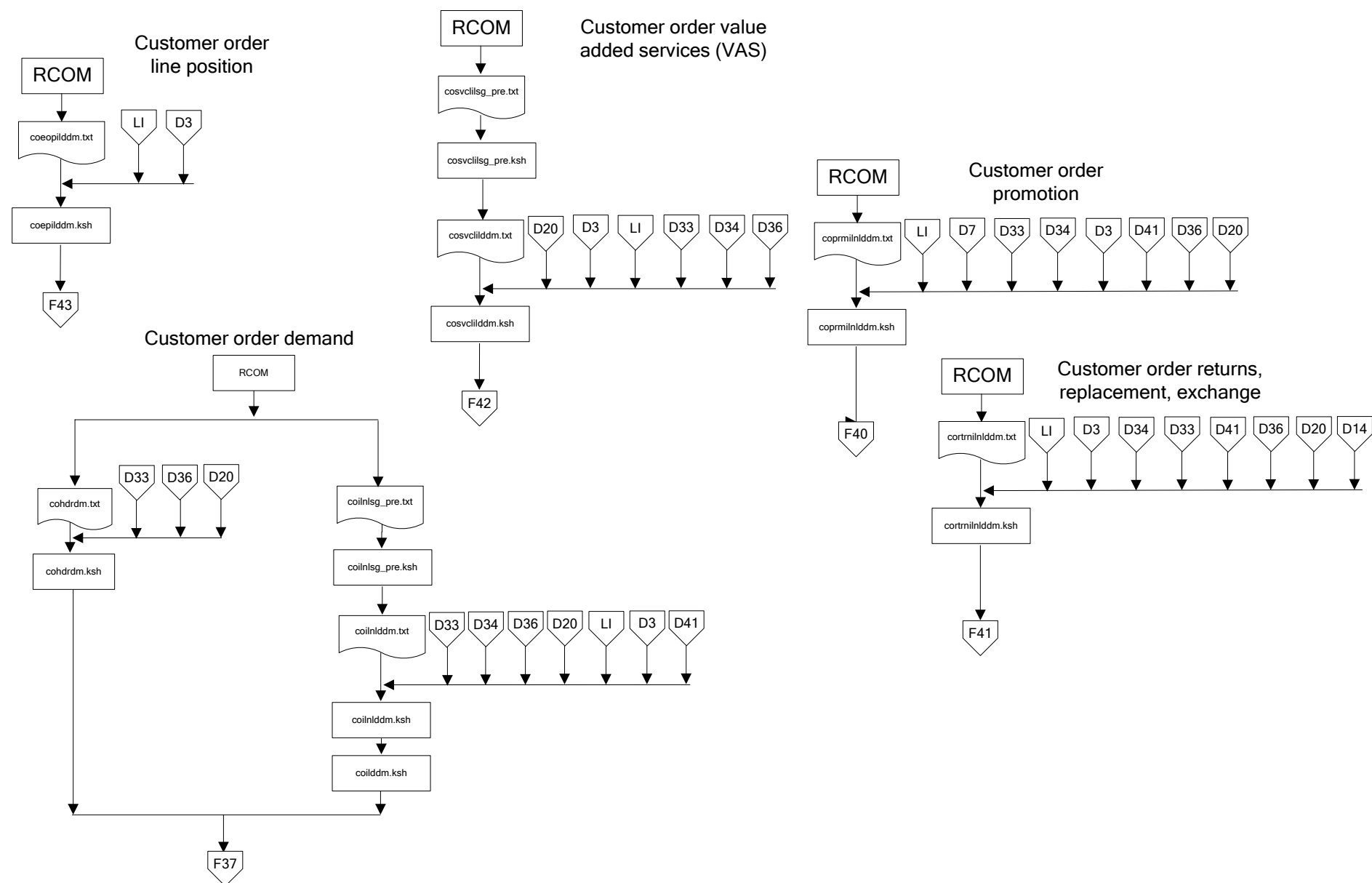


Fact dataflows



Fact dataflows





Chapter 8 – Program reference lists

This chapter serves as a reference to the following RDW programs and reference information:

- Dimension load (RETL Korn shell scripts)
- Fact load (RETL Korn shell scripts)
- Maintenance (RETL Korn shell scripts)
- The PROGRAM_CONTROL_DM values listed in the tables that follow are explained in a table at the end of this chapter.

By reviewing “Chapter 7 – Program flow diagrams”, along with this chapter and “Appendix A – Application Programming Interface (API) flat file specifications”, the client should be able to track, down to the table and column level, all the fact and dimension data that flows into RDW.

Dimension programs

When referencing the tables below, note the following:

The dimension DM KSH programs do not have an “argument” column in the following table because these programs do not require a path/file_name parameter. Dimension programs assume source text files will be located in \${MMHOME}/data and named <DM KSH program name>.txt. If clients wish to change this default path, they will need to pass in their own path/file_name at the command line.

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control_DM.program_type	Program_Control_DM.operation_type	Notes
cdedtlcomdm.ksh	Customer Order Management Codes	Dimension Load	RCOM	cdedtlcomdm.txt	cdedtlcomdm.schema	CDE_DTL_COM_DM	DIM_TOP	UPDATE	This program is a daily dimension snapshot program that processes inserts/updates to dynamic codes that originate in RCOM (such as Hold Events, Value Added Service Colors, and so on). There is also an install script, load_cde_dtl_com_dm.sql, that loads static RCOM codes (such as Customer Order Line Types, Customer Order Return Statuses, and so on) to the CDE_DTL_COM_DM table. All static codes have cde_keys up to 99, whereas the dynamic codes all have keys greater than 99.
cdedtldm.ksh	Codes	Dimension Load	RMS	cdedtldm.txt	cdedtldm.schema	CDE_DTL_DM	DIM_STANDALONE	UPDATE	There is an installation script to load this table initially. However, UOM codes will not be updated after the initial loading script. All other codes can be updated/added into the CDE_DTL_DM table daily using the cdedtldm.ksh program.

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control_DM.program_type	Program_Control_DM.operation_type	Notes
cllctrdm.ksh	Customer Order Dimension	Dimension Load	RCOM	cllctrdm.txt	cllctrdm.schema	CALL_CTR_DM	DIM_TOP	UPDATE	
cmptrdm.ksh	Competitor Dimension	Dimension Load	RMS	cmptrdm.txt	cmptrdm.schema	CMPTR_DM	DIM_TOP	UPDATE	
cmptrlmdm.ksh	Competitor Dimension	Dimension Load	RMS	cmptrlmdm.txt	cmptrlmdm.schema	CMPTR_LOC_MTX_DM	DIM_MTX	INSERT	
cmptrlocdm.ksh	Competitor Dimension	Dimension Load	RMS	cmptrlocdm.txt	cmptrlocdm.schema	CMPTR_LOC_DM	DIM_LOW	UPDATE	
coshptodm.ksh	Customer Order Dimension	Dimension Load	RCOM	coshptodm.txt	coshptodm.schema	CO_SHIP_TO_DM	DIM_TOP	UPDATE	
crncyddm.ksh	Currency Code Dimension	Dimension Load	RMS	crncyddm.txt	crncyddm.schema	CRNCY_CDE_DM	DIM_TOP	UPDATE	
crrdm.ksh	Customer Order Dimension	Dimension Load	RCOM	crrdm.txt	crrdm.schema	CARRIER_DM	DIM_TOP	UPDATE	
crrsvcdm.ksh	Customer Order Dimension	Dimension Load	RCOM	crrsvcdm.txt	crrsvcdm.schema	CARRIER_SVC_DM	DIM_TOP	UPDATE	
csrdm.ksh	Customer Order Dimension	Dimension Load	RCOM	csrdm.txt	csrdm.schema	CSR_DM	DIM_LOW	UPDATE	
custacctndm.ksh	Customer Account Dimension	Dimension Load	See notes	custacctndm.txt	custacctndm.schema	CUST_ACCNT_DM	DIM_TOP_DELTA_IDNT	UPDATE_D	Source file supplied by client. Data in this table helps tie the POS sales to the customer keys by associating those sales to a customer account (such as a loyalty account).

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control_DM.program_type	Program_Control_DM.operation_type	Notes
custaccttypedm.ksh	Customer Account Dimension	Dimension Load	See notes	custaccttypedm.txt	custaccttypedm.schema	CUST_ACCNT_TYPE_DM	DIM_MTX	INSERT	Source file supplied by client.
custclstrdm.ksh	Customer and Product Clustering Dimension	Dimension Load	See notes	custclstrdm.txt	custclstrdm.schema	CUST_CLSTR_DM	DIM_STANDALONE	UPDATE	Source file supplied by client.
custclstrimdm.ksh	Customer and Product Clustering Dimension	Dimension Load	See notes	custclstrimdm.txt	custclstrimdm.schema	CUST_CLSTR_ITEM_MTX_DM	DIM_MTX	INSERT	Source file supplied by client.
custclupddm.ksh	Customer and Product Clustering Dimension	Dimension Load	See notes	custclstrdm.txt	custclstrdm.schema	CUST_DM	DIM_STANDALONE	UPDATE	Source file supplied by client. Note that the schema and text files are different from the program name.
custdemogdm.ksh	Customer and Customer Demographics Dimension	Dimension Load		CUST_DM		CUST_MARITAL_DM, CUST_GENDER_DM, CUST_ETHNIC_DM, CUST_DT_OF_BIRTH_DM, CUST_INCOME_DM, CUST_CHILD_DM, CUST_HH_DM	DIM_STANDALONE_TABLE	UPDATE	

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control_DM.program_type	Program_Control_DM.operation_type	Notes
custdm.ksh	Customer and Customer Demographics Dimension	Dimension Load	RCOM	custdm.txt	custdm.schema	CUST_DM	DIM_TOP_DELTA	UPDATE	Source file supplied by RCOM or by the client.
dpctdm.ksh	Depiction Code Dimension	Dimension Load	RCOM	dpctdm.txt	dpctdm.schema	DPCT_DM	DIM_TOP	UPDATE	
emplydm.ksh	Employee Dimension	Dimension Load	RMS	emplydm.txt	emplydm.schema	EMPTY_DM	DIM_TOP	UPDATE	
geocdedm.ksh	Customer Geographic Dimension	Dimension Load	See notes	geocdedm.txt	geocdedm.schema	GEO_CDE_DM	DIM_TOP	UPDATE	Source file supplied by client.
itmclstrcmdm.ksh	Customer and Product Clustering Dimension	Dimension Load	See notes	itmclstrcmdm.txt	itmclstrcmdm.schema	ITEM_CLSTR_CUST_MTX_DM	DIM_MTX	INSERT	Source file supplied by client.
itmlkupdm.ksh	Product Dimension	Dimension Lookup		PROD_ITEM_DM, PROD_ITEM_SEASN_MTX_DM		ITEM_KEY_LKUP_TEMP	DIM_STANDALONE_TABLE	UPDATE	Builds item lookup temp table daily to aid performance of fact loads.
maralmdm.ksh	Market Data Dimension	Dimension Load	See notes	maralmdm.txt	maralmdm.schema	MKT_AREA_LOC_MTX_DM	DIM_MTX	INSERT	Source file supplied by client.
maralvldm.ksh	Market Data Dimension	Dimension Load	See notes	maralvldm.txt	maralvldm.schema	MKT_AREA_LEVEL1_DM, MKT_AREA_LEVEL2_DM, MKT_AREA_LEVEL3_DM	DIM_STANDALONE	UPDATE	Source file supplied by client.

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control_DM.program_type	Program_Control_DM.operation_type	Notes
mdepdm.ksh	Market Data Dimension	Dimension Load	See notes	mdepdm.txt	mdepdm.schema	MKT_PROD_DEPT_DM, MKT_PROD_DEPT_MTX_DM	DIM_STANDALONE	UPDATE	Source file supplied by client.
meddm.ksh	Media	Dimension Load	RCOM	meddm.txt	meddm.schema	MEDIA_DM	DIM_TOP	UPDATE	Source file supplied by RCOM. Although the MEDIA_DM table contains some 'fact' columns, this data is processed as a dimension. The media_dm_v view is built from the MEDIA_DM table and represents the dimensional lookup for related fact tables.
medlmdm.ksh	Media	Dimension Load	RCOM	meddm.txt	meddm.schema	MEDIA_LOC_MTX_DM	DIM_STANDALONE	UPDATE	Note that medlmdm.ksh uses meddm.txt and meddm.schema.
mitmdm.ksh	Market Data Dimension	Dimension Load	See notes	mitmdm.txt	mitmdm.schema	MKT_PROD_ITEM_DM, MKT_PROD_ITEM_MTX_DM	DIM_STANDALONE	UPDATE	Source file supplied by client.
orgaradm.ksh	Organization Dimension	Dimension Load	RMS	orgaradm.txt	orgaradm.schema	ORG_AREA_DM	DIM_LOW	UPDATE	
orgchandm.ksh	Organization Dimension	Dimension Load	RMS	orgchandm.txt	orgchandm.schema	ORG_CHANNEL_DM	DIM_STANDALONE	UPDATE	

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control_DM.program_type	Program_Control_DM.operation_type	Notes
orgchndm.ksh	Organization Dimension	Dimension Load	RMS	orgchndm.txt	orgchndm.schema	ORG_CHAIN_DM	DIM_LOW	UPDATE	
orgdisdm.ksh	Organization Dimension	Dimension Load	RMS	orgdisdm.txt	orgdisdm.schema	ORG_DISTT_DM	DIM_LOW	UPDATE	
orgllmdm.ksh	Organization Dimension	Dimension Load	RMS	orgllmdm.txt	orgllmdm.schema	ORG_LOCLST_MTX_DM	DIM_MTX	INSERT	
orglocdm.ksh	Organization Dimension	Dimension Load	RMS	orglocdm.txt	orglocdm.schema	ORG_LOC_DM	DIM_LOW	UPDATE	
orgloldm.ksh	Organization Dimension	Dimension Load	RMS	orgloldm.txt	orgloldm.schema	ORG_LOCLST_DM	DIM_TOP_F	UPDATE_D	
orgltmdm.ksh	Organization Dimension	Dimension Load	RMS	orgltmdm.txt	orgltmdm.schema	ORG_LOC_TRAIT_MTX_DM	DIM_MTX	INSERT	
orgltrdm.ksh	Organization Dimension	Dimension Load	RMS	orgltrdm.txt	orgltrdm.schema	ORG_LOC_TRAIT_DM	DIM_TOP_IDNT	UPDATE	
orglwmdm.ksh	Organization Dimension	Dimension Load	See notes	orglwmdm.txt	orglwmdm.schema	ORG_LOC_WK_MTX_DM	DIM_STANDALONE	UPDATE	Processes comparable location data at week. Source file supplied by client.
orgrgndm.ksh	Organization Dimension	Dimension Load	RMS	orgrgndm.txt	orgrgndm.schema	ORG_REGN_DM	DIM_LOW	UPDATE	
phasdm.ksh	Product Season Dimension	Dimension Load	RMS	phasdm.txt	phasdm.schema	PHASE_DM	DIM_LOW	UPDATE	

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control_DM.program_type	Program_Control_DM.operation_type	Notes
plnsendm.ksh	Plan Season Dimension	Dimension Load	See notes	plnsendm.txt	plnsendm.schema	PLN_SEASN_DM, TIME_PLN_STD_BY_WK_DM, PLN_SEASN_WK_MTX_DM	DIM_TOP_F	UPDATE_DL	Source file supplied by client.
prclsdm.ksh	Product Dimension	Dimension Load	RMS	prclsdm.txt	prclsdm.schema	PROD_CLASS_DM	DIM_LOW	UPDATE	
prdcmpdm.ksh	Company Dimension	Dimension Load	RMS	prdcmpdm.txt	prdcmpdm.schema	CMPY_DM	DIM_TOP	UPDATE_L	
prdepsdm.ksh	Product Dimension	Dimension Load	RMS	prdepsdm.txt	prdepsdm.schema	PROD_DEPT_DM	DIM_LOW	UPDATE	
prddiffdm.ksh	Product Dimension	Dimension Load	RMS	prddiffdm.txt	prddiffdm.schema	PROD_DIFF_DM	DIM_MTX	UPDATE	
prddivdm.ksh	Product Dimension	Dimension Load	RMS	prddivdm.txt	prddivdm.schema	PROD_DIV_DM	DIM_LOW	UPDATE	

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control_DM.program_type	Program_Control_DM.operation_type	Notes
prddtypdm.ksh	Product Dimension	Dimension Load	RMS	prddtypdm.txt	prddtypdm.schema	PROD_DIFF_TYPE_DM	DIM_STANDALONE	UPDATE	<p>1. No more than 30 DIFF types can exist between the text file and RDW. See Appendix A, “Application programming interface (API) flat file specifications”, for more information.</p> <p>2. For more information about DIFF type processing and RDW’s front end, see the RDW Middle Tier Installation Guide.</p>
prdgrpdm.ksh	Product Dimension	Dimension Load	RMS	prdgrpdm.txt	prdgrpdm.schema	PROD_GRP_DM	DIM_LOW	UPDATE	
prdisldm.ksh	Item-Supplier-Location Cross Dimension	Dimension Load	RMS	prdisldm.txt	prdisldm.schema	PROD_ITEM_SUPP_LOC_DM	DIM_MTX	INSERT	
prditmclstrdm.ksh	Customer and Product Clustering Dimension	Dimension Load	See notes	prditmclstrdm.txt	prditmclstrdm.schema	PROD_ITEM_CLSTR_DM	DIM_STANDALONE	UPDATE	Source file supplied by client.
prditmclupddm.ksh	Customer and Product Clustering Dimension	Dimension Load	See notes	prditmclstrdm.txt	prditmclstrdm.schema	PROD_ITEM_DM	DIM_STANDALONE	UPDATE	Source file supplied by client. Note that the text and the schema file names are different from the program name.

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control_DM.program_type	Program_Control_DM.operation_type	Notes
prditmdm.ksh	Product Dimension	Dimension Load	RMS	prditmdm.txt	prditmdm.schema	PROD_ITEM_DM	DIM_LOW	UPDATE_L	
prditlmdm.ksh	Product Dimension	Dimension Load	RMS	prditlmdm.txt	prditlmdm.schema	PROD_ITEMLST_DM	DIM_TOP_F	UPDATE_D	
prditmlmdm.ksh	Product Dimension	Dimension Load	RMS	prditmlmdm.txt	prditmlmdm.schema	PROD_ITEMLST_MTX_DM	DIM_MTX	INSERT	
prditmltmdm.ksh	Item-Location Trait Cross Dimension	Dimension Load	RMS	prditmltmdm.txt	prditmltmdm.schema	PROD_ITEM_LOC_TRAITS_MTX_DM	DIM_MTX	INSERT	
prditmsmdm.ksh	Product Dimension	Dimension Load	RMS	prditmsmdm.txt	prditmsmdm.schema	PROD_SEASN_ITEM_MTX_DM	DIM_STANDALONE	UPDATE	
prditmuddm.ksh	Product Dimension	Dimension Load	RMS	prditmuddm.txt	prditmuddm.schema	PROD_ITEM_UDA_DTL_DM	DIM_TOP_F	UPDATE_DL	
prditmuhdm.ksh	Product Dimension	Dimension Load	RMS	prditmuhdm.txt	prditmuhdm.schema	PROD_ITEM_UDA_HEAD_DM	DIM_TOP_F	UPDATE_D	
prditmumdm.ksh	Product Dimension	Dimension Load	RMS	prditmumdm.txt	prditmumdm.schema	PROD_ITEM_UDA_MTX_DM	DIM_STANDALONE	UPDATE	
prdlvldm.ksh	Product Dimension	Dimension Load	RMS	PROD_ITEM_DM		PROD_LEVEL1_DM, PROD_LEVEL2_DM, PROD_LEVEL3_DM	DIM_STANDALONE - TABLE	UPDATE	

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control_DM.program_type	Program_Control_DM.operation_type	Notes
prdpimdm.ksh	Product Dimension	Dimension Load	RMS	prdpimdm.txt	prdpimdm.schema	PROD_PACK_ITEM_MTX_DM	DIM_STANDALONE	UPDATE	
prdsbcdm.ksh	Product Dimension	Dimension Load	RMS	prdsbcdm.txt	prdsbcdm.schema	PROD_SBC_DM	DIM_LOW	UPDATE	
prmdtldm.ksh	Promotion Dimension	Dimension Load	RPM	prmdtldm.txt	prmdtldm.schema	PRMTN_DTL_DM	DIM_LOW	UPDATE	
prmevtdm.ksh	Promotion Dimension	Dimension Load	RPM	prmevtdm.txt	prmevtdm.schema	PRMTN_EVENT_DM	DIM_TOP	UPDATE	
prmhdrdm.ksh	Promotion Dimension	Dimension Load	RPM	prmhdrdm.txt	prmhdrdm.schema	PRMTN_HEAD_DM	DIM_LOW	UPDATE	

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control_DM.program_type	Program_Control_DM.operation_type	Notes
prmtndm_pre.ksh	Promotion Dimension	Dimension Pre-load	See notes	prmdtldm_dtc.txt prmdtldm.txt prmhdrdm.txt	prmdtldm.schema prmhdrdm.schema	prmdtldm.txt prmhdrdm.txt	DIM_STANDALONE	UPDATE	This program allows a retailer with RMS and RCOM (DTC) promotions to combine that dimensional data for loading to RDW. The program combines RMS promotion head, promotion detail, and RCOM promotion detail together and generates four output files: prmhdrdm.txt, prmdtldm.txt, prmhdrdm_rpm.txt and prmdtldm_rpm.txt. Text files prmhdrdm.txt and prmdtldm.txt are used for the RDW dimension head and detail load programs (prmhdrdm.ksh and prmdtldm.ksh). Text files prmhdrdm_rpm.txt and prmdtldm_rpm.txt are used internally by this program only. The text files prmhdrdm.ksh and prmdtldm.ksh are kept after the program runs to allow the client to rerun the program if necessary. A stand-alone client, which provides RDW promotion dimension with only one source system, can skip this program and run prmhdrdm.ksh and prmdtldm.ksh directly by using source files prmhdrdm.txt and prmdtldm.txt.
regngrpdm.ksh	Regionality Dimension	Dimension Load	RMS	regngrpdm.txt	regngrpdm.schema	REGIONALITY_GRP_DM	DIM_TOP_IDNT	UPDATE_D	
regnmtxdm.ksh	Regionality Dimension	Dimension Load	RMS	regngrpdm.txt	regnmtxdm.schema	REGIONALITY_MTX_DM	DIM_MTX	INSERT	
rgstrdm.ksh	Register Dimension	Dimension Load	RMS	ttldmdm.txt, rgstrdm.txt	ttldmdm.schema, rgstrdm.schema	RGSTR_DM	DIM_TOP	INSERT	

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control_DM.program_type	Program_Control_DM.operation_type	Notes
rsndm.ksh	Reason Dimension	Dimension Load	RMS	rsndm.txt	rsndm.schema	REASN_DM	DIM_TOP	UPDATE	
seasndm.ksh	Product Season Dimension	Dimension Load	RMS	seasndm.txt	seasndm.schema	SEASN_DM, TIME_STD_BY_DAY_DM, TIME_STD_BY_WK_DM	DIM_TOP	UPDATE_L	
selitmdm.ksh	Media	Dimension Load	RCOM	selitmdm.txt	selitmdm.schema	SELLING_ITEM_DM	DIM_TOP	UPDATE	
subtrantypedm.ksh	Sub-Transaction Type Dimension	Dimension Load	RMS	subtrantypedm.txt	subtrantypedm.schema	SUB_TRAN_TYPE_DM	DIM_TOP	UPDATE	
supctrdm.ksh	Supplier Dimension	Dimension Load	RMS	supctrdm.txt	supctrdm.schema	SUPP_CNTRCT_DM	DIM_LOW	UPDATE	
supsupdm.ksh	Supplier Dimension	Dimension Load	RMS	supsupdm.txt	supsupdm.schema	SUPP_DM	DIM_TOP	UPDATE_L	
suptrmdm.ksh	Supplier Dimension	Dimension Load	RMS	suptrmdm.txt	suptrmdm.schema	SUPP_TRAIT_MTX_DM	DIM_MTX	INSERT	
suptrtdm.ksh	Supplier Dimension	Dimension Load	RMS	suptrtdm.txt	suptrtdm.schema	SUPP_TRAIT_DM	DIM_TOP_IDNT	UPDATE	
tndrtypedm.ksh	Tender Type Dimension	Dimension Load	RMS	tndrtypedm.txt	tndrtypedm.schema	TNDR_TYPE_DM	DIM_TOP	UPDATE	
ttltypdm.ksh	ReSA Total Type Dimension	Dimension Load	RMS	ttltypdm.txt	ttltypdm.schema	TOTAL_TYPE_DM	DIM_TOP	UPDATE	

Fact programs

When referencing the tables below, note the following:

- All aggregation programs will derive data from temporary table *_TEMP that is created by the program that loads lowest-level facts from a source system for the fact datamart. For example, slsmkdnilddm.ksh is at item-location-day level. For markdowns, day is the lowest level for time, and week is the next level for time. The lowest level (or base) fact load program, slsmkdnilddm.ksh, needs to create a temp table for the next level aggregation. This temp table will hold today's changes/new facts and will be used by slsmkdnilddm.ksh to aggregate today's changes to the target week table. The “Source Table or File” column for fact aggregation programs, therefore, is left blank in the program reference list.
- The “Arguments” column lists all the command line parameters that exist in addition to the program name itself.
- For the base fact DM Kornshell programs below, the data file path/file_name is a required command line parameter. The “Arguments” column contains the RDW default data file directory path and file name, such as \${MMHOME}/data/cmptrcilddm.txt. If clients wish to change this default path, they will need to substitute their own path/file_name at the command line.

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control DM.program_type	Program_Control DM.operation_type	Argument s	Notes
cmptprcilddm.ksh	Competitor Pricing	Base Fact with compressed table	RMS	cmptprcilddm.txt	cmptprcilddm.schema	COMP_PRICING_ITEM_LD_DM, CMPTR_PRICING_IL_CUR_DM	BASEFACT_UPD	UPDATE_L	\${MMHOME}/data/cmptprcilddm.txt	See Chapter 4 for compressed table and cur table. This program allows backposted data to process to compressed target table.
coeopilddm.ksh	Customer Order Line Position	Base Fact	RCOM	coeopilddm.txt	coeopilddm.schema	CO_EOP_ITEM_LD_DM	BASEFACT_INS	INSERT_G	\${MMHOME}/data/coeopilddm.txt	

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control DM.program_type	Program_Control DM.operation_type	Argument s	Notes
cohdrdm.ksh	Customer Order	Base Fact	RCOM	cohdrdm.txt	cohdrdm.schema	CO_HDR_DM	BASEFACT_UPD	UPDATE_A	\${MMHOME}/data/cohdrdm.txt	
coilddm.ksh	Customer Order	Aggregation				CO_ITEM_LD_DM	FACT_STANDALONE	UPDATE		This program aggregates customer order line transactions for the current day.
coilnlddm.ksh	Customer Order	Base Fact		coilnlddm.txt	coilnlddm.schema	CO_ITEM_LINE_LD_SG, CO_ITEM_LINE_LD_DM	BASEFACT_UPD	UPDATE_L	\${MMHOME}/data/coilnlddm.txt	If the custom order line is not completed today, records will be loaded into the CO_ITEM_LINE_LD_SG table. If the custom order is completed (cancelled or shipped) today, the completed records are removed from the CO_ITEM_LINE_LD_SG table and loaded into the CO_ITEM_LINE_LD_DM table.

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control DM.program_type	Program_Control DM.operation_type	Argument s	Notes
coilnls_g_pre.ksh	Customer Order	Base Fact Tranformation	RCOM	coilnls_g_pre.txt	coilnls_g_pre.schema	CO_ITEM_LL_CUR_SG, coilnlddm.txt	FACT_STANDALONE	UPDATE	\${MMHOME}/data/coilnls_g_pre.txt \${MMHOME}/data/coilnlddm.txt	This program processes the incoming file coilnls_g_pre.txt. The results are written into an output file, coilnlddm.txt. The current day's open CO line record will also be inserted/updated into the table, CO_ITEM_LL_CUR_SG
coprmilnlddm.ksh	DTC Promotion	Base Fact	RCOM	coprmilnlddm.txt	coprmilnlddm.schema	CO_PRMTN_ITEM_LLD_DM	BASEFACT_UPD	UPDATE	\${MMHOME}/data/coprmilnlddm.txt	
cortnilnlddm.ksh	DTC Returns	Base Fact	RCOM	cortnilnlddm.txt	cortnilnlddm.schema	CO_RTRN_ITEM_LLD_DM	BASEFACT_UPD	UPDATE_A	\${MMHOME}/data/cortnilnlddm.txt	

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control DM.program_type	Program_Control DM.operation_type	Argument s	Notes
cosvclilsg_pre.ksh	Value Added Services	Base Fact Transformation	RCOM	cosvclilsg_pre.txt	cosvclilsg_pre.schema	CO_SVC_ITEM_LL_CUR_SG, cosvclilddm.txt	FACT_STANDALONE	UPDATE	\${MMHOME}/data/cosvclilsg_pre.txt \${MMHOME}/data/cosvclilddm.txt	This program processes the incoming file cosvclilsgpre.txt. The results are written into an output file, cosvclilddm.txt. The current day's CO service line record will also be inserted/update into the table CO_SVC_ITEM_LL_CUR_SG.
cosvclilddm.ksh	Value Added Services	Base Fact		cosvclilddm.txt	cosvclilddm.schema	CO_SVC_ITEM_LLD_DM	BASEFACT_INS	INSERT	\${MMHOME}/data/cosvclilddm.txt	
cstislddm.ksh	Cost	Base Fact with compressed table	RMS	cstislddm.txt	cstislddm.schema	COST_ITEM_SUPP_LD_DM, COST_ISL_CUR_DM	BASEFACT_UPD	UPDATE_L	\${MMHOME}/data/cstislddm.txt	See Chapter 4 for compressed table and cur table.
exchngratedm.ksh	Exchange Rates	Base Fact with insert	RMS	exchngratedm.txt	exchngratedm.schema	EXCHNG_RATE_CRNCY_DAY_DM	BASEFACT_INS	INSERT	\${MMHOME}/data/exchngratedm.txt	Compressed program without cur table.
invblddm.ksh	Inventory Position	Positional Aggregation				INV_SBC_LD_DM	FACT_AGG_POS	UPDATE		
invblwdm.ksh	Inventory Position	Positional Aggregation				INV_SBC_LW_DM	FACT_AGG_POS	UPDATE_F		

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control DM.program_type	Program_Control DM.operation_type	Argument	Notes
invlddm.ksh	Inventory Position	Positional Aggregation				INV_DEPT_LD_DM	FACT_AGG_POS	UPDATE_G		
invlwdm.ksh	Inventory Position	Positional Aggregation				INV_DEPT_LW_DM	FACT_AGG_POS	UPDATE_F		
invilddm.ksh	Inventory Position	Base Fact with compressed table	RMS	invilddm.txt	invilddm.schema	INV_ITEM_LD_DM, INV_IL_CUR_DM	BASEFACT_UPD	UPDATE_L	\${MMHOME}/data/invilddm.txt	See Chapter 4 for compressed table and cur table. Inv position cannot be back posted.
invilwdm.ksh	Inventory Position	Positional Aggregation				INV_ITEM_LW_DM	FACT_AGG_POS	INSERT		
ivailddm.ksh	Inventory Adjustment	Base Fact with incremental update	RMS	ivailddm.txt	ivailddm.schema	INV_ADJ_ITEM_LD_DM	BASEFACT_INCR_UPD	UPDATE	\${MMHOME}/data/ivailddm.txt	
ivrcplddm.ksh	Inventory Receipts	Aggregation				INV_RCPTS_SBC_LD_DM	FACT_AGG_STD	UPDATE_S		
ivrcplwdm.ksh	Inventory Receipts	Aggregation				INV_RCPTS_SBC_LW_DM	FACT_AGG_STD	UPDATE_F		
ivrcpilddm.ksh	Inventory Receipts	Base Fact with incremental update	RMS	ivrcpilddm.txt	ivrcpilddm.schema	INV_RCPTS_ITEM_LD_DM	BASEFACT_INCR_UPD	UPDATE_G	\${MMHOME}/data/ivrcpilddm.txt	
ivrcpilwdm.ksh	Inventory Receipts	Aggregation				INV_RCPTS_ITEM_LW_DM	FACT_AGG_STD	UPDATE_FS		

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control DM.program_type	Program_Control DM.operation_type	Argument s	Notes
ivrlddm.ksh	Return to Vendor	Base Fact with update	RMS	ivrlddm.txt	ivrlddm.schema	INV_RTV_SUPP_ITEM_LD_DM	BASEFACT_UPD	UPDATE	\${MMHOME}/data/ivrlddm.txt	
ivtblddm.ksh	Inventory Transfers	Aggregation				INV_TSF_SBC_LD_DM	FACT_AGG_STD	UPDATE_S		
ivtblwdm.ksh	Inventory Transfers	Aggregation				INV_TSF_SBC_LW_DM	FACT_AGG_STD	UPDATE_F		
ivtilddm.ksh	Inventory Transfers	Base fact with incremental update	RMS	ivtilddm.txt	ivtilddm.schema	INV_TSF_ITEM_LD_DM	BASEFACT_INCR_UPD	UPDATE_A	\${MMHOME}/data/ivtilddm.txt	
ivtilwdm.ksh	Inventory Transfers	Aggregation				INV_TSF_ITEM_LW_DM	FACT_AGG_STD	UPDATE_FS		
ivuilddm.ksh	Unavailable Inventory	Base Fact with update, for compressed table	RMS	ivuilddm.txt	ivuilddm.schema	INV_UNAVL_ITEM_LD_DM, INV_UNAVL_IL_CUR_DM	BASEFACT_UPD	UPDATE_L	\${MMHOME}/data/ivuilddm.txt	See Chapter 4 for compressed table and cur table.
lptldmdm.ksh	Loss Prevention Transactions(voids, no sales)	Base Fact with incremental update	RMS	lptldmdm.txt	lptldmdm.schema	LP_TRAN_LM_DM	BASEFACT_INCR_UPD	UPDATE_G	\${MMHOME}/data/lptldmdm.txt	
lptldqdm.ksh	Loss Prevention Transactions(voids, no sales)	Aggregation				LP_TRAN_LQ_DM	FACT_AGG_STD	UPDATE_FS		
lptotclddm.ksh	Loss Prevention Totals (cashier over or short)	Base Fact with incremental update	RMS	lptotclddm.txt	lptotclddm.schema	LP_TOT_CSHR_LD_DM	BASEFACT_INCR_UPD	UPDATE_G	\${MMHOME}/data/lptotclddm.txt	

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control DM.program_type	Program_Control DM.operation_type	Argument s	Notes
lptotlddm.ksh	Loss Prevention Totals (user defined totals)	Base Fact with incremental update	RMS	lptotlddm.txt	lptotlddm.schema	LP_TOT_LD_DM	BASEFACT_INCR_UPD	UPDATE_G	\${MMHOME}/data/lptotlddm.txt	
meditmsidm.ksh	Media (media/selling item/item)	Base Fact with update	RCOM	meditmsidm.txt	meditmsidm.schema	MEDIA_ITEM_SI_DM	BASEFACT_UPD	UPDATE	\${MMHOME}/data/meditmsidm.txt	This program is processed as a fact even though there is no time element associated with the facts.
medsidm.ksh	Media (media/selling item)	Base Fact with update	RCOM	medsidm.txt	medsidm.schema	MEDIA_SI_DM	BASEFACT_UPD	UPDATE	\${MMHOME}/data/medsidm.txt	This program is processed as a fact even though there is no time element associated with the facts.
medsidpctdm.ksh	Media (media/selling item/depiction code)	Base Fact with update	RCOM	medsidpctdm.txt	medsidpctdm.schema	MEDIA_SI_DPCT_DM	BASEFACT_UPD	UPDATE	\${MMHOME}/data/medsidpctdm.txt	This program is processed as a fact even though there is no time element associated with the facts.
mslsdlwdm.ksh	Market Sales Data	Fact Standalone	See Notes	mslsdlwdm.txt, MKT_PROD_DEPT_DM, TIME_WK_DM, MKT_AREA_LEVEL1_DM, MKT_AREA_LEVEL2_DM, MKT_AREA_LEVEL3_DM	mslsdlwdm.schema	MKT_SLS_DEPT_LEVEL1_W_DM, MKT_SLS_DEPT_LEVEL2_W_DM, MKT_SLS_DEPT_LEVEL3_W_DM	FACT_STANDALONE	UPDATE	\${MMHOME}/data/mslsdlwdm.txt	This program is using the fact matrix concept. See Chapter 4 for more details. Source file supplied by client.

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control DM.program_type	Program_Control DM.operation_type	Argument s	Notes
mslsilwdm.ksh	Market Sales Data	Fact standalone	See notes	mslsilwdm.txt, MKT_PROD_ITEM_DM, TIME_WK_DM, MKT_AREA_LEVEL1_DM, MKT_AREA_LEVEL2_DM, MKT_AREA_LEVEL3_DM	mslsilwdm.schema	MKT_SLS_ITEM_LEVEL1_W_DM, MKT_SLS_ITEM_LEVEL2_W_DM, MKT_SLS_ITEM_LEVEL3_W_DM	FACT_STANDALONE	UPDATE	\${MMHOME}/data/mslsilwdm.txt	This program is using the fact matrix concept. See Chapter 4 for more details. Source file supplied by client.
ncstuilddm.ksh	Net Cost	Base Fact with update, for compressed table	RMS	ncstuilddm.txt	ncstuilddm.schema	NET_COST_SUPP_ITEM_LD_DM, NET_COST_SIL_CUR_DM	BASEFACT_UPD	UPDATE_L	\${MMHOME}/data/ncstuilddm.txt	See Chapter 4 for compressed table and cur table. This program does not allow backposted data.
nprftuilddm.ksh	Profit on Base Cost	Derivation, see notes		SLS_ITEM_LD_DM, NET_COST_SUPP_ITEM_LD_DM, PROD_ITEM_SUPP_LOC_DM, TIME_DAY_DM		NET_PRFT_SUPP_ITEM_LD_DM	FACT_STANDALONE	UPDATE		This program combines Merchandise Sales data and Net Cost data to provide alternate profit calculations. The facts are not the same as the profit facts on the Sales Transaction tables.
nprftuilwdm.ksh	Profit on Base Cost	Aggregation				NET_PRFT_SUPP_ITEM_LW_DM	FACT_AGG_DW	UPDATE_F		
plcblwdm.ksh	Planning	Base Fact with update	Merchandise Financial Planning	plcblwdm.txt	plcblwdm.schema	PLN_CURR_SBC_LW_DM	BASEFACT_UPD	UPDATE	\${MMHOME}/data/plcblwdm.txt	See Chapter 6, “RDW interfaces” for more information about the Merchandise Financial Planning interface.

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control DM.program_type	Program_Control DM.operation_type	Argument s	Notes
poblwdm.ksh	Planning	Base Fact with update	Merchandise Financial Planning	poblwdm.txt	poblwdm.schema	PLN_ORIG_SBC_LW_DM, PLN_CURR_SBC_LW_DM	BASEFACT_UPD	UPDATE_L	\${MMHOME}/data/poblwdm.txt	See Chapter 6, “RDW interfaces” for more information about the Merchandise Financial Planning interface.
prcilddm.ksh	Pricing	Base Fact with update, for compressed tables	RMS	prcilddm.txt	prcilddm.schema	PRICING_ITEM_LD_DM, PRICING_IL_CUR_DM	BASEFACT_UPD	UPDATE_L	\${MMHOME}/data/prcilddm.txt	See Chapter 4 for compressed table and cur table.
rplcilddm.ksh	Replacements	Base Fact with incremental update	RMS	rplcilddm.txt	rplcilddm.schema	RPLC_ITEM_LD_DM	BASEFACT_INCR_UPD	UPDATE	\${MMHOME}/data/rplcilddm.txt	
rqstactvdmdm.ksh	Customer Request	Base Fact with insert	RCOM	rqstactvdmdm.txt	rqstactvdmdm.schema	RQST_ACTV_MIN_DM	BASEFACT_INS	INSERT	\${MMHOME}/data/rqstactvdmdm.txt	
rqstctlgddm.ksh	Customer Request	Base Fact with insert	RCOM	rqstctlgddm.txt	rqstctlgddm.schema	RQST_CTLG_DAY_DM	BASEFACT_INS	INSERT	\${MMHOME}/data/rqstctlgddm.txt	
saviddm.ksh	Supplier Availability	Base Fact with update for compressed tables	RMS	saviddm.txt	saviddm.schema	SUPP_AVAIL_ITEM_DAY_DM, SUPP_AVAIL_I_CUR_DM	BASEFACT_UPD	UPDATE_L	\${MMHOME}/data/saviddm.txt	See Chapter 4 for compressed table and cur table.

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control DM.program_type	Program_Control DM.operation_type	Argument s	Notes
scmilddm.ksh	Supplier Compliance	Base Fact with insert	See notes	scmidlddm.txt , scmialddm.txt , scmiolddm.txt	scmidlddm.schema, scmialddm.schema, scmiolddm.schema	SCMP_RCPT_MISS_LD_DM	FACT_MATRIX	UPDATE_A	\${MMHOME}/data/scmidlddm.txt \${MMHOME}/data/scmialddm.txt \${MMHOME}/data/scmiolddm.txt	This program joins facts supplied in three text files. Missed shipments (scmialddm.txt) and missed purchase orders (scmiolddm.txt) are supplied by RMS. Missed scheduled deliveries (scmidlddm.txt) is supplied by the retailer.
scmilwdm.ksh	Supplier Compliance	Aggregation				SCMP_RCPT_MISS_LW_DM	FACT_AGG_STD	UPDATE_FS		
scrclddm.ksh	Supplier Compliance	Base Fact with insert	See notes	scrtllddm.txt, scrqtllddm.txt, scqcdm.txt	scrtllddm.schema, scrqtllddm.schema, scqcdm.schema	SCMP_RCPT_ITEM_LD_DM	FACT_STANDALONE	UPDATE	\${MMHOME}/data/scrtllddm.txt \${MMHOME}/data/scrqtllddm.txt \${MMHOME}/data/scqcdm.txt	This program joins facts supplied in three text files. Delivery timeliness (scrtllddm.txt) and delivery quantities (scrqtllddm.txt) are supplied by RMS. Quality control (scqcdm.txt) is supplied by the retailer.
scrclwdm.ksh	Supplier Compliance	Aggregation				SCMP_RCPT_ITEM_LW_DM	FACT_AGG_DW	UPDATE_FS		
sctiddm.ksh	Supplier Contract	Base Fact with update for compressed tables	RMS	sctiddm.txt	sctiddm.schema	SUPP_CNTRCT_ITEM_DAY_DM, SUPP_CNTRCT_I_CUR_DM	BASEFACT_UPD	UPDATE_L	\${MMHOME}/data/sctiddm.txt	See Chapter 4 for compressed table and cur table.

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control DM.program_type	Program_Control DM.operation_type	Argument s	Notes
sfcblwdm.ksh	Sales Forecasts	Aggregation		See notes on the top for aggregation.		SLS_FCST_SBC_LW_DM	FACT_AGG_POS	UPDATE_GF		This program runs weekly.
sfcilwdm.ksh	Sales Forecasts	Base Fact with update	RMS	sfcilwdm.txt	sfcilwdm.schema	SLS_FCST_ITEM_LW_DM	BASEFACT_UPD	UPDATE_A		This program runs weekly.
sincilddm.ksh	Supplier Invoice Cost	Base Fact with insert	ReIM	sincilddm.txt	sincilddm.schema	SUPP_INVC_COST_ITEM_LD_DM	BASEFACT_INS	INSERT	\${MMHOME}/data/sincilddm.txt	
slsblddm.ksh	Sales and Returns Transactions	Aggregation				SLS_SBC_LD_DM	FACT_AGG_STD	UPDATE		
slsblgmthdm.ksh	Sales and Returns Transactions	Aggregation				SLS_SBC_LGMTH_DM	FACT_AGG_STD	UPDATE_KO		This module should not be run and the table should be dropped if Gregorian time is not populated.
slsblwdm.ksh	Sales and Returns Transactions	Aggregation				SLS_SBC_LW_DM	FACT_AGG_STD	UPDATE_FS		
slsdlldm.ksh	Sales and Returns Transactions	Aggregation				SLS_DEPT_LD_DM	FACT_AGG_STD	UPDATE_S		
slsdlwdm.ksh	Sales and Returns Transactions	Aggregation				SLS_DEPT_LW_DM	FACT_AGG_STD	UPDATE_F		
slsilddm.ksh	Sales and Returns Transactions	Aggregation	RMS			SLS_ITEM_LD_DM	FACT_AGG_STD	UPDATE_S		

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control DM.program_type	Program_Control DM.operation_type	Argument s	Notes
slsildmdm.ksh	Sales and Returns Transactions	Base Fact with incremental update		slsildmdm.txt	slsildmdm.schema	SLS_ITEM_LM_DM	BASEFACT_INCR_UPD	UPDATE_G	\${MMHOME}/data/slsildmdm.txt	
slsilgmthdm.ksh	Sales and Returns Transactions	Aggregation				SLS_ITEM_LGMT_H_DM	FACT_AGG_STD	UPDATE_KO		This module should not be run and the table should be dropped if Gregorian time is not populated.
slsilwdm.ksh	Sales and Returns Transactions	Aggregation				SLS_ITEM_LW_DM	FACT_AGG_STD	UPDATE_F		
slsilmdlddm.ksh	Sales and Returns Transactions	Aggregation				SLS_ITEM_LMD_LD_DM	FACT_AGG_STD	UPDATE_S		
slsilmdlwdm.ksh	Sales and Returns Transactions	Aggregation				SLS_ITEM_LMD_LW_DM	FACT_AGG_STD	UPDATE_F		
slsmkdndlddm.ksh	Markdowns	Aggregation				SLS_MKDN_DEPT_LD_DM	FACT_AGG_STD	UPDATE_S		
slsmkdndlwdm.ksh	Markdowns	Aggregation				SLS_MKDN_DEPT_LW_DW	FACT_AGG_STD	UPDATE_F		
slsmkdnilddm.ksh	Markdowns	Base Fact with update	RMS	slsmkdnilddm.txt	slsmkdnilddm.schema	SLS_MKDN_ITEM_LD_DM	BASEFACT_INCR_UPD	UPDATE_G	\${MMHOME}/data/slsmkdnilddm.txt	
slsmkdnilwdm.ksh	Markdowns	Aggregation				SLS_MKDN_ITEM_LW_DM	FACT_AGG_STD	UPDATE_FS		

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control DM.program_type	Program_Control DM.operation_type	Argument s	Notes
slspilddm.ksh	Pack Sales	Derivation , see notes		SLS_ITEM_LD_DM , PRICING_ITEM_LD_DM, PROD_PACK_ITEM_MTX_DM, TIME_DAY_DM		SLS_PACK_ITEM_LD_DM	FACT_MATRIX	UPDATE		This program selects sales facts for pack items and breaks down the pack items into their component items.
slsprmilnlmdm.ksh	Promotion Sales	Base Fact with incremental update	RMS	slsprmilnlmdm.txt	slsprmilnlmdm.schema	SLS_PRMTN_ITEM_LLM_DM	BASEFACT_INCR_UPD	UPDATE_G	\${MMHOME}/data/slsprmilnlmdm.txt	
spaldlddm.ksh	Space Allocation	Base Fact with update, for compressed table	See Notes	spaldlddm.txt	spaldlddm.schema	SPACE_ALLOC_DEPT_LD_DM, SPACE_ALLOC_DL_CUR_DM	BASEFACT_UPD	UPDATE_L	\${MMHOME}/data/spaldlddm.txt	The client either runs spaldlddm.ksh or department space allocation program spaldlddm2.ksh. Source data is supplied by the client. This program will allow backposted data to process to the compressed target table.
spaldlddm2.ksh	Space Allocation	Aggregation				SPACE_ALLOC_DEPT_LD_DM, SPACE_ALLOC_DL_CUR_DM	FACT_STANDALONE	UPDATE		This program aggregates space allocation.

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control DM.program_type	Program_Control DM.operation_type	Argument s	Notes
spalilddm.ksh	Space Allocation	Base Fact with update, for compressed table	See notes	spalilddm.txt	spalilddm.schema	SPACE_ALLOC_ITEM_LD_DM, SPACE_ALLOC_IL_CUR_DM	BASEFACT_UPD	UPDATE_L	\${MMHOME}/data/spalilddm.txt	Space allocation item source data supplied by the client. This program will allow backposted data to process to compressed table.
stlblwdm.ksh	Stock Ledger	Base fact with update	RMS	stlblwdm.txt	stlblwdm.schema	INV_VAL_SBC_LW_DM	BASEFACT_UPD	UPDATE	\${MMHOME}/data/stlblwdm.txt	This program runs weekly.
stblmthdm.ksh	Stock Ledger	Base fact with update	RMS	stblmthdm.txt	stblmthdm.schema	INV_VAL_SBC_LMTH_DM	BASEFACT_UPD	UPDATE	\${MMHOME}/data/stblmthdm.txt	If 454 time alone is loaded, this module runs once a week. If Gregorian time is loaded, this module runs once a month.
sttflddm.ksh	Store Traffic	Base Fact	See notes	sttflddm.txt	sttflddm.schema	STORE_TRAF_LD_DM	BASEFACT_UPD	UPDATE	\${MMHOME}/data/sttflddm.txt	Source file supplied by client.
tsilddm.ksh	Sales productivity	Aggregation				SLS_LD_DM	FACT_AGG_QD	UPDATE_FS		
tsildqdm.ksh	Sales productivity	Aggregation				SLS_LQ_DM	FACT_AGG_STD	UPDATE		
ttlddm.ksh	Tender Transaction(Loss Prevention)	Aggregation				TNDR_TRAN_LD_DM	FACT_AGG_QD	UPDATE_F		

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control DM.program_type	Program_Control DM.operation_type	Argument s	Notes
ttldmdm.ksh	Tender Transaction(Loss Prevention)	Base Fact with incremental update	RMS	ttldmdm.txt	ttldmdm.schema	TNDR_TRAN_LM_DM	BASEFACT_INCR_UPD	UPDATE_G	\${MMHOME}/data/ttldmdm.txt	
ttldqdm.ksh	Tender Transaction(Loss Prevention)	Aggregation				TNDR_TRAN_LQ_DM	FACT_AGG_STD	UPDATE_MS		
vchreschddm.ksh	Escheated Vouchers	Base Fact with incremental update	RMS	vchreschlddm.txt	vchreschlddm.schema	VCHR_ESCH_DAY_DM	BASEFACT_INCR_UPD	UPDATE	\${MMHOME}/data/vchreschlddm.txt	
vchrmoveiddm.ksh	Voucher Movement	Base Fact aggregated from a staging table				VCHR_MOVE_LD_DM	FACT_AGG_STD	UPDATE_F		

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Program_Control DM.program_type	Program_Control DM.operation_type	Argument s	Notes
vchrmoveldsg.ksh	Voucher Movement	Staging Table		vchrmoveldsg.txt	vchrmoveldsg.schema	VCHR_MOVE_LD_SG	FACT_MATRIX	UPDATE	\${MMHOME}/data/vchrmovelddm.txt	This program loads the staging table VCHR_MOVE_LD_SG, which holds voucher movement facts at the individual voucher level. The program also includes code to decrement voucher movement facts when key information on the source record is updated (for instance, changing cashier or store for an existing voucher).
vchroutlwdm.ksh	Outstanding Vouchers	Base Fact with insert		vchroutlwdm.txt	vchroutlwdm.schema	VCHR_OUT_LW_DM	FACT_MATRIX	INSERT_G	\${MMHOME}/data/vchroutlwdm.txt	This program runs weekly.

Maintenance programs

The maintenance program reference lists begin on the next page.

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Arguments	Notes
factclosedm.ksh	Pre-Batch Maintenance	Maintenance		INV_IL_CUR_DM, INV_UNAVL_IL_CUR_DM, COST_ISL_CUR_DM, SPACE_ALLOC_IL_CUR_DM, SPACE_ALLOC_DL_CUR_DM, NET_COST_SIL_CUR_DM, CMPTR_PRICING_IL_CUR_DM, SUPP_AVAIL_I_CUR_DM, SUPP_CNTRCT_I_CUR_DM, PRICING_IL_CUR_DM, INVSBC_LW_DM, PROD_ITEM_DM PROD_ITEM_RECLASS_DM PROD_DEPT_RECLASS_DM ORG_LOC_RECLASS_DM		INV_ITEM_LD_DM, INV_UNAVL_ITEM_LD_DM, COST_ITEM_SUPP_LD_DM, SPACE_ALLOC_ITEM_LD_DM, CMPTR_PRICING_ITEM_LD_DM, PRICING_ITEM_LD_DM, NET_COST_SUPP_ITEM_LD_DM, SUPP_CNTRCT_ITEM_DAY_DM, SPACE_ALLOC_DEPT_LD_DM, INV_IL_CUR_DM, INV_UNAVL_IL_CUR_DM, COST_ISL_CUR_DM, SPACE_ALLOC_IL_CUR_DM, SPACE_ALLOC_DL_CUR_DM, NET_COST_SIL_CUR_DM, CMPTR_PRICING_IL_CUR_DM, SUPP_AVAIL_I_CUR_DM, SUPP_CNTRCT_I_CUR_DM SUPP_AVAIL_ITEM_DAY_DM INV_SBC_LW_DM		This program processes fact records whose items and/or locations and/or departments have closed or been reclassified. It runs at the beginning of a batch cycle (before mt_prime and after factopendm) and inserts stop records into the compressed tables so the decompression views will no longer pick up records whose items/locations/departments have been reclassified or closed. See Chapter 4, “Compression and partitioning” for details on this program.

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Arguments	Notes
factopendm.ksh	Pre-Batch Maintenance	Maintenance		INV_IL_CUR_DM, INV_UNAVL_IL_CUR_DM, COST_ISL_CUR_DM, SPACE_ALLOC_IL_CUR_DM, SPACE_ALLOC_DL_CUR_DM, NET_COST_SIL_CUR_DM, CMPTR_PRICING_IL_CUR_DM, SUPP_AVAIL_I_CUR_DM, SUPP_CNTRCT_I_CUR_DM PRICING_IL_CUR_DM PROD_ITEM_RECLASS_DM PROD_DEPT_RECLASS_DM ORG_LOC_RECLASS_DM		INV_ITEM_LW_DM, INV_ITEM_LD_DM, INV_UNAVL_ITEM_LD_DM, COST_ITEM_SUPP_LD_DM, SPACE_ALLOC_ITEM_LD_DM, CMPTR_PRICING_ITEM_LD_DM, PRICING_ITEM_LD_DM, NET_COST_SUPP_ITEM_LD_DM, SUPP_CNTRCT_ITEM_DAY_DM, SPACE_ALLOC_DEPT_LD_DM, INV_IL_CUR_DM, INV_UNAVL_IL_CUR_DM, COST_ISL_CUR_DM, SPACE_ALLOC_IL_CUR_DM, SPACE_ALLOC_DL_CUR_DM, NET_COST_SIL_CUR_DM, CMPTR_PRICING_IL_CUR_DM, SUPP_AVAIL_I_CUR_DM, SUPP_CNTRCT_I_CUR_DM, SUPP_AVAIL_ITEM_DAT_DM		This program runs immediately before factclosedm.ksh. The program inserts new records into compressed tables with the newly reclassified item/location/departments keys after a reclassification day. See Chapter 4, “Compression and partitioning” for details on this program.

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Arguments	Notes
medfactopendm.ksh	Pre-Batch Maintenance	Maintenance		PROD_ITEM_RECLASS_DM, MEDIA_ITEM_SI_DM		MEDIA_ITEM_SI_DM		This program runs immediately before factclosedm.ksh (and can be run concurrently with factopendm.ksh). The program inserts new records into the media fact table MEDIA_ITEM_SI_DM with the newly reclassified item keys after a reclassification day.

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Arguments	Notes
medlfl_dm.ksh	Media Transformation	Ad hoc Run when new media transformation relationships are desired.		media_lfl_by_media_dm.txt	media_lfl_by_media_dm.schema	MEDIA_LFL_BY_MEDIA_DM		<p>1. This program loads client-provided media transformation relationships between surrogate media keys.</p> <p>2. This program inserts directly from the text files into the target tables. Only new records in the text files are included (as opposed to a full snapshot of all the relationships). To indicate <i>no</i> relationship in a MEDIA_LFL_BY_MEDIA_DM row, use a -2 for the last season media key or the last year media key. This logic results from the fact that a -1 media key indicates 'no media' on the fact tables.</p> <p>3. This program does not call any libraries and does not use the PROGRAM_CONTROL_DM table.</p>

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Arguments	Notes
mt_prime.ksh	Pre-Batch Maintenance	Maintenance		MAINT_LOAD_DT_DM, TIME_DAY_DM, TIME_WK_DM		datekey.txt, nextdatekey.txt, currdayidnt.txt, nextdayidnt.txt, currwkidnt.txt, wkenddt.txt, nextwkidnt.txt mthidnt.txt, MAINT_LOAD_DT_DM, PROGRAM_STATUS_DM		<p>1. This program increments the processing date (curr_load_dt) by one day. This program will populate all date-related text files within \$etc directory by joining time_day_dm and time_wk_dm.</p> <p>2. The program, mt_prime.ksh, also prepares the batch cycle to be run by updating the PROGRAM_STATUS_DM table to 'ready' for all programs that have a 'completed' status. Any programs that are still in 'error' status from the previous run will have to be manually updated.</p>
orapartseed.ksh	Post-Batch Maintenance	Maintenance		cur table		Partitioned, compressed datamart table	table_name cur_table_name table_level	For partitioned, compressed datamarts, this program seeds the first day of a new partition with the current data from the cur_table. Note that this program applies only to Oracle clients. Explanation of arguments: table name refers to name of the target partitioned table;

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Arguments	Notes
								cur_table_name = name of the CUR position table associated with partitioned target table; table_level refers to level of the target partitioned table, either DAY or WEEK. This program must be run on partition seed days (the first day of the partition) and needs to be called once for every CUR table/compressed target table that is partitioned. For example, run once for INV_ITEM_LD_DM, once for INV_ITEM_LW_DM, once for PRICING_ITEM_LD_DM, and so on. See Chapter 4 for more information. If this program is run on a non-partition seed day, this program processes no data and terminates successfully.

Program	Functional Area	Program Type	External Data Source	Source Table or File	Schema File	Target File or Table	Arguments	Notes
seasonopendm.ksh	Pre-Batch Maintenance	Maintenance		SEASN_DM PROD_SEASN_ITEM_MTX_DM INV_UNAVL_IL_CUR_DM, INV_IL_CUR_DM		INV_UNAVL_ITEM_LD_DM, INV_UNAVL_IL_CUR_DM, INV_IL_CUR_DM, INV_ITEM_LD_DM, INV_ITEM_LW_DM		When an item crosses from one product season to another, a new prod_season_key is associated with the item_key. This change needs to be reflected in compressed facts that contain prod_season_key, namely Inventory Position. Seasnopendm maintains the inventory position datamart facts when a season change occurs.

Program type and operation type descriptions

With only a few exceptions, every RDW RETL program contains a program type and an operation type. The program type and operation type tell specific dimension and/or fact RDW RETL libraries how to process the data. The following tables detail every program type and operation type combination.

Dimension types

With regard to dimension types, the following assumptions apply:

- All dimension programs need to have a valid program type and operation type to be able to process data correctly.
- Dimension libraries handle much of the data processing by:
 - Creating one or more temporary tables
 - Analyzing the temporary table(s)
 - Creating an index on the temporary table
 - Generating the surrogate key for new and/or major changed records
 - Updating next_key_val on the MAINT_DIM_KEY_DM table
 - Updating or inserting into the target table based on the temporary table(s)
 - Updating program status to ‘completed’

Any exceptions to the above are detailed in the program and/or operation type description fields.

- In most cases, a temporary table(s) is created to help with dimension processing. This temporary table might be retained for program(s) later in the flow, such as item_key_lkup_temp.
The last batch program that uses the temporary table drops the temporary table.

DIM_TOP

Program type	Program type description	Operation type	Operation type description
DIM_TOP	<ul style="list-style-type: none">Used for dimension programs at the top of the hierarchy or not part of a hierarchy (standalone dimension such as currency), which have surrogate keys for all dimensional identifiers and all maintenance columns.Inserts are treated as new records; therefore, surrogate keys and all maintenance fields are generated before being inserted into the ‘insert’ temporary table.Deletes and minor changes are treated as updated records; therefore, some maintenance fields are updated before being inserted into the ‘update’ temporary table.	INSERT (the same as UPDATE)	
		UPDATE	<ul style="list-style-type: none">Both temporary tables will be dropped in dim_top_ksh.
		UPDATE_L	<ul style="list-style-type: none">Both temporary tables will be kept around for the program itself so that the program can do more processing.The program will manually update program status to ‘completed.’

DIM_TOP_DELTA

Program type	Program type description	Operation type	Operation type description
DIM_TOP_DELTA	<ul style="list-style-type: none">Used for dimension programs that only have new or updated data (as opposed to a complete snapshot).Used for dimension programs at the top of the hierarchy or not part of a hierarchy (standalone dimension, such as customer and customer account dimensions), which have surrogate keys for all dimension identifiers and the dm_recd_load_dt maintenance column.Source data has records with recd_type (‘I’, ‘U’, ‘X’). Records with ‘I’ will be inserted as new records. Records with ‘U’ will be updated. Records with ‘X’ will be updated or deleted based on the operation type.	UPDATE	<ul style="list-style-type: none">Both temporary tables will be dropped in dim_top_delta.ksh.
		UPDATE_D	<ul style="list-style-type: none">All temporary tables will be dropped in dim_top_delta.ksh.Closed records will be physically deleted from the target table.

DIM_TOP_DELTA_IDNT

Program type	Program type description	Operation type	Operation type description
DIM_TOP_DELTA_IDNT	<ul style="list-style-type: none">Used for dimension programs that only have new or updated data (as opposed to a complete snapshot).Used for dimension programs at the top of the hierarchy or not part of a hierarchy, which do not have surrogate keys for all dimension identifiers and must have the dm_recd_load_dt maintenance column.Source data has records with recd_type ('I', 'U', 'X'). Records with 'I' will be inserted as new records. Records with 'U' will be updated. Records with 'X' will be updated or deleted based on the operation type.	UPDATE	<ul style="list-style-type: none">Both temporary tables will be dropped in dim_top_delta.ksh.

DIM_TOP_F

Program type	Program type description	Operation type	Operation type description
DIM_TOP_F	<ul style="list-style-type: none">• Used for dimension programs at the top of the hierarchy or not part of a hierarchy, which have surrogate keys for all dimensional identifiers but not all maintenance columns. In other words, the history will not be kept if the record is deleted from the system.• Inserts are treated as new records; therefore, surrogate keys and all maintenance fields are generated before being inserted into the ‘insert’ temporary table.• Deletes are treated as deleted records; therefore, they are inserted into the ‘delete’ temporary table.• Minor changes are treated as updated records; therefore, some maintenance fields are updated before being inserted into the ‘update’ temporary table.	UPDATE_D	<ul style="list-style-type: none">• All three temporary tables will be dropped in the library dim_top.ksh.
		UPDATE_DL	<ul style="list-style-type: none">• All three temporary tables will be kept around for the program itself so the program can finish its processing.• The program itself will manually update its program status to ‘completed’.

DIM_TOP_IDNT

Program type	Program type description	Operation type	Operation type description
DIM_TOP_IDNT	<ul style="list-style-type: none">Used for dimension programs at the top of the hierarchy or not part of a hierarchy, which do not have surrogate keys for all dimensional identifiers and might not have all maintenance columns.Inserts are treated as new records; therefore, all maintenance fields are generated before being inserted into the 'insert' temporary table.Deletes are treated as deleted records; therefore, they are inserted into the 'delete' temporary table.Minor changes are treated as updated records; therefore, some maintenance fields are updated before being inserted into the 'update' temporary table.	UPDATE	<ul style="list-style-type: none">Does not use the 'delete' temporary table.Both the 'insert' and 'update' temporary tables will be dropped.
		UPDATE_D	<ul style="list-style-type: none">All three temporary tables will be dropped.

DIM_LOW

Program type	Program type description	Operation type	Operation type description
DIM_LOW	<ul style="list-style-type: none">Used for dimension programs at the lower level of the hierarchy, which have surrogate keys for all dimensional identifiers and all maintenance columns.Records are joined with parental table(s) to populate the parental information.Inserts and major changed inserts are treated as new records; therefore, surrogate keys and all maintenance fields are generated before being inserted into the ‘insert’ temporary table.Deletes, major changes deletes and minor changes are treated as updated records; therefore, some maintenance fields are updated before being inserted into the ‘update’ temporary table.A reclass temporary table might be created to keep all major changed records if defined in the program. This temporary table will be used by maintenance programs later.	UPDATE	<ul style="list-style-type: none">The dim_low.ksh library will handle all the processes.
		UPDATE_L	<ul style="list-style-type: none">Both temporary tables will be kept around for the program itself so that the program can perform more processing.The program will manually update program status to ‘completed.’

DIM_MTX

Program type	Program type description	Operation type	Operation type description
DIM_MTX	<ul style="list-style-type: none">Used for matrix programs that hold a relationship between one dimension and another dimension.	INSERT	<ul style="list-style-type: none">All records on the target table are deleted.All new records from the text file will be assigned corresponding surrogate keys before being inserted into the target table.No temporary table is generated or used.
		UPDATE	<ul style="list-style-type: none">Records from the text file will be assigned corresponding surrogate keys. The records include new inserts and major changed inserts/updates, and are inserted into a temporary table.The target table will be updated based on the temporary table.The temporary table will be dropped.

DIM_STANDALONE

Program type	Program type description	Operation type	Operation type description
DIM_STANDALONE	<ul style="list-style-type: none">Used for dimension programs that use a text file as input and do not need to call libraries.Checks for the existence of a source and schema file.	UPDATE	<ul style="list-style-type: none">This is a default operation type. No processing depends on this operation type.

DIM_STANDALONE_TABLE

Program type	Program type description	Operation type	Operation type description
DIM_STANDALONE_TABLE	<ul style="list-style-type: none">Used for dimension programs that use a table as input (as opposed to a text file) and do not need to call libraries.Does not check for the existence of a source and schema file.	UPDATE	<ul style="list-style-type: none">This is a default operation type. No processing depends on this operation type.

Fact types

With regard to fact types, the following assumptions apply:

- All fact programs need to have a valid program type and operation type to be able to process data correctly
- Fact libraries handle much of the data processing by:
 - Creating one or more temporary tables
 - Analyzing the temporary table(s)
 - Creating an index on the temporary table
 - Updating or inserting into the target table based on temporary table(s)
 - Updating the program status to 'completed'

Any exceptions to the above are detailed in the program and/or operation type description fields

- In most cases, a temporary table(s) is created to help with fact processing. This temporary table might be kept around for program(s) later in the flow. The last program to use the temporary table should drop it.

BASEFACT_INS

Program type	Program type description	Operation type	Operation type description
BASEFACT_INS	<ul style="list-style-type: none">Used for programs that only insert new records.If records come through with changed positional facts compared to the target table's positional records, the new position will be inserted into the target table with today's date.	INSERT	<ul style="list-style-type: none">Records are appended directly onto the temporary table.No temporary table is generated or used.
		INSERT_G	<ul style="list-style-type: none">Records are appended directly onto the target table.Parameters are specified by the program to indicate the grouping/summing fields.No temporary table is generated or used.
		UPDATE_A	<ul style="list-style-type: none">Records are appended directly onto the temporary table.The temporary table is kept around for use by another program later in the flow.

BASEFACT_UPD

Program type	Program type description	Operation type	Operation type description
BASEFACT_UPD	<ul style="list-style-type: none">• Used for programs that insert new records, and/or update the current records.• A temporary table is used to hold the current day's data to be used in the inserts and updates.	UPDATE	<ul style="list-style-type: none">• Records are updated from the temporary table to the target table.• The temporary table is dropped.
		UPDATE_L	<ul style="list-style-type: none">• Records will be inserted into a temporary table.• The temporary table is kept around for use by the program itself and another program later in the scheduling flow.• The program itself performs updates and inserts based on the temporary table created by the library. It needs to update its program status to 'completed' and drops the temporary table if no aggregation is needed later.• All compressed day level tables use this operation type.
		UPDATE_A	<ul style="list-style-type: none">• Records are updated/inserted from the temporary table to the target table.• The temporary table is kept around for use by another program later in the scheduling flow.

BASEFACT_INCR_UPD

Program type	Program type description	Operation type	Operation type description
BASEFACT_INCR_UPD	<ul style="list-style-type: none">Used for programs that insert new records, and incrementally update the existing records.The first temporary table holds current day’s data on the table.The second temporary table holds incremental updates for today’s changed data.	UPDATE	<ul style="list-style-type: none">Records are merged from both temporary tables and updated/inserted into the target table.The first temporary table is dropped.The second temporary table is dropped.
		UPDATE_A	<ul style="list-style-type: none">Records are merged from both temporary tables and updated/inserted into the target table.The first temporary table is kept around for use by another program later in the scheduling flow.The second temporary table is dropped.
		UPDATE_G	<ul style="list-style-type: none">Records are merged from both temporary tables, grouped by the grouping keys specified in the program, and updated/inserted into the target table.If a GROUP_CARRYOVER value is not specified, the first temporary table is dropped.The second temporary table is dropped.

FACT_AGG_POS

Program type	Program type description	Operation type	Operation type description
FACT_AGG_POS	<ul style="list-style-type: none">Used for programs that hold positional data for time and aggregates from a lower level to a higher level in the product hierarchy only.A temporary table from the previous program in the aggregation flow is used to hold the current day's data.	INSERT	<ul style="list-style-type: none">Records are updated on the target table based on the temporary table created by the previous program in the aggregation flow.The temporary table will be dropped.
		UPDATE_F	<ul style="list-style-type: none">A temporary table is created by parameters specified by the program.Records are updated on the target table based on the temporary table.The temporary table will be dropped.Any existing temporary tables from previous programs will be dropped.
		UPDATE_G	<ul style="list-style-type: none">A temporary table is created by parameters specified by the program, including the standard aggregation for product hierarchy.Records are updated on the target table based on the temporary table.The temporary table will be kept around for another program in the flow.Any existing temporary table from previous programs will be dropped.

Program type	Program type description	Operation type	Operation type description
		UPDATE_GF	<ul style="list-style-type: none">• A temporary table is created by parameters specified by the program, including the standard aggregation for product hierarchy.• Records are updated on the target table based on the temporary table.• The temporary table will be dropped.• Any existing temporary table from previous programs will be dropped.
		UPDATE_KO	<ul style="list-style-type: none">• A temporary table is created by parameters specified by the program.• Records are updated on the target table based on the temporary table.• The temporary table will be dropped.• Any existing temporary table from previous programs will be kept around for another program in the flow.

FACT_AGG_STD

Program type	Program type description	Operation type	Operation type description
FACT_AGG_STD	<ul style="list-style-type: none">• Used for programs that aggregate from a lower level to a higher level in the time and product hierarchy.• The base temporary table has been created to hold the current day's new or changed data.• The aggregation temporary table is created to hold the current day's aggregated data.• The second temporary table is created to hold the target table joined with the aggregation temporary table This will	UPDATE (UPDATE or UPDATE_S)	<ul style="list-style-type: none">• Records are updated/inserted into the target table based on joining the aggregation and second temporary tables.• The temporary table from previous programs in the aggregation flow will be kept around for another program in the flow.• The temporary table from the current program will be kept around for another program in the scheduling flow.

Program type	Program type description	Operation type	Operation type description
	<p>contain the target table records that need to be re-aggregated because data has been changed on and/or inserted into the base temporary table today.</p> <ul style="list-style-type: none">For DB2 clients only, programs of operation type with or without suffix S use different table spaces for the temporary table. If the previous program in the batch schedule uses operation type with suffix S, the program should not use operation type with suffix S, and vice versa. Program type and operation type do not matter in determining whether the operation type should have a suffix X, the order of the batch schedule matters.	UPDATE_F (UPDATE_F or UPDATE_FS)	<ul style="list-style-type: none">Records are updated/inserted into the target table based on joining the aggregation and second temporary tables.The temporary table from the previous program in the aggregation flow will be dropped.The temporary table from the current program will be dropped.
		UPDATE_M (UPDATE_M or UPDATE_MS)	<ul style="list-style-type: none">Records are updated/inserted into the target table based on joining the aggregation and second temporary tables.The temporary table from the previous program in the aggregation flow will be dropped.The temporary table from the current program will be kept around for another program later in the flow.
		UPDATE_KO	<ul style="list-style-type: none">Records are updated/inserted into the target table based on joining the aggregation and second temporary tables.The temporary table from the previous program in the aggregation flow will be kept around for another module in the flow.The temporary table from the current program will be dropped.

FACT_AGG_DW

Program type	Program type description	Operation type	Operation type description
FACT_AGG_DW	<ul style="list-style-type: none">• Used for programs that aggregate from day to week in the time hierarchy.• Utilizes the temporary table created in the previous program, which holds the current day's data.• The temporary table is created to hold aggregates of the base table based on today's data.• For DB2 clients only, programs of operation type with or without suffix S use different table spaces for the temporary table. If the previous program in the batch schedule uses operation type with suffix S, the program should not use operation type with suffix S, and vice versa. Program type and operation type do not matter in determining whether the operation type should have a suffix X, the order of the batch schedule matters.	UPDATE_F (UPDATE_F or UPDATE_FS)	<ul style="list-style-type: none">• Records are aggregated from the base table based on the temporary table from the previous program. These records are then updated/inserted into the target table.• The temporary table from the previous program in the aggregation flow will be dropped.• The temporary table from the current program will be dropped.

FACT_AGG_IB

Program type	Program type description	Operation type	Operation type description
FACT_AGG_IB	<ul style="list-style-type: none">• Used for programs that aggregate from item to subclass in the product hierarchy.• Utilizes the temporary table created in the previous program, which holds the current day's data.• The temporary table is created to hold aggregates of the base table based on today's data.• For DB2 clients only, programs of operation type with or without suffix S use different table spaces for the temporary table. If the previous program in the batch schedule uses operation type with suffix S, the program should not use operation type with suffix S, and vice versa. Program type and operation type do not matter in determining whether the operation type should have a suffix X, the order of the batch schedule matters.	UPDATE	<ul style="list-style-type: none">• Records are aggregated from the base table based on the temporary table from the previous program. These records are then updated/inserted into the target table.

FACT_AGG_QD

Program type	Program type description	Operation type	Operation type description
FACT_AGG_QD	<ul style="list-style-type: none">• Used for programs that aggregate from quarter hour to day in the time hierarchy. ·• Utilizes the temporary table created in the previous program, which holds the current day’s data.• The temporary table is created to hold aggregates of the base table based on today’s data.• For DB2 clients only, programs of operation type with or without suffix S use different table spaces for the temporary table. If the previous program in the batch schedule uses operation type with suffix S, the program should not use operation type with suffix S, and vice versa. Program type and operation type do not matter in determining whether the operation type should have a suffix X, the order of the batch schedule matters.	UPDATE_F (UPDATE_F or UPDATE_FS)	<ul style="list-style-type: none">• Records are aggregated from the base table based on the temporary table from the previous program. These records are then updated/inserted into the target table.·• The temporary table from the previous program in the aggregation flow will be dropped.• The temporary table from the current program will be dropped.

FACT_AGG_BD

Program type	Program type description	Operation type	Operation type description
FACT_AGG_BD	<ul style="list-style-type: none">• Used for programs that aggregate from subclass to department in the product hierarchy.• Utilizes the temporary table created in the previous program, which holds the current day's data.• The temporary table is created to hold aggregates of the base table based on today's data.• For DB2 clients only: The programs of operation type with or without the suffix S use different table spaces for the temporary table. If the previous program in the batch schedule uses operation type with suffix S, the program should <i>not</i> use operation type with suffix S, and vice versa. Program type and operation type do <i>not</i> matter in determining whether the operation type should have a suffix X; the order of the batch schedule matters.	UPDATE (UPDATE or UPDATE_S)	<ul style="list-style-type: none">• Records are aggregated from the base table based on the temporary table from the previous program. These records are then updated/inserted into the target table.• The temporary table from previous programs in the aggregation flow is retained for another program in the flow.• The temporary table from the current program is retained for another program in the scheduling flow.

FACT_AGG_ID

Program type	Program type description	Operation type	Operation type description
FACT_AGG_ID	<ul style="list-style-type: none">• Used for programs that aggregate from item to department in the product hierarchy.• Utilizes the temporary table created in the previous program, which holds the current day's data.• The temporary table is created to hold aggregates of the base table based on today's data.• For DB2 clients only, programs of operation type with or without suffix S use different table spaces for the temporary table. If the previous program in the batch schedule uses operation type with suffix S, the program should <i>not</i> use operation type with suffix S, and vice versa. Program type and operation type do <i>not</i> matter in determining whether the operation type should have a suffix X; the order of the batch schedule matters.	UPDATE (UPDATE or UPDATE_S)	<ul style="list-style-type: none">• Records are aggregated from the base table based on the temporary table from the previous program. These records are then updated/inserted into the target table.• The temporary table from previous programs in the aggregation flow is retained for another program in the flow.• The temporary table from the current program is retained for another program in the scheduling flow.

FACT_MATRIX

Program type	Program type description	Operation type	Operation type description
FACT_MATRIX	<ul style="list-style-type: none">Used for programs that require exception code or additional code for calculations, and/or additional non-standard dimensional joins.Temporary table is created based on the parameters specified by the program.For DB2 clients only, programs of operation type with or without suffix S use different table spaces for the temporary table. If the previous program in the batch schedule uses operation type with suffix S, the program should not use operation type with suffix S, and vice versa. Program type and operation type do not matter in determining whether the operation type should have a suffix S; the order of batch schedule matters.	INSERT	<ul style="list-style-type: none">Records are appended directly onto the target table.No temporary table is generated or used.
		INSERT_G	<ul style="list-style-type: none">Records are appended directly onto the target table.Parameters are specified by the program to indicate the grouping/summing fields.No temporary table is generated or used.
		UPDATE (UPDATE or UPDATE_S)	<ul style="list-style-type: none">Records are updated on the target table based on the temporary table.The temporary table will be dropped.Any existing temporary table from previous programs in the aggregation flow will also be dropped.
		UPDATE_A	<ul style="list-style-type: none">All records from the target table will be updated based on that temporary table.The temporary table will be kept around for another program.Any existing temporary table from previous programs in the aggregation flow will also be dropped.

FACT_STANDALONE

Program type	Program type description	Operation type	Operation type description
FACT_STANDALONE	<ul style="list-style-type: none">Used for fact programs that do not need to call any fact libraries.	UPDATE	<ul style="list-style-type: none">This is a default operation. No processing depends on this operation type.

Maintenance types

MAINTENANCE

Program type	Program type description	Operation type	Operation type description
MAINTENANCE	Used for programs that perform maintenance work and only need to call generic libraries.	UPDATE	<ul style="list-style-type: none">This is a default operation type. No processing depends on this operation type.

Appendix A – Application programming interface (API) flat file specifications

This appendix contains APIs that describe the file format specifications for all text files that serve as the interface between source systems and RDW. For example, these APIs control the formatting of the same fact and dimensions data as it is loaded into RDW.

In addition to providing individual field description and formatting information, the APIs provide basic business rules for the incoming data.

API format

Each API contains a business rules section and a file layout. Some general business rules and standards are common to all APIs. The business rules are used to ensure the integrity of the information held within RDW. In addition, each API contains a list of rules that are specific to that particular API.

File layout

- **Field Name:** Provides the name of the field in the text file.
- **Description:** Provides a brief explanation of the information held in the field.
- **Data Type/Bytes:** Includes both data type and maximum column length. Data type identifies one of three valid data types: character, number, or date. Bytes identifies the maximum bytes available for a field. A field may not exceed the maximum number of bytes (note that ASCII characters usually have a ratio of 1 byte = 1 character)
 - **Character:** Can hold letters (a,b,c...), numbers (1,2,3...), and special characters (\$,#,&...)
 - **Numbers:** Can hold only numbers (1,2,3...)
 - **Date:** Holds a specific year, month, day combination. The format is “YYYYMMDD”, unless otherwise specified.
- **Any required formatting for a field is conveyed in the Bytes section.** For example, Number(18,4) refers to number precision and scale. The first value is the precision and always matches the maximum number of digits for that field; the second value is the scale and specifies, of the total digits in the field, how many digits exist to the right of the decimal point. For example, the number –12345678901234.1234 would take up twenty ASCII characters in the flat file; however, the overall precision of the number is still (18,4).
- **Field Order:** Identifies the order of the field in the schema file.
- **Required Field:** Identifies whether the field can hold a null value. This section holds either a ‘yes’ or a ‘no’. A ‘yes’ signifies the field may not hold a null value. A ‘no’ signifies that the field may, but is not required, to hold a null value.

General business rules and standards common to all APIs

- Complete 'snapshot' of dimension data:
A majority of RDW's dimension code requires a complete view of all current dimensional data (regardless of whether the dimension information has changed) once at the end of every business day. If a complete view of the dimensional data is not provided in the text file, invalid or incorrect dimensional data can result. For instance, not including an active item in the prditmdm.txt file causes that item to be closed (as of the extract date) in the data warehouse. When a sale for the item is processed, the fact program will not find a matching 'active' dimension record. Therefore, it is essential, unless otherwise noted in each API's specific business rules section, that a complete snapshot of the dimensional data be provided in each text file.

If there are no records for the day, an empty flat file must still be provided.

- Updated and new records of fact data:
Facts being loaded to RDW can either be new or updated facts. Unlike dimension snapshots, fact flat files will only contain new/updated facts exported from the source system once per day (or week, in some cases). Refer to each API's specific business rules section for more details.

If there are no new or changed records for the day, an empty flat file must still be provided.

- Primary and local currency amount fields
Amounts will be stored in both primary and local currencies for most fact tables. If the source system uses multi-currency, then the primary currency column holds the primary currency amount, and the local currency column holds the local currency amount. If the location happens to use the primary currency, then both primary and local amounts hold the primary currency amount. If the source system does not use multi-currency, then only the primary currency fields are populated and the local fields hold NULL values.
- Leading/trailing values:
Values entered into the text files are the exact values processed and loaded into the datamart tables. Therefore, the values with leading and/or trailing zeros, characters, or nulls are processed as such. RDW does not strip any of these leading or trailing values, unless otherwise noted in the individual API's business rules section.
- Indicator columns:
Indicator columns in RDW (such as prod_item_dm.pack_ind) are assumed to hold one of two values, either "Y" for yes or "N" for no.

- Delimiters:



Note: Make sure the delimiter is never part of your data.

- **Dimension Flat File Delimiter Standards:** Within dimension text files, each field must be separated by a pipe (|) character, for example a record from prddivdm.txt may look like the following:

```
1000|1|Homewares|2006|Henry Stubbs|2302|Craig Swanson
```

- **Fact Flat File Delimiter Standards:** Within facts text files, each field must be separated by a semi-colon character (;). For example a record from exchngratedm.txt may look like the following:

```
WIS;20010311;1.73527820592648544918
```

See the latest RETL Programmer's Guide for additional information.

- End of Record Carriage Return:

Each record in the text file must be separated by an end of line carriage return. For example, the three records below, in which each record holds four values, should be entered as:

```
1|2|3|4
```

```
5|6|7|8
```

```
9|10|11|12
```

and not as a continuous string of data, such as:

```
1|2|3|4|5|6|7|8|9|10|11|12
```

There should never be a carriage return at the end of the final record. If there is a carriage return, the system treats it as a new record, and you may receive an error when running RETL program.

Flat file specifications

cdedtlcomdm.txt

Business rules:

- This interface file cannot contain duplicate records for a cde_type_idnt, cde_idnt combination.
- This interface file contains the complete snapshot of active information.
- This interface file contains codes for the following code types (cde_type_idnt): CRQSTTYP (Catalog Request Types), ARQSTTYP (Activity Request Types), CTLGTYPE (Catalog Types), RQSTORGN (Request Origins), SVCCOLR (Value Added Service Color), SVCFONT (Value Added Service Font), SVCTYPE (Value Added Service Type), COHOLDEVENT (Customer Order Hold Events), DISPO (Disposition code type), PRMTRIG (Promotion trigger type), COPARTREASN (Customer Order Partial Line Reason).
- RCOM adds the following two rows to flat file for multi-color and multi-font value added service lines. CDE_TYPE_IDNT:SVCCOLR CDE_IDNT:Multi-Color
CDE_DESC:Multi-Color CDE_TYPE_IDNT:SVCFONT CDE_IDNT:Multi-Font
CDE_DESC:Multi-Font
- This interface file follows the dimension flat file interface layout standard.
- RCOM provides only distinct value added service colors and fonts. If ten suppliers have 'RED' as an available color, 'RED' appears once in the flat file.

Name	Description	Data Type/Bytes	Field Order	Required Field
CDE_TYPE_IDNT	The code type, which serves as a grouping mechanism for the different codes stored on the CDE_DTL_COM_DM table.	CHARACTER(12)	1	Yes
CDE_IDNT	The unique identifier for the code within a code type.	CHARACTER(120)	2	Yes
CDE_DESC	The description associated with the code.	CHARACTER(160)	3	No

cdedtldm.txt

Business rules:

- This data is loaded during installation.

- This interface file contains code and code description accessed by the front-end portion of RDW through specific database views.
- This interface file cannot contain duplicate records for a cde_type, cde combination.
- This interface file contains the complete snapshot of active information.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field order	Required field
CDE_TYPE	The code type, which serves as a grouping mechanism for the different codes stored on the CDE_DTL_DM table.	CHARACTER(6)	1	Yes
CDE	The unique identifier for the code within a code type.	CHARACTER(6)	2	Yes
CDE_DESC	The description associated with the code.	CHARACTER(120)	3	Yes

cllctrdm.txt

Business rules:

- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.
- This interface file cannot contain duplicate records for a call center location identifier.

Name	Description	Data Type/Bytes	Field Order	Required Field
CALL_CTR_IDNT	The unique identifier of a call center.	CHARACTER(10)	1	Yes
CALL_CTR_DESC	The description of a call center.	CHARACTER(120)	2	No

cmptrdm.txt

Business rules:

- This interface file contains competitor information.
- This interface file cannot contain duplicate records for a cmptr_idnt.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field order	Required field
CMPTR_IDNT	The unique identifier of the competitor	CHARACTER(10)	1	Yes
CMPTR_DESC	The name of competitor.	CHARACTER(120)	2	No
CMPTR_ADDR	The competitor address.	CHARACTER(255)	3	No
CMPTR_CITY_NAME	The competitor city	CHARACTER(120)	4	No
CMPTR_ST_OR_PRV NC_CDE	The competitor state or province.	CHARACTER(3)	5	No
CMPTR_CNTRY_CDE	The competitor country	CHARACTER(10)	6	No

cmptrlmdm.txt

Business rules:

- This interface file defines the association between location and competitor location.
- This interface file cannot contain duplicate records for a cmptr_loc_idnt and cmptr_idnt combination.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field order	Required field
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	1	Yes
CMPTR_LOC_IDNT	The unique identifier of the competitor location.	CHARACTER(10)	2	Yes

Name	Description	Data Type/Bytes	Field order	Required field
TARGET_CMPTR_IND	Identifies the target competitor of a retailer's store. This competitor's retail will be used along with the primary store within a zone when calculating a recommended retail in Price Management. Valid values are: Y, and N.	CHARACTER(1)	3	Yes
CMPTR_RANK	The rank of each competitor store compared to the other stores.	NUMBER(2)	4	No
DISTANCE	The distance between the retailer's store and the competitor's store.	NUMBER(4)	5	No
DISTANCE_UOM_CDE	The unit of measure code the distance is captured in. Valid values are 1 = 'Miles', 2 = 'Kilometers'.	CHARACTER(6)	6	No
DISTANCE_UOM_DESC	The unit of measure description the distance is captured in.	CHARACTER(120)	7	No

cmptrlocdm.txt

Business rules:

- This interface file contains non-historical information about competitors and their individual locations.
- This interface file cannot contain duplicate records for a cmptr_loc_idnt, cmptr_idnt combination.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field order	Required field
CMPTR_LOC_IDNT	The unique identifier of the competitor location	CHARACTER(10)	1	Yes
CMPTR_IDNT	The unique identifier of the competitor	CHARACTER(10)	2	Yes
CMPTR_LOC_DESC	The competitor store description	CHARACTER(120)	3	No
CMPTR_LOC_ADDR	The competitor store's address	CHARACTER(255)	4	No
CMPTR_LOC_CITY_NAME	The competitor store city	CHARACTER(120)	5	No
CMPTR_LOC_ST_OR_PRVNC_CDE	The competitor store state	CHARACTER(3)	6	No
CMPTR_LOC_CNTRY_CDE	The competitor store country	CHARACTER(10)	7	No

Name	Description	Data Type/Bytes	Field order	Required field
ESTIMATED_VOLUME	The estimated yearly sales volume of the competitor at assigned location.	NUMBER(18,4)	8	No
CMPTR_CRNCY_CDE_IDNT	The unique identifier of the currency code. E.g: USD is the local currency code for US Dollar	CHARACTER(10)	9	Yes

cmptprcildm.txt

Business rules:

- This interface file contains competitor's pricing facts for the client location, competitor location and item combination on a given day.
- This interface file cannot contain duplicate transactions for item_idnt, loc_idnt, cmptr_loc_idnt, day_dt combinations.
- This interface file follows the fact flat file interface layout standard.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
CMPTR_LOC_IDNT	The unique identifier of the competitor location.	CHARACTER(10)	3	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	4	Yes

Name	Description	Data Type/Bytes	Field order	Required field
F_CMPTR_UNIT_RTL_AMT	The competitor's unit retail amount for a particular item in primary currency.	NUMBER(18, 4)	5	No
F_CMPTR_UNIT_RTL_AMT_LCL	The competitor's unit retail amount for a particular item in local currency.	NUMBER(18, 4)	6	No
F_CMPTR_MULTI_UNIT_RTL_AMT	The competitor's multi unit retail amount for a particular item in primary currency.	NUMBER(18, 4)	7	No
F_CMPTR_MULTI_UNIT_RTL_AMT_LCL	The competitor's multi unit retail amount for a particular item in local currency.	NUMBER(18, 4)	8	No
RTL_TYPE_CDE	The price type ('R'egular, 'P'romotion, 'C'learance).	CHARACTER(2)	9	Yes
OFFER_TYPE_CDE	This non-aggregatable field identifies the offer type code of the competitor's promotional retail. Examples of valid values are 1 = 'Coupon', 2= 'Mailer', etc.	CHARACTER(6)	10	No
MULTI_UNITS_QTY	This non-aggregatable field identifies the multi units associated with F_CMPTR_UNIT_RTL_AMT for a particular item.	NUMBER(12, 4)	11	No

coeopilddm.txt

Business rules:

- This interface file cannot contain duplicate records for co_line_idnt, co_hdr_idnt, co_line_media_idnt, co_hdr_media_idnt, banner_idnt, selling_item_idnt, item_idnt, loc_idnt, cust_idnt, co_line_type_idnt, day_dt combination.

Appendix A – Application programming interface (API) flat file specifications

- This data must be extracted from the source system after midnight, and only data created in the system before midnight must be extracted.
- This interface file includes the complete snapshot of all open customer order lines (all order lines that have not been completely shipped or have not been completely cancelled).
- Only today's order line positions are expected (back posted position and/or updates of previous days positions should not be provided).
- This interface file follows the fact flat file interface layout standard.
- If a dimension identifier is required but is not available, a value of -1 is needed.
- Only records that have an f_eop_rsv_qty, f_eop_pick_qty, or f_eop_bo_qty greater than zero should be provided.
- This interface file contains order line positions for outgoing order line types only, not for incoming order line types, such as returns.

Name	Description	Data Type/Bytes	Field Order	Required Field
CO_LINE_IDNT	The unique identifier of a customer order line.	CHARACTER(30)	1	Yes
CO_HDR_IDNT	The unique identifier of a customer order header.	CHARACTER(30)	2	Yes
LINE_MEDIA_IDNT	The unique identifier of the line media.	CHARACTER(10)	3	Yes
HDR_MEDIA_IDNT	The unique identifier of the header media.	CHARACTER(10)	4	Yes
BANNER_IDNT	The identifier of a banner.	CHARACTER(4)	5	Yes
SELLING_ITEM_IDNT	The unique identifier of a selling item.	CHARACTER(25)	6	Yes
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	7	Yes
LOC_IDNT	The unique identifier of a location.	CHARACTER(10)	8	Yes
CUST_IDNT	The unique identifier of the customer placing the order.	CHARACTER(15)	9	Yes
CO_LINE_TYPE_IDNT	The unique identifier of a customer order line type.	CHARACTER(120)	10	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	11	Yes
CO_HOLD_EVENT_IDN	Identifies the event why	CHARACTER	12	No

Name	Description	Data Type/Bytes	Field Order	Required Field
T	an order line is being held.	R(60)		
DROP_SHIP_IND	An indicator to identify if an item is shipped directly to the customer.	CHARACTER(1)	13	No
CO_GIFT_IND	An indicator to identify if the item on the order line is a gift.	CHARACTER(1)	14	No
F_EOP_RSV_QTY	The reserve quantity at the end of the period.	NUMBER(12, 4)	15	No
F_EOP_PICK_QTY	The pick quantity at the end of the period.	NUMBER(12, 4)	16	No
F_EOP_BO_QTY	The backorder quantity at the end of the period.	NUMBER(12, 4)	17	No
F_EOP_RSV_AMT	The reserve retail value at the end of the period.	NUMBER(18, 4)	18	No
F_EOP_RSV_AMT_LCL	The reserve retail value at the end of the period in local currency.	NUMBER(18, 4)	19	No
F_EOP_PICK_AMT	The pick retail value at the end of the period.	NUMBER(18, 4)	20	No
F_EOP_PICK_AMT_LCL	The pick retail value at the end of the period in local currency.	NUMBER(18, 4)	21	No
F_EOP_BO_AMT	The backorder retail value at the end of the period.	NUMBER(18, 4)	22	No
F_EOP_BO_AMT_LCL	The backorder retail value at the end of the period in local currency.	NUMBER(18, 4)	23	No

cohdrdm.txt

Business rules:

- This data must be extracted from the source system after midnight, and only data created in the system before midnight must be extracted.
- This interface file cannot contain duplicate records for a co_hdr_idnt.
- This interface file contains only the new or changed customer order header information since the last time data was extracted from the source system.

Appendix A – Application programming interface (API) flat file specifications

- All monetary columns except return columns and cancel columns must keep the existing values when a return or cancel occurs. Only return columns should be populated with returned values when returns occur, and only cancel columns should be populated with cancelled values when cancels occur.
- All data in this interface must not include partial, replacement in and replacement out transactions
- For exchange out transactions of any of its order lines, the order header's affected columns must be updated.
- Only changes for the defined fields in the API specifications are considered.
- If a dimension identifier is required but is not available, a value of -1 is needed.
- This interface file follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
CO_HDR_IDNT	The unique identifier of a customer order header.	CHARACTER(30)	1	Yes
HDR_MEDIA_IDNT	The unique identifier of the customer order header-level media.	CHARACTER(10)	2	Yes
BANNER_IDNT	The identifier of a banner.	CHARACTER(4)	3	Yes
CSR_IDNT	The unique identifier for a customer service representative.	CHARACTER(30)	4	Yes
CUST_IDNT	The unique identifier of the customer.	CHARACTER(15)	5	Yes
DAY_DT	The calendar day on which the customer order was created.	DATE	6	Yes
F_ACCOM_AMT	The total accommodations given to a customer for this customer order in primary currency.	NUMBER(18,4)	7	No
F_ACCOM_AMT_LCL	The total accommodations given to a customer for this customer order, in local currency.	NUMBER(18,4)	8	No
F_RTRN_ACCOM_AMT	The amount taken off from the original accommodation due to returns in primary	NUMBER(18,4)	9	No

Name	Description	Data Type/Bytes	Field Order	Required Field
	currency.			
F_RTRN_ACCOM_AMT_LCL	The amount taken off from the original accommodation due to returns, in local currency.	NUMBER(18,4)	10	No
F_CNCL_ACCOM_AMT	The amount taken off from the original accommodation due to cancels in primary currency.	NUMBER(18,4)	11	No
F_CNCL_ACCOM_AMT_LCL	The amount taken off from the original accommodation due to cancels, in local currency	NUMBER(18,4)	12	No
F_DLVRY_AMT	The shipping and handling charge applied to the entire customer order in primary currency.	NUMBER(18,4)	13	No
F_DLVRY_AMT_LCL	The shipping and handling charge applied to the entire customer order in local currency.	NUMBER(18,4)	14	No
F_RTRN_DLVRY_AMT	The amount taken off from the original shipping and handling charges due to returns in primary currency.	NUMBER(18,4)	15	No
F_RTRN_DLVRY_AMT_LCL	The amount taken off from the original shipping and handling charges due to returns, in local currency.	NUMBER(18,4)	16	No
F_CNCL_DLVRY_AMT	The amount taken off from the original shipping and handling charges due to cancels in primary currency.	NUMBER(18,4)	17	No
F_CNCL_DLVRY_AMT_LCL	The amount taken off from the original shipping and handling charges due to cancels, in local currency.	NUMBER(18,4)	18	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_RUSH_DLVRY_AMT	The rush shipping and handling charge applied to the entire customer order in primary currency.	NUMBER(18,4)	19	No
F_RUSH_DLVRY_AMT_LCL	The rush shipping and handling charge applied to the entire customer order, in local currency.	NUMBER(18,4)	20	No
F_RTRN_RUSH_DLVRY_AMT	The amount taken off from the original rush shipping and handling charge due to returns, in primary currency.	NUMBER(18,4)	21	No
F_RTRN_RUSH_DLVRY_AMT_LCL	The amount taken off from the original rush shipping and handling charge due to returns, in local currency.	NUMBER(18,4)	22	No
F_CNCL_RUSH_DLVRY_AMT	The amount taken off from the original rush shipping and handling charge due to cancels, in primary currency.	NUMBER(18,4)	23	No
F_CNCL_RUSH_DLVRY_AMT_LCL	The amount taken off from the original rush shipping and handling charge due to cancels, in local currency.	NUMBER(18,4)	24	No
F_PRMTN_DSCNT_AMT	The shipping and handling promotional discount applied to the entire customer order, in primary currency.	NUMBER(18,4)	25	No
F_PRMTN_DSCNT_AMT_LCL	The shipping and handling promotional discount applied to the entire customer order, in local currency.	NUMBER(18,4)	26	No
F_RTRN_PRMTN_DSCNT_AMT	The amount taken off from the original shipping and handling promotional discount due	NUMBER(18,4)	27	No

Name	Description	Data Type/Bytes	Field Order	Required Field
	to returns, in primary currency.			
F_RTRN_PRMTN_DSCN T_AMT_LCL	The amount taken off from the original shipping and handling promotional discount due to returns, in local currency.	NUMBER(18,4)	28	No
F_CNCL_PRMTN_DSCN T_AMT	The amount taken off from the original shipping and handling promotional discount due to cancels, in primary currency.	NUMBER(18,4)	29	No
F_CNCL_PRMTN_DSCN T_AMT_LCL	The amount taken off from the original shipping and handling promotional discount due to cancels, in local currency.	NUMBER(18,4)	30	No
F_TAXABLE_AMT	The total taxable amount for the customer order, including services, in primary currency.	NUMBER(18,4)	31	No
F_TAXABLE_AMT_LCL	The total taxable amount for the customer order, including services, in local currency.	NUMBER(18,4)	32	No
F_RTRN_TAXABLE_A MT	The amount taken off from the original taxable amount due to returns, in primary currency.	NUMBER(18,4)	33	No
F_RTRN_TAXABLE_A MT_LCL	The amount taken off from the original taxable amount due to returns, in local currency.	NUMBER(18,4)	34	No
F_CNCL_TAXABLE_A MT	The amount taken off from the original taxable amount due to cancels, in primary currency.	NUMBER(18,4)	35	No
F_CNCL_TAXABLE_A MT_LCL	The amount taken off from the original taxable amount due to cancels, in	NUMBER(18,4)	36	No

Name	Description	Data Type/Bytes	Field Order	Required Field
	local currency.			

coilnlldm.txt

Business rules:

- This data must be extracted from the source system after midnight, and only data created in the system before midnight must be extracted.
- This interface file contains only the current day's new or changed Order Line information, one record per new or changed order line per day.
- All the quantity and amount fields are transactional quantities and amounts. These transactional quantities and amounts must contain the delta values of the previous value and current value. They must be filled in when a transaction occurs. If the order line goes to same status several times in one day, only the latest transaction quantity is captured.
- If there is no transaction quantity change but some other fields change, the record for this order line must include those changes and include the quantity fields with null values.
- When positional statuses (Backorder and Picking) change to other statuses, the end date fields must be filled in with the current date. The quantity fields for those statuses must be filled in with a null value.
- If an order line stays in backorder status but some of the items are partial shipped, an intermediate 'PICKING' status is expected even if the order line is actually still in backorder status. This logic results in the picking quantity bucket's being filled in with the partial shipped quantity and the picking end date's being filled in with current date.
- When all the remaining items for the order line are shipped, field ship_dt must be filled in with the current date.
- This interface file includes order lines of type NORMAL(N), UPSELL(U), CROSS-SELL(C), SUBSTITUTE(S), EXCHANGE OUT(ES), REPLACEMENT OUT(RS) and PARTIAL(P). Order lines of type RETURN(R), EXCHANGE IN(ER), and REPLACEMENT IN(RR) are excluded from this interface file.
- When all the remaining items for the order line are cancelled, field cncl_dt must be filled in with the current date.
- The banner_idnt corresponding to the hdr_media_idnt and line_media_idnt must be the same.
- If a dimension identifier is required but is not available, a value of -1 is needed.
- This interface file follows the fact flat file interface layout standard.
- If there are multiple ship-to address for an order line, the primary ship-to address is expected.
- The ship_dt and cncl_dt fields are mutually exclusive.
- Any cancel quantity that does not require a separate customer order line must be reflected in the cancelled quantity field f_cncl_qty.

Name	Description	Data Type/Bytes	Field Order	Required Field
CO_LINE_IDNT	The unique identifier of a customer order line.	CHARACTER(30)	1	Yes
CO_DAY_DT	The date when the customer order was created.	DATE	2	Yes
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	3	Yes
SELLING_ITEM_IDNT	The unique identifier of a selling item.	CHARACTER(25)	4	Yes
HDR_MEDIA_IDNT	The unique identifier of the customer order header-level media.	CHARACTER(10)	5	Yes
LINE_MEDIA_IDNT	The unique identifier of the customer order line level media.	CHARACTER(10)	6	Yes
BANNER_IDNT	The unique identifier of a banner.	CHARACTER(4)	7	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	8	Yes
CSR_IDNT	The unique identifier of a customer service representative.	CHARACTER(30)	9	Yes
CUST_IDNT	The unique identifier of the customer placing the order.	CHARACTER(15)	10	Yes
CO_HDR_IDNT	The unique identifier of a customer order header.	CHARACTER(30)	11	Yes
DAY_DT	The transaction date when the customer order line was created or modified.	DATE	12	Yes
CO_SHIP_TO_CITY	The name of the city this order line is shipped to.	CHARACTER(120)	13	Yes
CO_SHIP_TO_ST_OR_P RVNC_CDE	The code of the state or province this order line is shipped to.	CHARACTER(3)	14	Yes
CO_SHIP_TO_CNTRY_C DE	The code of the country this order line is shipped to.	CHARACTER(120)	15	Yes

Name	Description	Data Type/Bytes	Field Order	Required Field
CO_SHIP_TO_PSTL_CD E	The sip code of the address this order line is shipped to (no additional 4 digits for US address).	CHARACTER(30)	16	Yes
CARRIER_IDNT	The unique identifier if a carrier delivering the order line.	CHARACTER(10)	17	Yes
CARRIER_SVC_IDNT	The unique identifier of a carrier level of service (rush, standard, and so on) to be used in delivering the order line.	CHARACTER(10)	18	Yes
CO_DMND_STTS_IDNT	Predefined demand status codes to show the reason when the order line is cancelled.	CHARACTER(12)	19	Yes
REF_ITEM_IDNT	The unique identifier of the item that triggered the up-sell, cross-sell, substitute, or partial activity. This column is only be populated when the customer order line type is upsell, cross-sell, or substitute; otherwise it will be -1.	CHARACTER(25)	20	Yes
CO_LINE_TYPE_IDNT	Identifies a customer order line type. Examples are up-sell, cross-sell, normal, partial, etc.	CHARACTER(120)	21	Yes
CO_PARTIAL_REASN_IDNT	Identifies the reason the partial order line was created.	CHARACTER(120)	22	Yes
CO_HOLD_EVENT_IDNT	Identifies the event for which an order line is being held.	CHARACTER(120)	23	Yes
DROP_SHIP_IND	An indicator to identify if an item is shipped directly to the customer.	CHARACTER(1)	24	No
CO_GIFT_IND	An indicator to identify if the item on the order line is a gift.	CHARACTER(1)	25	No

Name	Description	Data Type/Bytes	Field Order	Required Field
CO_EST_DLVRY_DT	The estimated date the customer order line will be delivered.	DATE	26	No
PICK_END_DT	The date when the items in the customer order line have been completely picked in the warehouse.	DATE	27	No
BO_END_DT	The date the customer order line moves out of 'Back Order' status for the last time.	DATE	28	No
SHIP_DT	The date the customer order line has been completely shipped.	DATE	29	No
CNCL_DT	The date the customer order line has been completely canceled.	DATE	30	No
F_CO_QTY	The quantity ordered.	NUMBER(12, 4)	31	No
F_RSV_QTY	The quantity reserved.	NUMBER(12, 4)	32	No
F_PICK_QTY	The quantity picked.	NUMBER(12, 4)	33	No
F_BO_QTY	The quantity backordered.	NUMBER(12, 4)	34	No
F_SHIP_QTY	The quantity shipped.	NUMBER(12, 4)	35	No
F_CNCL_QTY	The quantity cancelled.	NUMBER(12, 4)	36	No
F_CO_UNIT_RTL_AMT	The transaction unit price of the order line item expected to be paid by the customer, in primary currency.	NUMBER(18, 4)	37	No
F_CO_UNIT_RTL_AMT_LCL	The transaction unit price of the order line item expected to be paid by the customer, in local currency.	NUMBER(18, 4)	38	No
F_CO_MEDIA_UNIT_RT	The media selling price of the order line item, in	NUMBER(18, 4)	39	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field Order	Required Field
L_AMT	primary currency.	4)		
F_CO_MEDIA_UNIT_RT L_AMT_LCL	The media selling price of the order line item, in local currency.	NUMBER(18, 4)	40	No
F_ADDL_DLVR_Y_AMT	The additional shipping and handling charge applied to the order line, in primary currency.	NUMBER(18, 4)	41	No
F_ADDL_DLVR_Y_AMT _LCL	The additional shipping and handling charge applied to the order line, in local currency.	NUMBER(18, 4)	42	No
F_CNCL_ADDL_DLVR_Y _AMT	The amount taken off from the original additional shipping and handling charge due to cancels, in primary currency.	NUMBER(18, 4)	43	No
F_CNCL_ADDL_DLVR_Y _AMT_LCL	The amount taken off from the original additional shipping and handling charge due to cancels, in local currency.	NUMBER(18, 4)	44	No
F_PRMTN_L_DSCNT_A MT	The total promotional discount applied to the order line, in primary currency.	NUMBER(18, 4)	45	No
F_PRMTN_L_DSCNT_A MT_LCL	The total promotional discount applied to the order line, in local currency.	NUMBER(18, 4)	46	No
F_CNCL_PRMTN_L_DS CNT_AMT	The amount taken off from the original promotional discount due to cancels, in primary currency.	NUMBER(18, 4)	47	No
F_CNCL_PRMTN_L_DS CNT_AMT_LCL	The amount taken off from the original promotional discount due to cancels, in local currency.	NUMBER(18, 4)	48	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_SVC_AMT	The sum of the service charges applied to the customer order service lines, in primary currency.	NUMBER(18, 4)	49	No
F_SVC_AMT_LCL	The sum of the service charges applied to the customer order service lines, in local currency.	NUMBER(18, 4)	50	No
F_CNCL_SVC_AMT	The amount taken off from the original service charge due to cancels, in primary currency.	NUMBER(18, 4)	51	No
F_CNCL_SVC_AMT_LCL	The amount taken off from the original service charge due to cancels, in local currency.	NUMBER(18, 4)	52	No
F_ADDL_DLVR_TAX_AMT	The additional shipping and handling charge tax applied to the order line, in primary currency.	NUMBER(18, 4)	53	No
F_ADDL_DLVR_TAX_AMT_LCL	The additional shipping and handling charge tax applied to the order line, in local currency.	NUMBER(18, 4)	54	No
F_CNCL_ADDL_DLVR_TAX_AMT	The amount taken off from the original additional shipping and handling charge tax due to cancels, in primary currency.	NUMBER(18, 4)	55	No
F_CNCL_ADDL_DLVR_TAX_AMT_LCL	The amount taken off from the original additional shipping and handling charge tax due to cancels, in local currency.	NUMBER(18, 4)	56	No
F_MRCH_TAX_AMT	The merchandise tax applied to the order line in primary currency.	NUMBER(18, 4)	57	No
F_MRCH_TAX_AMT_LCL	The merchandise tax applied to the order line	NUMBER(18, 4)	58	No

Name	Description	Data Type/Bytes	Field Order	Required Field
	in local currency.			
F_CNCL_MRCH_TAX_AMT	The amount taken off from the original merchandise tax due to cancels, in primary currency.	NUMBER(18, 4)	59	No
F_CNCL_MRCH_TAX_AMT_LCL	The amount taken off from the original merchandise tax due to cancels, in local currency.	NUMBER(18, 4)	60	No
F_SVC_TAX_AMT	The service charge tax applied to the order line, in primary currency.	NUMBER(18, 4)	61	No
F_SVC_TAX_AMT_LCL	The service charge tax applied to the order line, in local currency.	NUMBER(18, 4)	62	No
F_CNCL_SVC_TAX_AMT	The amount taken off from the original service charge tax due to cancels, in primary currency.	NUMBER(18, 4)	63	No
F_CNCL_SVC_TAX_AMT_LCL	The amount taken off from the original service charge tax due to cancels, in local currency.	NUMBER(18, 4)	64	No

coilnlsq_pre.txt

Business rules:

- This data must be extracted from the source system after midnight, and only data created in the system before midnight must be extracted.
- This interface file includes order lines of type NORMAL(N), UPSELL(U), CROSS-SELL(C), SUBSTITUTE(S), EXCHANGE OUT(ES), REPLACEMENT OUT(RS) and PARTIAL(P). Order lines of type RETURN(R), EXCHANGE IN(ER), and REPLACEMENT IN(RR) are excluded from this interface file.
- This interface file contains only the current day's new or changed Order Line information, one record per new or changed order line per day.
- The banner_idnt corresponding to the hdr_media_idnt and line_media_idnt must be the same.

- For all quantity fields except f_pick_eod_qty and f_bo_eod_qty, the maximum quantity for a given status must be provided each day. For example, if quantity 2 goes into picking in the morning, and then into another status, and then 4 go back into picking by the end of the day, f_pick_qty is 4. All other fields must be populated with the latest value for those fields as of the end of the day.
- If an order line stays in backorder status but some of the items have been partially shipped, an intermediate picking status is expected even if the order line is actually still in backorder status. This logic results in the f_pick_qty bucket's being filled in with the partial shipped quantity.
- When all the remaining items for the order line are shipped, field ship_dt must be filled in with the current date.
- If there are multiple ship-to addresses for an order line, the primary ship to address is expected.
- If a dimension identifier is required but is not available, a value of -1 is needed.
- This interface file follows the fact flat file interface layout standard.
- Any decrease in the order quantity field must be reflected in the cancelled quantity field f_cncl_qty.
- When all the remaining items for the order line are cancelled, field cncl_dt must be filled in with the current date.
- When this order line has a partial cancel and partial shipment, but the full amount is completed for this order line, the ship_dt is populated if the last item of this order line is shipped. The cncl_dt is populated if the last item of this order line is cancelled.
- f_svc_amt and f_svc_amt_lcl must keep their original values when any service return or service cancel occurs.
- The ship_dt and cncl_dt fields are mutually exclusive.
- Any cancel quantity that does not require a separate customer order line must be reflected in the cancelled quantity field f_cncl_qty.

Name	Description	Data Type/Bytes	Field Order	Required Field
CO_LINE_IDNT	The unique identifier of a customer order line.	CHARACTER (30)	1	Yes
CO_DAY_DT	The customer order creation date.	DATE	2	Yes
ITEM_IDNT	The unique identifier of an item.	CHARACTER (25)	3	Yes
SELLING_ITEM_IDNT	The unique identifier of a selling item.	CHARACTER (25)	4	Yes
HDR_MEDIA_IDNT	The unique identifier of the customer order header-level media.	CHARACTER (10)	5	Yes

Name	Description	Data Type/Bytes	Field Order	Required Field
LINE_MEDIA_IDNT	The unique identifier of the customer order line-level media.	CHARACTER (10)	6	Yes
BANNER_IDNT	The unique identifier of a banner. Banner represents the name of a retail company's subsidiary that is recognizable to the consumer or the name of the store as it appears on the catalog, web channel or brick and mortar store.	CHARACTER (4)	7	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER (10)	8	Yes
CSR_IDNT	The unique identifier of a customer service representative.	CHARACTER (30)	9	Yes
CUST_IDNT	The unique identifier of the customer.	CHARACTER (15)	10	Yes
CO_HDR_IDNT	The unique identifier of a customer order header.	CHARACTER (30)	11	Yes
DAY_DT	The transaction date when the customer order line is created or modified.	DATE	12	Yes
CO_SHIP_TO_CITY	The name of the city this order line is shipped to.	CHARACTER (120)	13	Yes
CO_SHIP_TO_ST_OR_P RVNC_CDE	The code of the state or province this order line is shipped to.	CHARACTER (3)	14	Yes
CO_SHIP_TO_CNTRY_C DE	The code of the country this order line is shipped to.	CHARACTER (10)	15	Yes
CO_SHIP_TO_PSTL_CD E	The zip code of the address this order line is shipped to (no additional 4 digits code for US address).	CHARACTER (30)	16	Yes

Name	Description	Data Type/Bytes	Field Order	Required Field
CARRIER_IDNT	The unique identifier of a carrier. A carrier is an entity that ships orders to customers.	CHARACTER (10)	17	Yes
CARRIER_SVC_IDNT	The unique identifier of a carrier service.	CHARACTER (10)	18	Yes
CO_DMND_STTS_IDNT	Predefined demand status codes to show the reason why the order line is cancelled.	CHARACTER (120)	19	Yes
REF_ITEM_IDNT	The unique identifier of the item that triggered the up-sell, cross-sell, substitute, or partial activity. This column will only be populated when the customer order line type is upsell, cross-sell, substitute, or partial, otherwise it will be -1.	CHARACTER (25)	20	Yes
CO_LINE_TYPE_IDNT	Identifies a customer order line type. The types can be up-sell, cross-sell, normal, return etc.	CHARACTER (120)	21	Yes
CO_PARTIAL_REASN_IDNT	Identifies the reason the partial order line was created.	CHARACTER (120)	22	Yes
CO_HOLD_EVENT_IDNT	Identifies the event why an order line is being held.	CHARACTER (120)	23	No
DROP_SHIP_IND	An indicator to identify if an item is shipped directly to the customer.	CHARACTER (1)	24	No
CO_GIFT_IND	An indicator to identify if the item on the order line is a gift.	CHARACTER (1)	25	No
CO_EST_DLVRY_DT	Estimated delivery date of the order line.	DATE	26	No
SHIP_DT	The date when the order	DATE	27	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field Order	Required Field
	line was fully shipped.			
CNCL_DT	The date when the order line was fully canceled.	DATE	28	No
F_CO_QTY	The quantity ordered.	NUMBER(12, 4)	29	No
F_RSV_QTY	The quantity reserved.	NUMBER(12, 4)	30	No
F_PICK_QTY	The quantity picked.	NUMBER(12, 4)	31	No
F_PICK_EOD_QTY	The quantity in pick status at the end of the day.	NUMBER(12, 4)	32	No
F_BO_QTY	The quantity backordered.	NUMBER(12, 4)	33	No
F_BO_EOD_QTY	The quantity in backorder status at the end of the day.	NUMBER(12, 4)	34	No
F_SHIP_QTY	The quantity shipped.	NUMBER(12, 4)	35	No
F_CNCL_QTY	The quantity cancelled.	NUMBER(12, 4)	36	No
F_CO_UNIT_RTL_AMT	The transaction unit price of the order line item expected to be paid by the customer, in primary currency.	NUMBER(18, 4)	37	No
F_CO_UNIT_RTL_AMT_LCL	The transaction unit price of the order line item expected to be paid by the customer, in local currency.	NUMBER(18, 4)	38	No
F_CO_MEDIA_UNIT_RTL_AMT	The media selling price of the order line item, in primary currency.	NUMBER(18, 4)	39	No
F_CO_MEDIA_UNIT_RTL_AMT_LCL	The media selling price of the order line item, in local currency.	NUMBER(18, 4)	40	No
F_ADDL_DLVR_Y_AMT	The additional shipping and handling charge	NUMBER(18, 4)	41	No

Name	Description	Data Type/Bytes	Field Order	Required Field
	applied to the order line, in primary currency.			
F_ADDL_DLVR_Y_AMT_LCL	The additional shipping and handling charge applied to the order line, in local currency.	NUMBER(18, 4)	42	No
F_CNCL_ADDL_DLVR_Y_AMT	The amount taken off from the original additional shipping and handling charge due to cancels, in primary currency.	NUMBER(18, 4)	43	No
F_CNCL_ADDL_DLVR_Y_AMT_LCL	The amount taken off from the original additional shipping and handling charge due to cancels, in local currency.	NUMBER(18, 4)	44	No
F_PRMTN_L_DSCNT_AMT	The total promotional discount applied to the order line, in primary currency.	NUMBER(18, 4)	45	No
F_PRMTN_L_DSCNT_AMT_LCL	The total promotional discount applied to the order line, in local currency.	NUMBER(18, 4)	46	No
F_CNCL_PRMTN_L_DSCNT_AMT	The amount taken off from the original promotional discount due to cancels, in primary currency.	NUMBER(18, 4)	47	No
F_CNCL_PRMTN_L_DSCNT_AMT_LCL	The amount taken off from the original promotional discount due to cancels, in local currency.	NUMBER(18, 4)	48	No
F_SVC_AMT	The sum of the service charges applied to the customer order service lines, in primary currency.	NUMBER(18, 4)	49	No
F_SVC_AMT_LCL	The sum of the service	NUMBER(18, 4)	50	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field Order	Required Field
	charges applied to the customer order service lines, in local currency.	4)		
F_CNCL_SVC_AMT	The amount taken off from the original service charge due to cancels, in primary currency.	NUMBER(18, 4)	51	No
F_CNCL_SVC_AMT_LCL	The amount taken off from the original service charge due to cancels, in local currency.	NUMBER(18, 4)	52	No
F_ADDL_DLVR_TAX_AMT	The additional shipping and handling charge tax applied to the order line, in primary currency.	NUMBER(18, 4)	53	No
F_ADDL_DLVR_TAX_AMT_LCL	The additional shipping and handling charge tax applied to the order line in local currency.	NUMBER(18, 4)	54	No
F_CNCL_ADDL_DLVR_TAX_AMT	The amount taken off from the original additional shipping and handling charge tax due to cancels, in primary currency.	NUMBER(18, 4)	55	No
F_CNCL_ADDL_DLVR_TAX_AMT_LCL	The amount taken off from the original additional shipping and handling charge tax due to cancels, in local currency.	NUMBER(18, 4)	56	No
F_MRCH_TAX_AMT	The merchandise tax applied to the order line, in primary currency.	NUMBER(18, 4)	57	No
F_MRCH_TAX_AMT_LCL	The merchandise tax applied to the order line, in local currency.	NUMBER(18, 4)	58	No
F_CNCL_MRCH_TAX_AMT	The amount taken off from the original merchandise tax due to cancels, in primary currency.	NUMBER(18, 4)	59	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_CNCL_MRCH_TAX_AMT_LCL	The amount taken off from the original merchandise tax due to cancels, in local currency.	NUMBER(18, 4)	60	No
F_SVC_TAX_AMT	The service tax applied to the order line, in primary currency.	NUMBER(18, 4)	61	No
F_SVC_TAX_AMT_LCL	The service tax applied to the order line, in local currency.	NUMBER(18, 4)	62	No
F_CNCL_SVC_TAX_AMT	The amount taken off from the original service tax due to cancels, in primary currency.	NUMBER(18, 4)	63	No
F_CNCL_SVC_TAX_AMT_LCL	The amount taken off from the original service charge tax due to cancels, in local currency.	NUMBER(18, 4)	64	No

coprmilnlddm.txt

Business rules:

- This data must be extracted from the source system after midnight, and only data created in the system before midnight must be extracted.
- This interface file cannot contain duplicate records for the combination of co_line_idnt, prmtn_dtl_idnt, co_hdr_idnt, and day_dt.
- This interface file contains only the new or changed information since the last extraction from the source system, one record per new or changed order line per day (only the customer order lines that include promotion data are sent through this interface file).
- If the promotion is at the order header level and cannot be broken down to the order line level (free shipping handling), column co_line_idnt and other key values should be populated with -1. In addition, column f_prmtn_dscnt_amt and f_prmtn_dscnt_amt_lcl must be populated (f_prmtn_l_dscnt_amt and f_prmtn_l_dscnt_amt_lcl will not be populated). If the promotion is at the order line level, column f_prmtn_l_dscnt_amt and f_prmtn_l_dscnt_amt_lcl must be populated, and column f_prmtn_dscnt_amt and f_prmtn_dscnt_amt_lcl must not be populated.
- When an order or order line gets canceled, the source system should send promotions under that order or order line with cancelled status 'C'.

Appendix A – Application programming interface (API) flat file specifications

- For service line promotions, the item_idnt in the interface file must be the non-merchandise item identifier for that service.
- The banner_idnt corresponding to the hdr_media_idnt and line_media_idnt must be the same.
- If a dimension identifier is required but is not available, a value of -1 is needed.
- This interface file follows the fact flat file interface layout standard.
- Only changes for the defined fields in the API specifications are considered.
- f_prmtn_dscnt_amt, f_prmtn_dscnt_amt_lcl, f_prmtn_l_dscnt_amt, f_prmtn_l_dscnt_amt_lcl must keep their original values when any promotional return or promotional cancel occurs. Return columns must be populated when returns happen. Cancel columns must be populated when cancels happen. These columns must be populated for the original order line's service line (as opposed to creating a new promotion line for a return).

Name	Description	Data Type/Bytes	Field Order	Required Field
CO_LINE_IDNT	The unique identifier of a customer order line.	CHARACTER(30)	1	Yes
CO_HDR_IDNT	The unique identifier of a customer order header.	CHARACTER(30)	2	Yes
PRMTN_DTL_IDNT	The identifier of the promotion detail.	CHARACTER(10)	3	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	4	Yes
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	5	Yes
SELLING_ITEM_IDNT	The unique identifier of a selling item.	CHARACTER(25)	6	Yes
HDR_MEDIA_IDNT	The unique identifier of the customer order header-level media.	CHARACTER(10)	7	Yes
LINE_MEDIA_IDNT	The unique identifier of the customer order line level media.	CHARACTER(10)	8	Yes
BANNER_IDNT	The unique identifier of a banner.	CHARACTER(4)	9	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	10	Yes
CSR_IDNT	The unique identifier of a customer service representative.	CHARACTER(30)	11	Yes
CUST_IDNT	The unique identifier of	CHARACTER	12	Yes

Name	Description	Data Type/Bytes	Field Order	Required Field
	the customer.	R(15)		
CO_SHIP_TO_CITY	The name of the city to which this order line is shipped.	CHARACTER(120)	13	Yes
CO_SHIP_TO_ST_OR_PRVNC_CDE	The code of the state or province to which this order line is shipped.	CHARACTER(3)	14	Yes
CO_SHIP_TO_CNTRY_CDE	The code of the country to which this order line is shipped.	CHARACTER(120)	15	Yes
CO_SHIP_TO_PSTL_CDE	The zip code of the address to which this order line is shipped (no additional four digit code for a US address).	CHARACTER(30)	16	Yes
CARRIER_IDNT	The unique identifier of a carrier. A carrier is an entity that ships orders to customers.	CHARACTER(10)	17	Yes
CARRIER_SVC_IDNT	The unique identifier of a carrier service.	CHARACTER(10)	18	Yes
PRMTN_TRIG_TYPE_IDNT	The unique identifier of the promotion trigger type. Valid values can be 'offer code', 'media code', and so on.	NUMBER(6)	19	Yes
PRMTN_TRIG_IDNT	The promotion trigger code identifier.	CHARACTER(25)	20	Yes
CO_HOLD_EVENT_IDNT	Identifies the event why an order line is being held.	CHARACTER(120)	21	Yes
CO_LINE_TYPE_IDNT	Identifies a customer order line type. The types can be up-sell, cross-sell, normal, return etc.	CHARACTER(120)	22	Yes
CO_HDR_STTS	The order header status.	CHARACTER(2)	23	Yes
CO_LINE_STTS	The order line status.	CHARACTER(2)	24	Yes
F_PRMTN_DSCNT_AM	The shipping and	NUMBER(1)	25	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field Order	Required Field
T	handling promotional discount applied to the entire customer order, in primary currency.	8,4)		
F_PRMTN_DSCNT_AMT_LCL	The total promotional discount applied to the order line, in primary currency.	NUMBER(18,4)	26	No
F_RTRN_PRMTN_DSCNT_AMT	The amount taken off from the original shipping and handling promotional discount due to returns, in primary currency.	NUMBER(18,4)	27	No
F_RTRN_PRMTN_DSCNT_AMT_LCL	The amount taken off from the original shipping and handling promotional discount due to returns, in local currency.	NUMBER(18,4)	28	No
F_PRMTN_L_DSCNT_AMT	The promotional discount amount applied to the customer order line in primary currency. This includes promotional discounts applied at the order header but were prorated to the order lines.	NUMBER(18,4)	29	No
F_PRMTN_L_DSCNT_AMT_LCL	The promotion discount amount applied to the customer order line in local currency. This includes promotional discounts applied at the order header but were prorated to the order lines.	NUMBER(18,4)	30	No
F_RTRN_PRMTN_L_DSCNT_AMT	The amount taken off from the original order line promotional discount due to returns, in primary currency.	NUMBER(18,4)	31	No
F_RTRN_PRMTN_L_DS	The amount taken off	NUMBER(18,4)	32	No

Name	Description	Data Type/Bytes	Field Order	Required Field
CNT_AMT_LCL	from the original order line promotional discount due to returns, in local currency.	8,4)		

cortrnlrlddm.txt

Business rules:

- This data must be extracted from the source system after midnight, and only data created in the system before midnight must be extracted.
- This interface file cannot contain duplication records for a combination of co_line_idnt and co_day_dt.
- This interface file contains only the previous day's new or changed return, replacement in, or exchange in customer order line information. There is one record per new or changed order line per day. Only customer order lines with a type of return, replacement in or exchange in should be sent through this interface file.
- day_dt must be populated with the order line create date when the return order line is still in "Pending Return" status. It must be populated with the return date or cancel date when the order line eventually becomes returned or cancelled
- The banner_idnt corresponding to the hdr_media_idnt and line_media_idnt must be the same.
- Only changes for the defined fields in the API specifications are considered.
- If a dimension identifier is required but is not available, a value of -1 is needed.
- This interface file follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
CO_LINE_IDNT	The unique identifier of a customer order line.	CHARACTER(30)	1	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	2	Yes
CO_LINE_DAY_DT	The unique identifier of the date the customer order line was created.	DATE	3	Yes
ITEM_IDNT	The unique identifier of the item being returned.	CHARACTER(25)	4	Yes
SELLING_ITEM_IDNT	The unique identifier of a selling item.	CHARACTER(25)	5	Yes
HDR_MEDIA_IDNT	The unique identifier of the customer order	CHARACTER(10)	6	Yes

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field Order	Required Field
	header-level media.			
LINE_MEDIA_IDNT	The unique identifier of the customer order line level media.	CHARACTER(10)	7	Yes
BANNER_IDNT	The unique identifier of the banner.	CHARACTER(4)	8	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	9	Yes
CO_DAY_DT	The customer order creation date.	DATE	10	Yes
CSR_IDNT	The unique identifier for the customer service representative facilitating the return.	CHARACTER(30)	11	Yes
CUST_IDNT	The unique identifier of the customer.	CHARACTER(15)	12	Yes
RTRN_REASN_IDNT	The unique identifier used to identify a return reason code. These codes should exist in the RMS CODE_DETAIL table under 'SARR' code type.	CHARACTER(6)	13	Yes
DISPO_IDNT	The unique identifier used to identify a disposition code.	CHARACTER(6)	14	Yes
CO_HDR_IDNT	The unique identifier of the customer order to which this return line belongs.	CHARACTER(30)	15	Yes
CO_LINE_TYPE_IDNT	Identifies a customer order line type. The types can be up-sell, cross-sell, normal, return etc.	CHARACTER(120)	16	Yes
RTRN_STTS_IDNT	The status of the return order line such as pending, return, cancel.	CHARACTER(120)	17	Yes
DROP_SHIP_IND	An indicator to identify if an item is shipped directly to the customer.	CHARACTER(1)	18	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_CO_RTRN_QTY	The quantity involved in return, exchange return or replacement return transactions.	NUMBER(12,4)	19	No
F_RFND_DLVRY_AMT	The total shipping and handling charge refunded to customers, in primary currency.	NUMBER(18,4)	20	No
F_RFND_DLVRY_AMT_LCL	The total shipping and handling charge refunded to customers, in primary currency.	NUMBER(18,4)	21	No
F_CO_RTRN_RTL_AMT	The transaction retail amount of items requested to be refunded by the customer in the return or exchange transaction, in primary currency	NUMBER(18,4)	22	No
F_CO_RTRN_RTL_AMT_LCL	The transaction retail amount of items requested to be refunded by the customer in the return or exchange transaction, in local currency	NUMBER(18,4)	23	No
F_RTRN_PRMTN_L_DS_CNT_AMT	The amount taken off from the original promotional discount due to returns, in primary currency.	NUMBER(18,4)	24	No
F_RTRN_PRMTN_L_DS_CNT_AMT_LCL	The amount taken off from the original promotional discount due to returns, in local currency.	NUMBER(18,4)	25	No
F_RTRN_ADDL_DLVRY_AMT	The amount taken off from the original additional shipping and handling charge due to returns, in primary currency.	NUMBER(18,4)	26	No
F_RTRN_ADDL_DLVRY	The amount taken off	NUMBER(1	27	No

Name	Description	Data Type/Bytes	Field Order	Required Field
_AMT_LCL	from the original additional shipping and handling charge due to returns, in local currency.	8,4)		
F_RTRN_SVC_AMT	The amount taken off from the original service charge due to returns, in primary currency.	NUMBER(18,4)	28	No
F_RTRN_SVC_AMT_LCL	The amount taken off from the original service charge due to returns, in local currency.	NUMBER(18,4)	29	No
F_RTRN_MRCH_TAX_AMT	The amount taken off from the original merchandise tax due to returns, in primary currency.	NUMBER(18,4)	30	No
F_RTRN_MRCH_TAX_AMT_LCL	The amount taken off from the original merchandise tax due to returns, in local currency.	NUMBER(18,4)	31	No
F_RTRN_ADDL_DLVR_TAX_AMT	The amount taken off from the original additional shipping and handling charge tax due to returns, in primary currency.	NUMBER(18,4)	32	No
F_RTRN_ADDL_DLVR_TAX_AMT_LCL	The amount taken off from the original additional shipping and handling charge tax due to returns, in local currency.	NUMBER(18,4)	33	No
F_RTRN_SVC_TAX_AMT	The amount taken off from the original service charge tax due to returns, in primary currency.	NUMBER(18,4)	34	No
F_RTRN_SVC_TAX_AMT_LCL	The amount taken off from the original service charge tax due to	NUMBER(18,4)	35	No

Name	Description	Data Type/Bytes	Field Order	Required Field
	returns, in local currency.			

coshptodm.txt

Business rules:

- This interface file follows the dimension flat file interface layout standard.
- This interface file cannot contain duplicate records for the combination of cust_ship_to_city, cust_ship_to_county, cust_ship_to_st_or_prvnc_cde, cust_ship_to_cntry_cde and cust_ship_to_pstl_cde.
- This interface file contains only the new customer order ship to addresses for the day.
- If a dimension identifier is required but is not available, a value of -1 is needed.

Name	Description	Data Type/Bytes	Field order	Required field
CO_SHIP_TO_CITY	The customer order ship-to city.	CHARACTER (120)	1	Yes
CO_SHIP_TO_COUNTY	The customer order ship-to county.	CHARACTER (120)	2	Yes
CO_SHIP_TO_ST_OR_P RVNC_CDE	The customer order ship-to state or province code.	CHARACTER (3)	3	Yes
CO_SHIP_TO_ST_OR_P RVNC_DESC	The customer order ship-to state or province description.	CHARACTER (120)	4	No
CO_SHIP_TO_CNTRY_CDE	The customer order ship-to country code.	CHARACTER (10)	5	Yes
CO_SHIP_TO_PSTL_CDE	The customer order ship-to postal code.	CHARACTER (30)	6	Yes

cosvcliddm.txt

Business rules:

- This interface file cannot contain duplicate records for a co_sl_idnt, co_line_idnt, day_dt combination.
- This interface file contains only the current day's new or updated Value Added Service Line information, one record per new or updated service line per day. Do not include cancelled service lines.
- svc_colr_cde contains 'Multi-Color' if a service line has more than one color.
- svc_font_cde contains Multi-Font if a service line has more than one font.

Appendix A – Application programming interface (API) flat file specifications

- `svc_style_cde` for Personalization(P), Monogramming(M), Gift wrap(W), Care Card(C) and Gift card(G) must match the code for these four service types on the `CDE_DTL_COM_DM` table.
- `loc_idnt` is the unique identifier of the virtual store that sells the items.
- `f_svc_amt`, `f_svc_amt_lcl`, and `f_svc_qty` must keep their existing values when any service return or service cancel occurs. Only return columns are populated when returns occur, and only cancel columns are populated when cancels occur. These fields remain in the same record rather than in a separate record.
- All the quantity and amount fields are transactional quantities and amounts. These transactional quantities and amounts must contain the delta values of the previous value and current value. They must be filled in when a transaction occurred.
- The `banner_idnt` corresponding to the `hdr_media_idnt` and `line_media_idnt` must be the same.
- If a dimension identifier is required but is not available, a value of -1 is needed.
- Only changes for the defined fields in the API specifications are considered.
- This interface file follows the fact flat file interface layout standard.
- This data must be extracted from the source system after midnight, and only data created in the system before midnight must be extracted.

Name	Description	Data Type/Bytes	Field Order	Required Field
CO_SL_IDNT	The identifier for a customer order service line.	CHARACTER (30)	1	Yes
CO_LINE_IDNT	The unique identifier of a customer order line.	CHARACTER (30)	2	Yes
DAY_DT	The calendar day on which the customer order service line is created or modified.	DATE	3	Yes
CO_DAY_DT	The customer order creation date.	DATE	4	Yes
ITEM_IDNT	The unique identifier of an item.	CHARACTER (25)	5	Yes
SELLING_ITEM_IDNT	The unique identifier of a selling item.	CHARACTER (25)	6	Yes
HDR_MEDIA_IDNT	The unique identifier of the customer order header-level media.	CHARACTER (10)	7	Yes
LINE_MEDIA_IDNT	The unique identifier of the customer order line level media.	CHARACTER (10)	8	Yes

Name	Description	Data Type/Bytes	Field Order	Required Field
BANNER_IDNT	The unique identifier of a banner.	CHARACTER (4)	9	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER (10)	10	Yes
CSR_IDNT	The unique identifier of a customer service representative.	CHARACTER (30)	11	Yes
CUST_IDNT	The unique identifier of the customer placing the order.	CHARACTER (15)	12	Yes
SVC_STYLE_CDE	The code used to identify a service line style.	CHARACTER (120)	13	Yes
SVC_COLR_CDE	The code used to identify a service line color.	CHARACTER (120)	14	Yes
SVC_FONT_CDE	The code used to identify a service line font.	CHARACTER (120)	15	Yes
SVC_TYPE_CDE	The code used to identify a type of value added service.	CHARACTER (120)	16	Yes
CO_LINE_TYPE_IDNT	Identifies a customer order line type. The types can be up-sell, cross-sell, normal, return etc.	CHARACTER (120)	17	Yes
F_SVC_AMT	The service charge applied in primary currency.	NUMBER(18, 4)	18	No
F_SVC_AMT_LCL	The service charge applied in local currency.	NUMBER(18, 4)	19	No
F_RTRN_SVC_AMT	The amount taken off from the original service charge due to returns, in primary currency.	NUMBER(18, 4)	20	No
F_RTRN_SVC_AMT_LCL	The amount taken off from the original service charge due to returns, in local currency.	NUMBER(18, 4)	21	No
F_CNCL_SVC_AMT	The amount taken off from the original service charge due to cancels, in primary currency.	NUMBER(18, 4)	22	No
F_CNCL_SVC_AMT_LCL	The amount taken off	NUMBER(18, 4)	23	No

Name	Description	Data Type/Bytes	Field Order	Required Field
L	from the original service charge due to cancels, in local currency.	4)		
F_SVC_QTY	The service quantity ordered	NUMBER(12, 4)	24	No
F_RTRN_SVC_QTY	The quantity taken off from the original service quantity due to returns.	NUMBER(12, 4)	25	No
F_CNCL_SVC_QTY	The quantity taken off from the original service quantity due to cancels.	NUMBER(12, 4)	26	No

cosvclilsg_pre.txt

Business rules:

- This interface file cannot contain duplicate records for a co_sl_idnt, co_line_idnt combination.
- This interface file follows the fact flat file interface layout standard.
- This data must be extracted from the source system after midnight, and only data created in the system before midnight must be extracted.
- This interface file contains only the current day's new or updated Value Added Service Line information, one record per new or updated service line per day. Do not include cancelled service lines.
- svc_colr_cde contains 'Multi-Color' if a service line has more than one color.
- svc_font_cde contains Multi-Font if a service line has more than one font.
- svc_style_cde for Personalization(P), Monogramming(M), Gift wrap(W), Care Card(C) and Gift card(G) must match the code for these four service types on the CDE_DTL_COM_DM table.
- If a dimension identifier is required but is not available, a value of -1 is needed.
- loc_idnt is the unique identifier of the virtual store that sells the items.
- f_svc_amt, f_svc_amt_lcl, and f_svc_qty must keep their existing values when any service return or service cancel occurs. Only return columns are populated when returns occur, and only cancel columns are populated when cancels occur. These fields remain in the same record rather than in a separate record.
- The banner_idnt corresponding to the hdr_media_idnt and line_media_idnt must be the same.
- Fields must be populated with the latest value for the field as of the end of the day.

Name	Description	Data Type/Bytes	Field Order	Required Field
CO_SL_IDNT	The identifier for a customer order service line.	CHARACTER(30)	1	Yes
CO_LINE_IDNT	The unique identifier of a customer order line.	CHARACTER(30)	2	Yes
DAY_DT	The calendar day when the customer order service line is created or modified.	DATE	3	Yes
CO_DAY_DT	The customer order creation date.	DATE	4	Yes
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	5	Yes
SELLING_ITEM_IDNT	The unique identifier of a selling item.	CHARACTER(25)	6	Yes
HDR_MEDIA_IDNT	The unique identifier of the customer order header-level media.	CHARACTER(10)	7	Yes
LINE_MEDIA_IDNT	The unique identifier of the customer order line-level media.	CHARACTER(10)	8	Yes
BANNER_IDNT	The unique identifier of a banner. Banner represents the name of a retail company's subsidiary that is recognizable to the consumer or the name of the store as it appears on the catalog, web channel or brick and mortar store.	CHARACTER(40)	9	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	10	Yes
CSR_IDNT	The unique identifier of a customer service representative.	CHARACTER(30)	11	Yes
CUST_IDNT	The unique identifier of the customer.	CHARACTER(15)	12	Yes
SVC_STYLE_IDNT	The identifier of the service style.	CHARACTER(120)	13	Yes

Name	Description	Data Type/Bytes	Field Order	Required Field
SVC_COLR_IDNT	The identifier of the service color.	CHARACTER(120)	14	Yes
SVC_FONT_IDNT	The identifier of the service font.	CHARACTER(120)	15	Yes
SVC_TYPE_IDNT	The identifier of the service type.	CHARACTER(120)	16	Yes
CO_LINE_TYPE_IDNT	Identifies a customer order line type. The types can be up-sell, cross-sell, normal, return etc.	CHARACTER(120)	17	Yes
F_SVC_AMT	The service charge applied in primary currency.	NUMBER(18,4)	18	No
F_SVC_AMT_LCL	The service charge applied in local currency.	NUMBER(18,4)	19	No
F_RTRN_SVC_AMT	The amount taken off from the original service charge due to returns, in primary currency.	NUMBER(18,4)	20	No
F_RTRN_SVC_AMT_LCL	The amount taken off from the original service charge due to returns, in local currency.	NUMBER(18,4)	21	No
F_CNCL_SVC_AMT	The amount taken off from the original service charge due to cancels, in primary currency.	NUMBER(18,4)	22	No
F_CNCL_SVC_AMT_LCL	The amount taken off from the original service charge due to cancels, in local currency.	NUMBER(18,4)	23	No
F_SVC_QTY	The service quantity ordered.	NUMBER(12,4)	24	No
F_RTRN_SVC_QTY	The quantity taken off from the original service quantity due to returns.	NUMBER(18,4)	25	No
F_CNCL_SVC_QTY	The quantity taken off from the original service quantity due to cancels.	NUMBER(12,4)	26	No

crnycddm.txt

Business rules:

- This interface file contains currency code information.
- This interface file cannot contain duplicate records for a crncy_cde_idnt.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field order	Required field
CRNCY_CDE_IDNT	The unique identifier of the currency code.	CHARACTER(10)	1	Yes
CRNCY_CDE_DESC	The description of local currency code. E.g. description for USD = US Dollar.	CHARACTER(120)	2	Yes

crrdm.txt

Business rules:

- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.
- This interface file cannot contain duplicate records for a carrier_idnt.

Name	Description	Data Type/Bytes	Field Order	Required Field
CARRIER_IDNT	The unique identifier of a carrier. A carrier is an entity that ships orders to customers.	CHARACTER (10)	1	Yes
CARRIER_DESC	Description for the carrier.	CHARACTER (120)	2	No

crrsvcdm.txt

Business rules:

- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.
- This interface file cannot contain duplicate records for a carrier_svc_idnt.

Name	Description	Data Type/Bytes	Field Order	Required Field
CARRIER_SVC_IDNT	The unique identifier of a carrier service.	CHARACTER(10)	1	Yes
CARRIER_SVC_DESC	Description for the carrier service.	CHARACTER(120)	2	No

csrdm.txt

Business rules:

- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.
- This interface file cannot contain duplicate records for a csr_idnt.

Name	Description	Data Type/Bytes	Field Order	Required Field
CSR_IDNT	The unique identifier of a customer service representative.	CHARACTER(30)	1	Yes
CALL_CTR_IDNT	The unique identifier of a call center.	CHARACTER(10)	3	Yes
CSR_NAME	The name of the customer service representative.	CHARACTER(120)	2	No

cstislddm.txt

Business rules:

- This interface file contains cost information for an item, supplier, and location combination on a given day.
- This interface file cannot contain duplicate transactions for an item_idnt, loc_idnt, supp_idnt and day_dt combination.
- This interface file follows the fact flat file interface layout standard.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes

Name	Description	Data Type/Bytes	Field order	Required field
SUPP_IDNT	The unique identifier of a supplier.	CHARACTER(10)	3	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	4	Yes
F_BASE_COST_AMT	The cost valuation in primary currency	NUMBER(18,4)	5	No
F_BASE_COST_AMT_LCL	The cost valuation in local currency	NUMBER(18,4)	6	No

custacctdm.txt

Business rules:

- This interface file contains customer and account number relationships. This table allows account numbers to be linked to specific customers. In the case that two customers have the same account, only the primary account holder can be in this file.
- This interface file cannot contain duplicate records for a cust_idnt, acct_nbr, acct_type_idnt combination.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
CUST_IDNT	The unique identifier of the customer.	CHARACTER(15)	1	Yes
ACCNT_NBR	The customer account number(i.e. from a checking, credit card or loyalty card account).	CHARACTER(30)	2	Yes
ACCNT_TYPE_IDNT	The unique identifier of an account type.	CHARACTER(6)	3	Yes
ACCNT_TYPE_DESC	The description of account type (i.e., checking, VISA, Master Card ...).	CHARACTER(120)	4	Yes
RECD_TYPE	The type code of the record. Valid values are 'I' for	CHARACTER(1)	5	Yes

Name	Description	Data Type/Bytes	Field Order	Required Field
	insert, 'U' for update, and 'X' for delete.			

custaccttypedm.txt

Business rules:

- This interface file cannot contain duplicate records for a customer acct_type_idnt.
- This interface file contains the complete snapshot of active information.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
ACCNT_TYPE_IDNT	The unique identifier of an account type.	CHARACTER(6)	1	Yes
ACCNT_TYPE_DESC	The description of account type (i.e., checking, VISA, Master Card ...).	CHARACTER(120)	2	No
ACCNT_GRP_IDNT	The unique identifier of an account group.	CHARACTER(6)	3	Yes
ACCNT_GRP_DESC	The description of account group (i.e., credit cards, loyalty cards ...).	CHARACTER(120)	4	No

custclstrdm.txt

Business rules:

- This interface file contains all customer clusters and their descriptions. The data must come from an external source.
- This interface file cannot contain duplicate records for a cust_clstr_key, cust_idnt combination.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
CUST_CLSTR_KEY	Surrogate key to identify a customer cluster.	NUMBER(4)	1	Yes
CUST_CLSTR_DESC	The reference name for this cluster of customers.	CHARACTER(30)	2	No
CUST_IDNT	The unique identifier of	CHARACTER(15)	3	Yes

Name	Description	Data Type/Bytes	Field Order	Required Field
	the customer.			

custclstrmdm.txt

Business rules:

- This interface file defines the associations between tracking level items and customer clusters.
- This interface file cannot contain duplicate records for a cust_clstr_key, item_idnt combination.

Name	Description	Data Type/Bytes	Field Order	Required Field
CUST_CLSTR_KEY	Surrogate key to identify a customer cluster.	NUMBER(4)	1	yes
ITEM_IDNT	The unique identifier of an item.	CHARACTER (25)	2	Yes

custdm.txt

Business rules:

- This interface file contains customer information.
- This interface file cannot contain duplicate records for a customer identifier.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains only the latest information of the current day's new customers or updated customers in the source system.
- Only changes for the defined fields in the API specifications are considered.

Name	Description	Data Type/Bytes	Field Order	Required Field
CUST_IDNT	The unique identifier of the customer.	CHARACTER(15)	1	Yes
CUST_FIRST_NAME	First name of customer.	CHARACTER(120)	2	Yes
CUST_LAST_NAME	Last name of customer.	CHARACTER(120)	3	Yes
CUST_MIDDLE_NAME	The middle initial of customer.	CHARACTER(120)	4	No
CUST_TITLE	The label or	CHARACTER(12)	5	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field Order	Required Field
	heading preceding an individual's name. For example: Mr., Ms., Mrs., Dr.			
CUST_SUFFIX	The label following an individual's name. For example: Jr. or Sr.	CHARACTER(12)	6	No
CUST_ADDR_1	The customer's address line 1, for example, street address.	CHARACTER(255)	7	No
CUST_ADDR_2	The customer's address line 2, for example, suite or apartment number.	CHARACTER(255)	8	No
CUST_ADDR_3	The customer's address line 3, for example, company name.	CHARACTER(255)	9	No
CUST_CITY	The customer's city.	CHARACTER(120)	10	No
CUST_COUNTY	The customer's county.	CHARACTER(120)	11	No
CUST_ST_OR_PRVNC_CDE	The customer's state or province code.	CHARACTER(3)	12	No
CUST_ST_OR_PRVNC_DESC	The customer's state or province description.	CHARACTER(120)	13	No
CUST_CNTRY_CDE	The customer's country code.	CHARACTER(10)	14	No
CUST_PSTL_CDE	The customer's postal code.	CHARACTER(30)	15	No
CUST_PSTL_CDE_4	The customer's postal code	CHARACTER(4)	16	No

Name	Description	Data Type/Bytes	Field Order	Required Field
	extension.			
CUST_MAIL_ALLWD_IND	Indicates if marketing information can be sent to the customer.	CHARACTER(1)	17	No
CUST_EMAIL	The email address for the customer.	CHARACTER(100)	18	No
CUST_DT_OF_BIRTH	The date of birth of the customer.	DATE	19	No
CUST_OCCPN	The job which the customer holds.	CHARACTER(64)	20	No
CUST_INCOME	The customer's annual income.	NUMBER(18,4)	21	No
CUST_HH_SIZE	The number of people within one household.	NUMBER(2)	22	No
CUST_CHILD_QTY	The number of children the customer has.	NUMBER(2)	23	No
CUST_MARITAL_CDE	The code used to identify a marital status.	CHARACTER(12)	24	No
CUST_MARITAL_DESC	The marital description of the customer.	CHARACTER(120)	25	No
CUST_GENDER_CDE	The code used to identify a gender.	CHARACTER(12)	26	No
CUST_GENDER_DESC	The gender description of the customer.	CHARACTER(120)	27	No
CUST_ETHNIC_CDE	The code assigned to a customer to identify the ethnicity of the customer.	CHARACTER(12)	28	No
CUST_ETHNIC_DESC	The ethnic	CHARACTER(120)	29	No

Name	Description	Data Type/Bytes	Field Order	Required Field
	background of the customer.			
CUST_STTS_CDE	The code assigned to a customer to identify the status of a customer.	CHARACTER(15)	30	No
CUST_STTS_DESC	The status of a customer. For example: active or inactive.	CHARACTER(160)	31	No
CUST_TAX_IDNT	The unique identifier given to a customer by government for taxing purposes.	CHARACTER(30)	32	No
CUST_LEGAL_IDNT	The unique identifier given to a customer by government to identify the customer's legal identity. For example a Social Security Number.	CHARACTER(20)	33	No
CUST_LEGAL_DESC	The type of legal identity, such a Social Security Number.	CHARACTER(160)	34	No
CUST_ST_IDNT	The unique identifier given to a customer by a state government agency. Often this is a driver's license number.	CHARACTER(20)	35	No
CUST_TYPE_IDNT	The unique identifier used to determine the type of customer.	CHARACTER(15)	36	No

Name	Description	Data Type/Bytes	Field Order	Required Field
CUST_TYPE_DESC	The description of the type of customer. For example: employee, distributor, etc	CHARACTER(160)	37	No
CUST_EXT_STRAT_IDNT	The unique identifier used to determine how a customer was obtained.	CHARACTER(15)	38	No
RECD_TYPE	The type code of the record. Valid values are 'I' for insert, 'U' for update, and 'X' for delete.	CHARACTER(1)	39	Yes

dpctdm.txt

Business rules:

- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.
- This interface file cannot contain duplicate records for a dpct_idnt.
- This interface file contains depiction code information.

Name	Description	Data Type/Bytes	Field Order	Required Field
DPCT_IDNT	The unique identifier of a depiction. A depiction identifies the creative representation that was used to present a selling item or group of selling items to the customer within a media.	CHARACTER(25)	1	Yes
DPCT_DESC	The description of the depiction.	CHARACTER(120)	2	No

emptydm.txt

Business rules:

- This interface file contains the employee data.
- This interface file cannot contain duplicate records for an `emply_idnt`.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
EMPLY_IDNT	The unique identifier of the employee.	CHARACTER(10)	1	Yes
EMPLY_NAME	The name of the employee.	CHARACTER(120)	2	Yes
EMPLY_ROLE	Indicates the type of position the employee holds. 'C'ashier, 'S'alesperson, 'O'ther.	CHARACTER(1)	3	Yes

exchngratedm.txt

Business rules:

- This interface file contains currency exchange rate information.
- This interface file cannot contain duplicate records for a `crncy_cde_idnt`, `day_dt` combination.
- This interface file follows the fact flat file interface layout standard.
- This interface file contains only the current day's new or changed information.

Name	Description	Data Type/Bytes	Field Order	Required Field
CRNCY_CDE_IDNT	The unique identifier of the currency code.	CHARACTER(10)	1	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	2	Yes
F_EXCHNG_RATE	The current exchange rate for the currency based on the primary currency.	NUMBER(18,4)	3	No

geocdedm.txt

Business rules:

- This interface file contains the different types of geographical codes.

- This interface file cannot contain duplicate records for a geo_cde_idnt.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
GEO_CDE_IDNT	The unique Identifier for a geographic area.	CHARACTER(10)	1	Yes
GEO_CDE_DESC	The description for a geographic area.	CHARACTER(30)	2	No
GEO_AGE	The average age of the people living within the geographic area.	NUMBER(4,1)	3	No
GEO_ANCESTRY_CDE	The ancestry code for a geographic area.	CHARACTER(4)	4	No
GEO_ANCESTRY_CDE_DESC	The ancestry code description for a geographic area.	CHARACTER(30)	5	No
GEO_AUTO_AVAIL_NBR	The auto available number for a geographic area.	NUMBER(3,1)	6	No
GEO_COMMUTE_TIME	The average commute time for a geographic area.	NUMBER(5,2)	7	No
GEO_EDU_LVL_CDE	The cde identifying the average education level for a geographic	CHARACTER(4)	8	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field Order	Required Field
	area.			
GEO_EDU_LVL_CDE_DESC	The average education level for a geographic area.	CHARACTER(30)	9	No
GEO_FAMILY_TYPE_CDE	The family type code for a geographical area	CHARACTER(4)	10	No
GEO_FAMILY_TYPE_CDE_DESC	The description of the family type code.	CHARACTER(30)	11	No
GEO_HOME_NBR_ROOMS	The average number of rooms per home.	NUMBER(4,1)	12	No
GEO_HOUSEHOLD_INCOME	The average household income in a geographical area.	NUMBER(15)	13	No
GEO_HOUSING_VALUE	The average house value for a geographic area.	NUMBER(15)	14	No
GEO_INDUSTRY_CDE	The code for the type of industry in a geographical area.	CHARACTER(4)	15	No
GEO_INDUSTRY_CDE_DESC	The description of the industry code.	CHARACTER(30)	16	No
GEO_MALE_TO_FEMALE_RAT	The male to female ratio for a geographic	NUMBER(12,4)	17	No

Name	Description	Data Type/Bytes	Field Order	Required Field
	area.			
GEO_PER_CAPITA_INCOME	Per capita income for a geographic area.	NUMBER(15)	18	No
GEO_PERSONS_TOT	The total number of people in a geographical area.	NUMBER(12)	19	No
GEO_POVERTY_TOT	The total number of people in poverty for a geographic area.	NUMBER(9)	20	No
GEO_RENT_TO_OWN_RAT	The ratio of number of people who rent to the number of people who own houses for a geographic area.	NUMBER(12,4)	21	No
GEO_RETIREMENT_INCOME	The average retirement income for a geographical area.	NUMBER(15)	22	No
GEO_URBAN_TO_RURAL_RAT	The urban to rural ratio for a geographical area.	NUMBER(12,4)	23	No
GEO_YR_HOME_BUILT	The average year a home was built in a geographic area.	NUMBER(4)	24	No

invilddm.txt

Business rules:

- This interface file contains end of day inventory levels and status for an item and location combination on a given day.
- This interface file cannot contain duplicate records for an item_idnt, loc_idnt, day_dt combination.
- It is not possible to have a different prod_seasn_key for the same item, loc and day combination. Therefore, the prod_seasn_key is not part of a primary key for any facts on the item, loc and day level. With the aggregation, it is possible to have a different prod_seasn_key at the subclass level for the same loc and day combination, or at the week level for the same item and loc combination. Therefore, the prod_seasn_key is part of the primary key for facts at the subclass and/or the week level.
- This interface file follows the fact flat file interface layout standard.
- This interface file contains only the current day's new or changed information.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	3	Yes
LOC_TYPE_CDE	The code that indicates whether the location is a store or warehouse.	CHARACTER(2)	4	Yes
RTL_TYPE_CDE	The price type ('R'egular, 'P'romotion, 'C'learance).	CHARACTER(2)	5	Yes
F_I_SOH_QTY	The total quantity of inventory on hand.	NUMBER(12,4)	6	No

Name	Description	Data Type/Bytes	Field order	Required field
F_I_SOH_COST_AMT	The extended cost amount of inventory in stock in primary currency. The product of the weighted average cost in primary currency and the current stock on hand quantity.	NUMBER(18,4)	7	No
F_I_SOH_COST_AMT_LCL	The extended cost amount of inventory in stock in local currency. The product of the weighted average cost in local currency and the current stock on hand quantity.	NUMBER(18,4)	8	No
F_I_SOH_RTL_AMT	The extended retail amount of inventory in stock in primary currency. The product of the unit retail in primary currency and the current stock on hand quantity.	NUMBER(18,4)	9	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_I_SOH_RTL_AMT_LCL	The extended retail amount of inventory in stock in local currency. The product of the unit retail in local currency and the current stock on hand quantity.	NUMBER(18,4)	10	No
F_I_ON_ORD_QTY	The quantity of inventory on order.	NUMBER(12,4)	11	No
F_I_ON_ORD_COST_AMT	The extended cost amount of inventory on order in primary currency. The product of the order unit cost in primary currency and the current on order quantity.	NUMBER(18,4)	12	No
F_I_ON_ORD_COST_AMT_LCL	The extended cost amount of inventory on order in local currency. The product of the order unit cost in local currency and the current on order quantity.	NUMBER(18,4)	13	No

Name	Description	Data Type/Bytes	Field order	Required field
F_I_ON_ORD_RTL_AMT	The extended retail amount of inventory on order in primary currency. The product of the order unit retail in primary currency and the current on order quantity.	NUMBER(18,4)	14	No
F_I_ON_ORD_RTL_AMT_LCL	The extended retail amount of inventory on order in local currency. The product of the order unit retail in local currency and the current on order quantity.	NUMBER(18,4)	15	No
F_I_IN_TRNST_QTY	The total quantity of inventory in transit.	NUMBER(12,4)	16	No
F_I_IN_TRNST_COST_AMT	The extended cost amount of inventory in transit in primary currency. The product of the weighted average cost in primary currency and the current in transit quantity.	NUMBER(18,4)	17	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_I_IN_TRNST_COST_AMT_LCL	The extended cost amount of inventory in transit in local currency. The product of the weighted average cost in local currency and the current in transit quantity.	NUMBER(18,4)	18	No
F_I_IN_TRNST_RTL_AMT	The extended retail amount of inventory in transit in primary currency. The product of the unit retail in primary currency and the current in transit quantity.	NUMBER(18,4)	19	No
F_I_IN_TRNST_RTL_AMT_LCL	The extended retail amount of inventory in transit in local currency. The product of the unit retail in local currency and the current in transit quantity.	NUMBER(18,4)	20	No

Name	Description	Data Type/Bytes	Field order	Required field
F_I_ALLOC_RSV_QTY	The allocated reserved quantity. The warehouse-to-store reserved quantity, composed of reserved quantity for allocations and the reserved quantity for transfers from warehouse to store.	NUMBER(12,4)	21	No
F_I_ALLOC_RSV_COST_AMT	The allocated reserved extended cost amount in primary currency. The product of the weighted average cost in primary currency and the current allocated reserved quantity.	NUMBER(18,4)	22	No
F_I_ALLOC_RSV_COST_AMT_LCL	The allocated reserved extended cost amount in local currency. The product of the weighted average cost in local currency and the current allocated reserved quantity.	NUMBER(18,4)	23	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_I_ALLOC_RSV_RTL_AMT	The allocated reserved extended retail amount in primary currency. The product of the unit retail in primary currency and the current allocated reserved quantity.	NUMBER(18,4)	24	No
F_I_ALLOC_RSV_RTL_AMT_LCL	The allocated reserved extended retail amount in local currency. The product of the unit retail in local currency and the current allocated reserved quantity.	NUMBER(18,4)	25	No
F_I_TRNSFR_RSV_QTY	The transfer reserved quantity. The store-to-store reserved quantity, composed of the quantity of transfers from store to store that have not been shipped.	NUMBER(12,4)	26	No

Name	Description	Data Type/Bytes	Field order	Required field
F_I_TRNSFR_RSV_COST_AMT	The transfer reserved extended cost amount in primary currency. The product of the weighted average cost in primary currency and the current transfer reserved quantity.	NUMBER(18,4)	27	No
F_I_TRNSFR_RSV_COST_AMT_LCL	The transfer reserved extended cost amount in local currency. The product of the weighted average cost in local currency and the current transfer reserved quantity.	NUMBER(18,4)	28	No
F_I_TRNSFR_RSV_RTL_AMT	The transfer reserved extended retail amount in primary currency. The product of the unit retail in primary currency and the current transfer reserved quantity.	NUMBER(18,4)	29	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_I_TRNSFR_RSV_RTL_AMT_LCL	The transfer reserved extended retail amount in local currency. The product of the unit retail in local currency and the current transfer reserved quantity.	NUMBER(18,4)	30	No
F_I_REPL_ACTV_FLAG	Flag to indicate if end date of this record's time period is within the active and inactive dates for replenishment.	CHARACTER(1)	31	No
F_I_REPL_CALC_MTHD_CDE	This column holds the replenishment method code value.	CHARACTER(2)	32	No
F_I_MIN_SOH_QTY	The minimum stock on hand quantity.	NUMBER(12,4)	33	No

Name	Description	Data Type/Bytes	Field order	Required field
F_I_MIN_SOH_COST_AMT	The extended cost amount of minimum stock on hand in primary currency. The product of the average weighted cost in primary currency and the current minimum stock on hand quantity.	NUMBER(18,4)	34	No
F_I_MIN_SOH_COST_AMT_LCL	The extended cost amount of minimum stock on hand in local currency. The product of the average weighted cost in local currency and the current minimum stock on hand quantity.	NUMBER(18,4)	35	No
F_I_MIN_SOH_RTL_AMT	The extended retail amount of minimum stock on hand in primary currency. The product of the unit retail in primary currency and the current minimum stock on hand quantity.	NUMBER(18,4)	36	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_I_MIN_SOH_RTL_AMT_LCL	The extended retail amount of minimum stock on hand in local currency. The product of the unit retail in local currency and the current minimum stock on hand quantity.	NUMBER(18,4)	37	No
F_I_MAX_SOH_QTY	The maximum stock on hand quantity.	NUMBER(12,4)	38	No
F_I_MAX_SOH_COST_AMT	The extended cost amount of maximum stock on hand in primary currency. The product of the average weighted cost in primary currency and the current maximum stock on hand quantity.	NUMBER(18,4)	39	No

Name	Description	Data Type/Bytes	Field order	Required field
F_I_MAX_SOH_COST_AMT_LCL	The extended cost amount of maximum stock on hand in local currency. The product of the average weighted cost in local currency and the current maximum stock on hand quantity.	NUMBER(18,4)	40	No
F_I_MAX_SOH_RTL_AMT	The extended retail amount of maximum stock on hand in primary currency. The product of the unit retail in primary currency and the current maximum stock on hand quantity.	NUMBER(18,4)	41	No
F_I_MAX_SOH_RTL_AMT_LCL	The extended retail amount of maximum stock on hand in local currency. The product of the unit retail in local currency and the current maximum stock on hand quantity.	NUMBER(18,4)	42	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_I_INCR_PCT	The replenishment incremental percentage or multiple value.	NUMBER(12,4)	43	No
F_I_COST_AMT	The weighted average cost for stock in primary currency.	NUMBER(18,4)	44	No
F_I_COST_AMT_LCL	The weighted average cost for stock in local currency.	NUMBER(18,4)	45	No
F_I_STD_COST_AMT	The cost of the latest item supplied in primary currency. Used to reflect the difference in unit cost if cost method accounting is used.	NUMBER(18,4)	46	No
F_I_STD_COST_AMT_LCL	The cost of the latest item supplied in local currency. Used to reflect the difference in unit cost if cost method accounting is used.	NUMBER(18,4)	47	No
F_I_RTL_AMT	The corporate unit purchase price for stock in primary currency.	NUMBER(18,4)	48	No

Name	Description	Data Type/Bytes	Field order	Required field
F_I_RTL_AMT_LCL	The corporate unit purchase price for stock in local currency.	NUMBER(18,4)	49	No
F_I_AGED_30_60_QTY	This column is not populated in the base version of RDW. This fact is used to record the quantity of inventory that is between 30 and 60 days old at this location on this day.	NUMBER(12,4)	50	No
F_I_AGED_61_90_QTY	This column is not populated in the base version of RDW. This fact is used to record the quantity of inventory that is between 61 and 90 days old at this location on this day.	NUMBER(12,4)	51	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_I_AGED_91_120_QTY	This column is not populated in the base version of RDW. This fact is used to record the quantity of inventory that is between 91 and 120 days old at this location on this day.	NUMBER(12,4)	52	No
F_I_AGED_121_QTY	This column is not populated in the base version of RDW. This fact is used to record the quantity of inventory that is 121days old or older at this location on this day.	NUMBER(12,4)	53	No
F_I_SLS_ADMN_COST_AMT	This fact could be used to store additional cost information for this item, location, and day relationship. Sales and admin cost.	NUMBER(18,4)	54	No

Name	Description	Data Type/Bytes	Field order	Required field
F_I_DIST_COST_AMT	This column is not populated in the base version of RDW. This fact could be used to store additional cost information for this item, location, and day relationship. Supply chain cost.	NUMBER(18,4)	55	No

itmclstrcmdm.txt

Business rules:

- This interface file contains the relationship between customers and item clusters.
- This interface file cannot contain duplicate records for an item_clstr_key, cust_idnt combination.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
ITEM_CLSTR_KEY	Surrogate key used to identify an Item cluster. This column is used for Behavior Profiling.	NUMBER(4)	1	Yes
CUST_IDNT	The unique identifier of the customer.	CHARACTER(15)	2	Yes

ivailddm.txt

Business rules:

- This interface file contains the inventory adjustment data for an item, location, and reason combination on a given day.
- This interface file cannot contain duplicate transactions for an item_idnt, loc_idnt, reasn_type_idnt, reasn_cde_idnt, and day_dt combination.

- It is not possible to have a different prod_seasn_key for the same item, loc and day combination. Therefore, the prod_seasn_key is not part of a primary key for any facts on the item, loc and day level. With the aggregation, it is possible to have a different prod_seasn_key at the subclass level for the same loc and day combination, or at the week level for the same item and loc combination. Therefore, the prod_seasn_key is part of the primary key for facts at the subclass and/or the week level.
- This interface file follows the fact flat file interface layout standard.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
LOC_TYPE_CDE	The code that indicates whether the location is a store or warehouse.	CHARACTER(2)	3	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	4	Yes
F_I_ADJ_QTY	The quantity of the adjustment to the total stock on hand.	NUMBER(12,4)	5	No
F_I_ADJ_COST_AMT	The cost amount of total stock on hand adjustment in primary currency.	NUMBER(18,4)	6	No
F_I_ADJ_COST_AMT_LCL	The cost amount of total stock on hand adjustment in local currency.	NUMBER(18,4)	7	No
F_I_ADJ_RTL_AMT	The retail amount of total stock on hand adjustment in primary currency.	NUMBER(18,4)	8	No
F_I_ADJ_RTL_AMT_LCL	The retail amount of total stock on hand adjustment in local currency.	NUMBER(18,4)	9	No
REASN_TYPE_IDNT	The unique identifier of the reason type.	CHARACTER(6)	10	Yes

Name	Description	Data Type/Bytes	Field order	Required field
REASN_CODE_IDNT	The unique identifier of the reason code.	CHARACTER(6)	11	Yes

ivrcpilddm.txt

Business rules:

- This interface file contains inventory receipts for an item and location combination on a given day.
- This interface file cannot contain duplicate transactions for an item_idnt, loc_idnt, and day_dt combination.
- It is not possible to have a different prod_seasn_key for the same item, loc and day combination. Therefore, the prod_seasn_key is not part of a primary key for any facts on the item, loc and day level. With the aggregation, it is possible to have a different prod_seasn_key at the subclass level for the same loc and day combination, or at the week level for the same item and loc combination. Therefore, the prod_seasn_key is part of the primary key for facts at the subclass and/or the week level.
- This interface file follows the fact flat file interface layout standard.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	3	Yes
F_I_RCPTS_QTY	The receipt quantity.	NUMBER(12,4)	4	No
F_I_RCPTS_COST_AMT	The receipt cost amount in primary currency.	NUMBER(18,4)	5	No
F_I_RCPTS_COST_AMT_LCL	The receipt cost amount in local currency.	NUMBER(18,4)	6	No

Name	Description	Data Type/Bytes	Field order	Required field
F_I_RCPTS_RTL_AMT	The receipt retail amount in primary currency.	NUMBER(18,4)	7	No
F_I_RCPTS_RTL_AMT_LCL	The receipt retail amount in local currency.	NUMBER(18,4)	8	No

ivrilddm.txt

Business rules:

- This interface file contains data on inventory returned to a supplier for a supplier, item, reason, and location combination on a given day.
- This interface file cannot contain duplicate transactions for an item_idnt, supp_idnt, loc_idnt, and day_dt combination.
- It is not possible to have a different prod_seasn_key for the same item, loc and day combination. Therefore, the prod_seasn_key is not part of a primary key for any facts on the item, loc and day level. With the aggregation, it is possible to have a different prod_seasn_key at the subclass level for the same loc and day combination, or at the week level for the same item and loc combination. Therefore, the prod_seasn_key is part of the primary key for facts at the subclass and/or the week level.
- This interface file follows the fact flat file interface layout standard.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
SUPP_IDNT	The unique identifier of a supplier.	CHARACTER(10)	1	Yes
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	2	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	3	Yes
LOC_TYPE_CDE	The code that indicates whether the location is a store or warehouse.	CHARACTER(2)	4	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	5	Yes

Name	Description	Data Type/Bytes	Field order	Required field
F_I_RTV_QTY	The quantity of the stock returned to vendor.	NUMBER(12,4)	6	No
F_I_RTV_COST_AMT	The cost of the stock returned to vendor in primary currency.	NUMBER(18,4)	7	No
F_I_RTV_COST_AMT_LCL	The cost of the stock returned to vendor in local currency.	NUMBER(18,4)	8	No
F_I_RTV_RTL_AMT	The retail amount of the stock returned to vendor, in primary currency.	NUMBER(18,4)	9	No
F_I_RTV_RTL_AMT_LCL	The retail amount of the stock returned to vendor, in local currency.	NUMBER(18,4)	10	No
REASN_TYPE_IDNT	The unique identifier of the reason type.	CHARACTER(6)	11	Yes
REASN_CODE_IDNT	The unique identifier of the reason code.	CHARACTER(6)	12	Yes

ivtilddm.txt

Business rules:

- This interface file contains inventory transfers for an item, from-location, to-location, and transfer type combination on a given day.
- This interface file cannot contain duplicate transactions for an item_idnt, loc_idnt, from_loc_idnt, tsf_type_cde, and day_dt combination.
- It is not possible to have a different prod_seasn_key for the same item, loc and day combination. Therefore, the prod_seasn_key is not part of a primary key for any facts on the item, loc and day level. With the aggregation, it is possible to have a different prod_seasn_key at the subclass level for the same loc and day combination, or at the week level for the same item and loc combination. Therefore, the prod_seasn_key is part of the primary key for facts at the subclass and/or the week level.
- This interface file follows the fact flat file interface layout standard.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
FROM_LOC_IDNT	The unique identifier for a source location for the transfer.	CHARACTER(10)	3	Yes
TSF_TYPE_CDE		CHARACTER(2)	4	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	5	Yes
F_I_TSF_TO_LOC_QTY	The quantity transferred to a destination location.	NUMBER(12,4)	6	No
F_I_TSF_TO_LOC_COST_AMT	The transfer cost amount for a destination location in primary currency.	NUMBER(18,4)	7	No
F_I_TSF_TO_LOC_COST_AMT_LCL	The transfer cost amount for a destination location in the destination's local currency.	NUMBER(18,4)	8	No

Name	Description	Data Type/Bytes	Field order	Required field
F_I_TSF_TO_LOC_RTL_AMT	The transfer retail amount for a destination location in primary currency.	NUMBER(18,4)	9	No
F_I_TSF_TO_LOC_RTL_AMT_LCL	The transfer retail amount for a destination location in the destination's local currency.	NUMBER(18,4)	10	No
F_I_TSF_FROM_LOC_QTY	The quantity transferred from a source location.	NUMBER(12,4)	11	No
F_I_TSF_FROM_LOC_COST_AMT	The transfer cost amount for a source location in primary currency.	NUMBER(18,4)	12	No
F_I_TSF_FROM_LOC_COST_AMT_LCL	The transfer cost amount for a source location in the source's local currency.	NUMBER(18,4)	13	No
F_I_TSF_FROM_LOC_RTL_AMT	The transfer retail amount for a source location in primary currency.	NUMBER(18,4)	14	No

Name	Description	Data Type/Bytes	Field order	Required field
F_I_TSF_FROM_LOC_RTL_AMT_LCL	The transfer retail amount for a source location in the source's local currency.	NUMBER(18,4)	15	No

ivuilddm.txt

Business rules:

- This interface file contains unavailable inventory for an item, location combination on a given day.
- This interface file cannot contain duplicate transactions for an item_idnt, loc_idnt and day_dt combination.
- It is not possible to have a different prod_seasn_key for the same item, loc and day combination. Therefore, the prod_seasn_key is not part of a primary key for any facts on the item, loc and day level. With the aggregation, it is possible to have a different prod_seasn_key at the subclass level for the same loc and day combination, or at the week level for the same item and loc combination. Therefore, the prod_seasn_key is part of the primary key for facts at the subclass and/or the week level.
- This interface file follows the fact flat file interface layout standard.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	3	Yes

Name	Description	Data Type/Bytes	Field order	Required field
F_I_UNAVL_QTY	The quantity of the item marked as non-sellable at the location.	NUMBER(12,4)	4	No
F_I_UNAVL_COST_AMT	The extended cost amount of unavailable inventory in primary currency. The product of the weighted average cost in primary currency and the current unavailable quantity.	NUMBER(18,4)	5	No
F_I_UNAVL_COST_AMT_LCL	The extended cost amount of unavailable inventory in local currency. The product of the weighted average cost in local currency and the current unavailable quantity.	NUMBER(18,4)	6	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_I_UNAVL_RTL_AMT	The extended retail amount of unavailable inventory in primary currency. The product of the unit retail in primary currency and the current unavailable quantity.	NUMBER(18,4)	7	No
F_I_UNAVL_RTL_AMT_LCL	The extended retail amount of unavailable inventory in local currency. The product of the unit retail in local currency and the current unavailable quantity.	NUMBER(18,4)	8	No
REASN_TYPE_IDNT	The unique identifier of the reason type.	CHARACTER(6)	9	Yes
REASN_CODE_IDNT	The unique identifier of the reason code.	CHARACTER(6)	10	Yes
LOC_TYPE_CDE	The code that indicates whether the location is a store or warehouse.	CHARACTER(2)	11	Yes

Iptldmdm.txt

Business rules:

- This interface file contains all the loss prevention transactions at the transaction-location-day-minute level.
- This interface file follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field order	Required field
TRAN_IDNT	The unique identifier of the transaction.	CHARACTER(30)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	3	Yes
MIN_IDNT	The unique identifier of the minute.	NUMBER(4)	4	Yes
REASN_CODE_IDNT	The unique identifier of the reason code.	CHARACTER(6)	5	Yes
REASN_TYPE_IDNT	The unique identifier of the reason type.	CHARACTER(6)	6	Yes
CSHR_IDNT	The unique identifier for a cashier.	CHARACTER(10)	7	Yes
RGSTR_IDNT	The unique identifier of the register.	CHARACTER(10)	8	Yes
F_LP_AMT	The loss prevention amount, in primary currency.	NUMBER(18,4)	9	No
F_LP_AMT_LCL	The loss prevention transaction amount, in local currency.	NUMBER(18,4)	10	No

Name	Description	Data Type/Bytes	Field order	Required field
F_DISC_COUPON_COUNT	Total count of discount coupons used on one transaction. Discount coupons are issued by the store as opposed to the manufacturer.	NUMBER(16,4)	11	No
F_DISC_COUPON_AMT	Total amount of discount coupons used on one transaction, in primary currency. Discount coupons are issued by the store as opposed to the manufacturer.	NUMBER(18,4)	12	No
F_DISC_COUPON_AMT_LCL	Total amount of discount coupons used on one transaction, in local currency. Discount coupons are issued by the store as opposed to the manufacturer.	NUMBER(18,4)	13	No

lptotclddm.txt

Business rules:

- This interface file contains loss prevention over/short totals.
- Amounts are summed in the target table by cshr_idnt, rgstr_idnt, loc_idnt, and day_dt.
- This interface file follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field order	Required field
CSHR_IDNT	The unique identifier for a cashier.	CHARACTER(10)	1	Yes

Name	Description	Data Type/Bytes	Field order	Required field
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	3	Yes
RGSTR_IDNT	The unique identifier of the register.	CHARACTER(10)	4	Yes
F_DRAWER_OS_AMT	The over/short amount in primary currency.	NUMBER(18,4)	5	No
F_DRAWER_OS_AMT_LCL	The over/short amount in local currency.	NUMBER(18,4)	6	No

lptotlddm.txt

Business rules:

- This interface file contains user-defined loss prevention totals.
- Amounts are summed in the target table by total type, location, and day.
- This interface file follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field order	Required field
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	1	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	2	Yes
TOTAL_TYPE_IDNT	The original identifier for the total to be reconciled.	CHARACTER(10)	3	Yes
F_TOTAL_AMT	The total amount in primary currency.	NUMBER(18,4)	4	No
F_TOTAL_AMT_LCL	The total amount in local currency.	NUMBER(18,4)	5	No

maralmdm.txt

Business rules

- This interface file contains the associations between location and market data.
- This interface file cannot contain duplicate records for a loc_idnt, mkt_area_level1_idnt, mkt_area_level2_idnt, and mkt_area_level3_idnt combination.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	1	Yes
MKT_AREA_LEVEL3_IDNT	The unique identifier of the level three market area.	CHARACTER(16)	2	Yes
MKT_AREA_LEVEL2_IDNT	The unique identifier of the level two market area.	CHARACTER(16)	3	Yes
MKT_AREA_LEVEL1_IDNT	The unique identifier of the level one market area.	CHARACTER(16)	4	Yes

maralvldm.txt

Business rules:

- This interface file contains market area level information.
- This interface file cannot contain duplicate records for a mkt_area_level1_idnt, mkt_area_level2_idnt and mkt_area_level3_idnt combination.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
MKT_AREA_LEVEL3_IDNT	The unique identifier of the level three market area.	CHARACTER(16)	1	Yes
MKT_AREA_LEVEL2_IDNT	The unique identifier of the level two market area.	CHARACTER(16)	2	Yes

Name	Description	Data Type/Bytes	Field Order	Required Field
MKT_AREA_LEVEL1_IDNT	The unique identifier of the level one market area.	CHARACTER(16)	3	Yes
MKT_AREA_LEVEL3_DESC	The description of a level three market area.	CHARACTER(30)	4	No
MKT_AREA_LEVEL2_DESC	The description of a level two market area.	CHARACTER(30)	5	No
MKT_AREA_LEVEL1_DESC	The description of a level one market area.	CHARACTER(30)	6	No

mdepdm.txt

Business rules

- This interface file contains market departments.
- This interface file cannot contain duplicate records for a mkt_dept_idnt.
- This interface file follows the dimension flat file interface layout standard

Name	Description	Data Type/Bytes	Field Order	Required Field
MKT_DEPT_IDNT	The unique identifier of a market department.	CHARACTER (13)	1	Yes
MKT_DEPT_DESC	The market category description.	CHARACTER (30)	2	No
OWNED_FLAG_IND	Indicates an owned department	CHARACTER (1)	3	Yes

meddm.txt

Business rules:

- This interface file cannot contain duplicate records for a media_idnt and banner_idnt combination.
- This interface file contains the complete snapshot of media information for active and released media.
- Media should not be closed if outstanding customer orders exist for the media.
- This interface file follows the dimension flat file interface layout standard.

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field Order	Required Field
MEDIA_IDNT	The identifier of a media.	CHARACTER(10)	1	Yes
BANNER_IDNT	The unique identifier of a banner. Banner represents the name of a retail company's subsidiary that is recognizable to the consumer or the name of the store as it appears on the catalog, web channel or brick and mortar store.	CHARACTER(4)	2	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	3	Yes
MEDIA_START_DT	The start date of the media. It identifies the day that the prices become effective in the media.	DATE	4	No
MEDIA_END_DT	The end date of the media. It identifies the last day that the prices are in effect for the media.	DATE	5	No

Name	Description	Data Type/Bytes	Field Order	Required Field
MEDIA_YR_IDNT	The fiscal year of the media, for example 2000 or 2001.	NUMBER(4)	6	No
MEDIA_SEASN_IDNT	The unique identifier of a media season for the media, for example fall, spring, or summer.	CHARACTER(6)	7	No
MEDIA_STATUS_CDE	The current status code of the media, for example active or released.	CHARACTER(12)	8	No
MEDIA_DESC	The description of the media.	CHARACTER(120)	9	No
MEDIA_SEASN_DESC	The description of the media season.	CHARACTER(120)	10	No
MEDIA_TYPE	The media type used to communicate with the customer, for example catalog, internet, postcard.	CHARACTER(120)	11	No
IN_HOME_DT	The date that the media is expected to arrive at customers' homes.	DATE	12	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field Order	Required Field
CO_RELEASE_DT	The date that customer orders placed under the media can be released.	DATE	13	No
CO_ACTV_DT	The first date that a customer order can be taken for the media.	DATE	14	No
MEDIA_PAGE_UOM	The unit of measure associated with the media page size.	CHARACTER(4)	15	No
F_PAGE_LEN_AMT	The length of a page in the media.	NUMBER(18,4)	16	No
F_PAGE_WID_AMT	The width of a page in the media.	NUMBER(18,4)	17	No
F_PAGE_QTY	The total number of all pages within the media.	NUMBER(12,4)	18	No
F_ONSALE_PAGE_QTY	The total number of pages within the media that are identified as "Sale" pages.	NUMBER(12,4)	19	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_SELLING_PAGE_QTY	The number of pages with selling items. This number is equal to or smaller than total number of pages within the media.	NUMBER(12,4)	20	No
F_SELLING_ITEM_QTY	The number of selling items within the media.	NUMBER(12,4)	21	No
F_ITEM_QTY	The number of inventory items within the media.	NUMBER(12,4)	22	No
F_ONSALE_ITEM_QTY	The total number of inventory items identified as "sale" priced.	NUMBER(12,4)	23	No
F_TOTAL_CRCL_QTY	The total circulation for the media.	NUMBER(12,4)	24	No
F_SPACE_COST_AMT	The space cost of the media in primary currency.	NUMBER(18,4)	25	No
F_SPACE_COST_AMT_LCL	The space cost of the media in local currency.	NUMBER(18,4)	26	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field Order	Required Field
F_EXPCT_RSPND_RATE	The total response rate expected for the media, over the life of the media. Response rate is the number of customer orders generated by a media divided by the number of media sent.	NUMBER(12,4)	27	No
F_EXPCT_AVG_CO_AMT	The expected average customer order amount in primary currency.	NUMBER(18,4)	28	No
F_EXPCT_AVG_CO_AMT_LCL	The expected average customer order amount in local currency.	NUMBER(18,4)	28	No
F_ORIG_FCST_AMT	The original forecasted marketing demand for the media, in primary currency.	NUMBER(18,4)	30	No
F_ORIG_FCST_AMT_LCL	The original forecasted marketing demand for the media, in local currency.	NUMBER(18,4)	31	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_CURR_FCST_AMT	The current forecasted marketing demand for the media, in primary currency.	NUMBER(18,4)	32	No
F_CURR_FCST_AMT_LCL	The current forecasted marketing demand for the media, in local currency.	NUMBER(18,4)	33	No
F_AVG_PRICE_POINT_AMT	The average price point for all inventory items in the media, in primary currency.	NUMBER(18,4)	34	No
F_AVG_PRICE_POINT_AMT_LCL	The average price point for all inventory items in the media, in local currency.	NUMBER(18,4)	35	No
F_MEDIAN_PRICE_POINT_AMT	The median price point for all inventory items in the media, in primary currency.	NUMBER(18,4)	36	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_MEDIAN_PRICE_POINT_AMT_LCL	The median price point for all inventory items in the media, in local currency.	NUMBER(18,4)	37	No

meddm.txt

Add loc_idnt into the API section, where loc_idnt will have column order 3. See Table media_loc_mtx_dm for more details.

Business rules:

- This interface file cannot contain duplicate records for a media_idnt and banner_idnt combination.
- This interface file contains the complete snapshot of media information for active and released media.
- Media should not be closed if outstanding customer orders exist for the media.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field order	Required field
MEDIA_IDNT	The identifier of a media.	CHARACTER(10)	1	Yes
BANNER_IDNT	The unique identifier of a banner. Banner represents the name of a retail company's subsidiary that is recognizable to the consumer or the name of the store as it appears on the catalog, web channel or brick and mortar store.	CHARACTER(4)	2	Yes
MEDIA_START_DT	The start date of the media. It identifies the day that the prices become effective in the media.	DATE	4	No
MEDIA_END_DT	The end date of the media. It identifies the last day that the prices are in effect for the media.	DATE	5	No

Name	Description	Data Type/Bytes	Field order	Required field
MEDIA_YR_IDNT	The fiscal year of the media, for example 2000 or 2001.	NUMBER(4)	6	No
MEDIA_SEASN_IDNT	The unique identifier of a media season for the media, for example fall, spring, or summer.	CHARACTER(6)	7	No
MEDIA_STATUS_CDE	The current status code of the media, for example active or released.	CHARACTER(12)	8	No
MEDIA_DESC	The description of the media.	CHARACTER(120)	9	No
MEDIA_SEASN_DESC	The description of the media season.	CHARACTER(120)	10	No
MEDIA_TYPE	The media type used to communicate with the customer, for example catalog, internet, postcard.	CHARACTER(120)	11	No
IN_HOME_DT	The date that the media is expected to arrive at customers' homes.	DATE	12	No
CO_RELEASE_DT	The date that customer orders placed under the media can be released.	DATE	13	No
CO_ACTV_DT	The first date that a customer order can be taken for the media.	DATE	14	No
MEDIA_PAGE_UOM	The unit of measure associated with the media page size.	CHARACTER(4)	15	No
F_PAGE_LEN_AMT	The length of a page in the media.	NUMBER(18,4)	16	No
F_PAGE_WID_AMT	The width of a page in the media.	NUMBER(18,4)	17	No
F_PAGE_QTY	The total number of all pages within the media.	NUMBER(12,4)	18	No
F_ONSALE_PAGE_QTY	The total number of pages within the media that are identified as "Sale" pages.	NUMBER(12,4)	19	No

Name	Description	Data Type/Bytes	Field order	Required field
F_SELLING_PAGE_QTY	The number of pages with selling items. This number is equal to or smaller than total number of pages within the media.	NUMBER(12,4)	20	No
F_SELLING_ITEM_QTY	The number of selling items within the media.	NUMBER(12,4)	21	No
F_ITEM_QTY	The number of inventory items within the media.	NUMBER(12,4)	22	No
F_ONSALE_ITEM_QTY	The total number of inventory items identified as "sale" priced.	NUMBER(12,4)	23	No
F_TOTAL_CRCL_QTY	The total circulation for the media.	NUMBER(12,4)	24	No
F_SPACE_COST_AMT	The space cost of the media in primary currency.	NUMBER(18,4)	25	No
F_SPACE_COST_AMT_LCL	The space cost of the media in local currency.	NUMBER(18,4)	26	No
F_EXPCT_RSPND_RATE	The total response rate expected for the media, over the life of the media. Response rate is the number of customer orders generated by a media divided by the number of media sent.	NUMBER(12,4)	27	No
F_EXPCT_AVG_CO_AMT	The expected average customer order amount in primary currency.	NUMBER(18,4)	28	No
F_EXPCT_AVG_CO_AMT_LCL	The expected average customer order amount in local currency.	NUMBER(18,4)	28	No
F_ORIG_FCST_AMT	The original forecasted marketing demand for the media, in primary currency.	NUMBER(18,4)	30	No
F_ORIG_FCST_AMT_LCL	The original forecasted marketing demand for the media, in local currency.	NUMBER(18,4)	31	No

Name	Description	Data Type/Bytes	Field order	Required field
F_CURR_FCST_AMT	The current forecasted marketing demand for the media, in primary currency.	NUMBER(18,4)	32	No
F_CURR_FCST_AMT_LCL	The current forecasted marketing demand for the media, in local currency.	NUMBER(18,4)	33	No
F_AVG_PRICE_POINT_AMT	The average price point for all inventory items in the media, in primary currency.	NUMBER(18,4)	34	No
F_AVG_PRICE_POINT_AMT_LCL	The average price point for all inventory items in the media, in local currency.	NUMBER(18,4)	35	No
F_MEDIAN_PRICE_POINT_AMT	The median price point for all inventory items in the media, in primary currency.	NUMBER(18,4)	36	No
F_MEDIAN_PRICE_POINT_AMT_LCL	The median price point for all inventory items in the media, in local currency.	NUMBER(18,4)	37	No

media_lfl_by_media_dm.txt

Business rules:

- This interface file cannot contain duplicate records for a media_key, last_yr_media_key, and last_seasn_media_key combination.
- This interface file contains user-defined relationships between a given media and the previous year media and previous season media.
- last_yr_media_key and last_seasn_media_key fields should be filled with a -2 if no relationship is to be defined.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
MEDIA_KEY	Surrogate key used to identify a media.	NUMBER(8)	1	Yes
LAST_YR_MEDIA_KEY	The surrogate key of the last year's media that corresponds to this	NUMBER(8)	2	Yes

	media.			
LAST_SEASN_MEDIA_KEY	The surrogate key of the last season's media that corresponds to this media.	NUMBER(8)	3	Yes

meditmsidm.txt

Business rules:

- This interface file cannot contain duplicate records for a media_idnt, banner_idnt, selling_item_idnt, and item_idnt combination.
- This interface file contains only the current day's new or changed information at the media, selling item, and inventory item level.
- Only changes for the defined fields need to be considered.
- This interface file follows the fact flat file interface layout standard.
- This interface file contains only active or released media.

Name	Description	Data Type/Bytes	Field Order	Required Field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
SELLING_ITEM_IDNT	The unique identifier of a selling item.	CHARACTER(25)	2	Yes
MEDIA_IDNT	The identifier of a media.	CHARACTER(10)	3	Yes
BANNER_IDNT	The unique identifier of a banner.	CHARACTER(4)	4	Yes
FEATURED_ITEM_IND	Indicates whether the inventory item is the featured item for the media/selling item.	CHARACTER(1)	5	No
F_MEDIA_UNIT_RTL_AMT	The unit retail amount for an item printed in the media, in primary currency.	NUMBER(18,4)	6	No
F_MEDIA_UNIT_RTL_AMT_LCL	The unit retail amount for an item printed in the media, in local currency.	NUMBER(18,4)	7	No
F_ADDL_DMSTC_DELVRY_AMT	The additional domestic delivery charge associated to the item in the media, in primary currency.	NUMBER(18,4)	8	No
F_ADDL_DMSTC_DELV	The additional domestic delivery charge associated	NUMBER(18)	9	No

Name	Description	Data Type/Bytes	Field Order	Required Field
RY_AMT_LCL	to the item in the media, in local currency.	,4)		
F_ADDL_INTL_DLVR_Y_AMT	The additional international delivery charge associated to the item in the media, in primary currency.	NUMBER(18,4)	10	No
F_ADDL_INTL_DLVR_Y_AMT_LCL	The additional international delivery charge associated to the item in the media, in local currency.	NUMBER(18,4)	11	No

medsidm.txt

Business rules:

- This interface file cannot contain duplicate records for a media_idnt, banner_idnt, and selling_item_idnt combination.
- Only changes for the defined fields in the API specifications are considered.
- This interface file contains only the current day's new or changed information.
- This interface file follows the fact flat file interface layout standard.
- This interface file contains only active or released media.

Name	Description	Data Type/Bytes	Field Order	Required Field
SELLING_ITEM_IDNT	The unique identifier of a selling item.	CHARACTER(25)	1	Yes
MEDIA_IDNT	The identifier of a media.	CHARACTER(10)	2	Yes
BANNER_IDNT	The unique identifier of a banner.	CHARACTER(4)	3	Yes
RECIPE_CDE	The recipe code that is associated to the selling item within the media.	CHARACTER(6)	4	No

Name	Description	Data Type/Bytes	Field Order	Required Field
ONSALE_PAGE_IND	Indicate whether the selling item is presented on a "Sale" page.	CHARACTER(1)	5	No
WEB_STORE_FEATURE_IND	Indicates whether the selling item is featured in the web store.	CHARACTER(1)	6	No

medsidpctdm.txt

Business rules:

- This interface file cannot contain duplicate records for a media_idnt, banner_idnt, dpct_idnt, and selling_item_idnt combination.
- This interface file follows the fact flat file interface layout standard.
- Only changes for the defined fields in the API specifications are considered.
- This interface file contains only the current day's new or changed information.
- This interface file contains only active or released media.

Name	Description	Data Type/Bytes	Field Order	Required Field
SELLING_ITEM_IDNT	The unique identifier of a selling item.	CHARACTER(25)	1	Yes
DPCT_IDNT	The unique identifier of a depiction. A depiction identifies the creative representation that was used to present a selling item or group of selling items to the customer within a media.	CHARACTER(25)	2	Yes
MEDIA_IDNT	The identifier of a media.	CHARACTER(10)	3	Yes
BANNER_IDNT	The unique identifier of a banner.	CHARACTER(4)	4	Yes
MEDIA_PLACEMENT	The placement of the depiction within the	CHARACTER(50)	5	No

Name	Description	Data Type/Bytes	Field Order	Required Field
	media. Example is front cover, back cover, etc.			
PAGE_SPREAD	The page spread assignment for the depiction within the media.	CHARACTER(15)	6	No
PICTURE_CDE	The alpha/numeric pictorial assignment that represents the location of the depiction on the page. Example is "A" for the saucepan, "B" for the saute'.	CHARACTER(15)	7	No
MEDIA_DPCT_UOM	The unit of measure for the media/selling item/depiction area.	CHARACTER(4)	8	No
F_SQUARE_AMT	The amount of two-dimensional space allotted to the depiction in the media, expressed in the customer's preferred unit of measure.	NUMBER(18,4)	9	No
F_SPACE_COST_AMT	The space cost of the depiction in primary currency.	NUMBER(18,4)	10	No
F_SPACE_COST_AMT_LCL	The space cost of the depiction in local currency.	NUMBER(18,4)	11	No

mitmdm.txt

Business rules

- This interface file contains market items.
- This interface file cannot contain duplicate records for a mkt_item_idnt.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
MKT_ITEM_IDNT	The unique identifier of a market item.	CHARACTER(25)	1	Yes

MKT_DEPT_IDNT	The unique identifier of a market department.	CHARACTER (13)	2	Yes
MKT_ITEM_DESC	The market item description.	CHARACTER (40)	3	No
MKT_DEPT_DESC	The market category description.	CHARACTER (30)	4	No
VENDOR_NAME	The vendor/manufacturer of the market item.	CHARACTER (30)	5	No
BRAND_NAME	The Brand label of the market item.	CHARACTER (30)	6	No
FLAVOR_SCENT	The flavor or scent of the market item.	CHARACTER (30)	7	No
MKT_ITEM_SIZE	The market item size.	CHARACTER (10)	8	No
PROD_TYPE	The product classification.	CHARACTER (20)	9	No
PACK_TYPE	The type of packaging of the market item	CHARACTER (20)	10	No
GENERATION_CDE	Three digit code that indicates if the UPC has been revised.	CHARACTER (3)	11	No
OWNED_FLAG_IND	Indicates if it's an owned item or not.	CHARACTER (1)	12	Yes

mslsdlwdm.txt

Business rules

- This interface file contains market sales data for a market category and market area level on a given week.
- This interface file cannot contain duplicate transactions for a mkt_dept_idnt, mkt_area_level_idnt, and wk_end_dt combination.
- This interface file follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
MKT_DEPT_IDNT	The unique identifier of a market department.	CHARACTER (13)	1	Yes
MKT_AREA_LEVEL_IDNT	The unique identifier of the level one market area.	CHARACTER (16)	2	Yes

Name	Description	Data Type/Bytes	Field Order	Required Field
WK_END_DT	The date in which the week ends.	DATE	3	Yes
MKT_GEO_LEVEL	The field decides what level table the market sales data goes to. Valid values are 1, 2, 3. For example, all market sales data with MKT_GEO_LEVEL = 1 goes to MKT_SLS_ITEM_LEVEL1_W_DM table.	CHARACTER(1)	4	Yes
MKT_RECD_CURR_DT	The market data creation date.	DATE	5	Yes
F_MKT_SLS_AMT_LCL	The total sales of the market item in local currency for the week.	NUMBER(18, 4)	6	No
F_MKT_SLS_AMT	The total sales of the market item in primary currency for the week.	NUMBER(18, 4)	7	No
F_MKT_SLS_QTY	The total number of the market item sold for the week.	NUMBER(12, 4)	8	No
F_MKT_AVG_ACV_WGT_DIST_PCT	The average Weekly All Commodity Volume Weighted Distribution. A measure of the percent of stores stocking the product, weighted by All Commodity Volume.	NUMBER(12, 4)	9	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field Order	Required Field
F_MKT_AVG_MMACHV_SLS_RATE	The average Weekly Sales Value per \$MM (million dollar) All Commodity Volume (Sales Rate). The sales efficiency of the product in relation to its distribution, based on All Commodity Volume per \$MM.	NUMBER(12, 4)	10	No
F_MKT_AVG_WGT_PRICE_REDT_PCT	The weighted Average percent Price Reduction. The average amount the retail was reduced for stores selling the item, weighted by units sold at each retail.	NUMBER(12, 4)	11	No
F_MKT_AVG_STORE_SELL_ITEM_QTY	The average Weekly Items per Store Selling. The average number of different UPCs of a selected product available in each store carrying the product.	NUMBER(12, 4)	12	No
F_MKT_NORMAL_AMT_LCL	The estimated sales in local currency that would have been recorded if there were no impact from display, promotion or price reduction for the week.	NUMBER(18, 4)	13	No
F_MKT_NORMAL_AMT	The estimated sales in primary currency that would have been recorded if there were no impact from display, promotion or price reduction for the week.	NUMBER(18, 4)	14	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_MKT_NORMAL_QTY	The estimated sales units that would have been recorded if there were no impact from display, promotion or price reduction for the week.	NUMBER(12, 4)	15	No
F_MKT_SLS_PRICE_CUT_AMT_LCL	The Sales Main Ad or Price Cut in local currency. The total sales value for any item on feature, display and/or with price reductions	NUMBER(18, 4)	16	No
F_MKT_SLS_PRICE_CUT_AMT	The Sales Main Ad or Price Cut in primary currency. The total sales value for any item on feature, display and/or with price reductions	NUMBER(18, 4)	17	No
F_MKT_SLS_PRICE_CUT_QTY	The unit Sales Main Ad or Price Cut. The total unit sales for any item on feature, display and/or with price reductions.	NUMBER(12, 4)	18	No
F_MKT_MAIN_AD_AMT_LCL	The total sales in local currency for any item on feature	NUMBER(18, 4)	19	No
F_MKT_MAIN_AD_AMT	The total sales in primary currency for any item on feature.	NUMBER(18, 4)	20	No
F_MKT_MAIN_AD_QTY	The total unit sales for any item on feature	NUMBER(12, 4)	21	No

mslsilwdm.txt

Business rules

- This interface file follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
MKT_ITEM_IDNT	The unique identifier of a market item.	CHARACTER(25)	1	Yes
MKT_AREA_LEVEL_IDNT	The unique identifier of the level one market area.	CHARACTER(16)	2	Yes
WK_END_DT	The date on which the week ends.	DATE	3	Yes
MKT_GEO_LEVEL	The field decides what level table the market sales data goes to. Valid values are 1, 2, 3. For example, all market sales data with MKT_GEO_LEVEL = 1 goes to MKT_SLS_ITEM_LEVEL1_W_DM table.	CHARACTER(1)	4	Yes
MKT_RECD_CURR_DT	The market data creation date.	DATE	5	Yes
F_MKT_SLS_AMT_LCL	The total sales of the market item in local currency for the week.	NUMBER(18, 4)	6	No
F_MKT_SLS_AMT	The total sales of the market item in primary currency for the week.	NUMBER(18, 4)	7	No
F_MKT_SLS_QTY	The total number of the market item sold for the week.	NUMBER(12, 4)	8	No
F_MKT_AVG_ACV_WGT_DIST_PCT	The average Weekly All Commodity Volume Weighted Distribution. A measure of the percent of stores stocking the product, weighted by All Commodity Volume.	NUMBER(12, 4)	9	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_MKT_AVG_MMACV_SLS_RATE	The average Weekly Sales Value per \$MM (million dollar) All Commodity Volume (Sales Rate). The sales efficiency of the product in relation to its distribution, based on All Commodity Volume per \$MM.	NUMBER(12, 4)	10	No
F_MKT_AVG_WGT_PRICE_REDT_PCT	The weighted Average percent Price Reduction. The average amount the retail was reduced for stores selling the item, weighted by units sold at each retail.	NUMBER(12, 4)	11	No
F_MKT_AVG_STORE_SELL_ITEM_QTY	The average Weekly Items per Store Selling. The average number of different UPCs of a selected product available in each store carrying the product.	NUMBER(12, 4)	12	No
F_MKT_NORMAL_AMT_LCL	The estimated sales in local currency that would have been recorded if there were no impact from display, promotion or price reduction for the week.	NUMBER(18, 4)	13	No
F_MKT_NORMAL_AMT	The estimated sales in primary currency that would have been recorded if there were no impact from display, promotion or price reduction for the week.	NUMBER(18, 4)	14	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_MKT_NORMAL_QTY	The estimated sales units that would have been recorded if there were no impact from display, promotion or price reduction for the week.	NUMBER(12, 4)	15	No
F_MKT_SLS_PRICE_CUT_AMT_LCL	The Sales Main Ad or Price Cut in local currency. The total sales value for any item on feature, display and/or with price reductions	NUMBER(18, 4)	16	No
F_MKT_SLS_PRICE_CUT_AMT	The Sales Main Ad or Price Cut in primary currency. The total sales value for any item on feature, display and/or with price reductions	NUMBER(18, 4)	17	No
F_MKT_SLS_PRICE_CUT_QTY	The unit Sales Main Ad or Price Cut. The total unit sales for any item on feature, display and/or with price reductions.	NUMBER(12, 4)	18	No
F_MKT_MAIN_AD_AMT_LCL	The total sales in local currency for any item on feature	NUMBER(18, 4)	19	No
F_MKT_MAIN_AD_AMT	The total sales in primary currency for any item on feature.	NUMBER(18, 4)	20	No
F_MKT_MAIN_AD_QTY	The total unit sales for any item on feature.	NUMBER(12, 4)	21	No

ncstuiddm.txt

Business rules

- This interface file contains net cost information.
- This interface file cannot contain duplicate transactions for an item_idnt, supp_idnt, loc_idnt, day_dt combination.
- This interface file follows the fact flat file interface layout standard.

- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
SUPP_IDNT	The unique identifier of a supplier.	CHARACTER(10)	2	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	3	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	4	Yes
F_SUPP_BASE_COST_A MT	The supplier base cost of the item/supplier at a given location on a given day. It is the initial cost before any deals or discounts are applied in primary currency.	NUMBER(18,4)	5	No
F_SUPP_BASE_COST_A MT_LCL	The supplier base cost of the item/supplier at a given location on a given day. It is the initial cost before any deals or discounts are applied. It is stored in local currency.	NUMBER(18,4)	6	No

Name	Description	Data Type/Bytes	Field order	Required field
F_SUPP_NET_COST_AMT	The supplier net cost for the item/supplier/location on a given day. It is defined as the base cost minus any deal components that have been applied by the retailer. If no deals or discounts are applied at this level, the supplier net cost = supplier base cost. It is stored in primary currency.	NUMBER(18,4)	7	No
F_SUPP_NET_COST_AMT_LCL	The supplier net cost for the item/supplier/location on a given day. It is defined as the base cost minus any deal components that have been applied by the retailer. If no deals or discounts are applied at this level, the supplier net cost = supplier base cost. It is stored in local currency.	NUMBER(18,4)	8	No

Name	Description	Data Type/Bytes	Field order	Required field
F_SUPP_NET_NET_COST_AMT	The supplier net net cost of the item/supplier/location on a given day. It is defined as the net cost minus any deal components designated by a retailer as applicable to the net net cost. If no deals or discounts are applied at this level, the supplier net net cost = supplier net cost. It is stored in primary currency.	NUMBER(18,4)	9	No
F_SUPP_NET_NET_COST_AMT_LCL	The supplier net net cost of the item/supplier/location on a given day. It is defined as the net cost minus any deal components designated by a retailer as applicable to the net net cost. If no deals or discounts are applied at this level, the supplier net net cost = supplier net cost. It is stored in local currency.	NUMBER(18,4)	10	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_SUPP_DEAD_NET_COST_AMT	The supplier dead net cost of the item/supplier/location on a given day. It is the final cost after all deals or discounts have been applied. It is defined as the net net cost minus any deal components designated by a retailer as applicable to the dead net cost. If no deals or discounts are applied at this level, the supplier dead net cost = supplier net net cost. It is stored in primary currency.	NUMBER(18,4)	11	No
F_SUPP_DEAD_NET_COST_AMT_LCL	The supplier dead net cost of the item/supplier/location on a given day. It is the final cost after all deals or discounts have been applied. It is defined as the net net cost minus any deal components designated by a retailer as applicable to the dead net cost. If no deals or discounts are applied at this level, the supplier dead net cost = supplier net net cost. It is stored in local currency.	NUMBER(18,4)	12	No

orgaradm.txt

Business rules

- This interface file contains areas within a chain.
- This interface file cannot contain duplicate records for an area_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
AREA_IDNT	The unique identifier of an area in the organizational hierarchy.	CHARACTER(4)	1	Yes
AREA_DESC	The name of the area in the organizational hierarchy.	CHARACTER(120)	2	No
AREA_MGR_NAME	The name of the manager for the area.	CHARACTER(120)	3	No
CHAIN_IDNT	The unique identifier of the chain in the organizational hierarchy.	CHARACTER(4)	4	Yes

orgchandm.txt

Business rules

- This interface file contains channels within a company.
- This interface file cannot contain duplicate records for a channel_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
CHANNEL_IDNT	The unique identifier of the channel in the organizational hierarchy.	CHARACTER(4)	1	Yes

Name	Description	Data Type/Bytes	Field order	Required field
BANNER_IDNT	The unique identifier of a banner. Banner represents the name of a retail company's subsidiary that is recognizable to the consumer or the name of the store as it appears on the catalog, web channel or brick and mortar store.	CHARACTER(4)	2	Yes
CHANNEL_TYPE	The type of channel.	CHARACTER(6)	3	No
CHANNEL_DESC	The name of the channel.	CHARACTER(120)	4	No
BANNER_DESC	The name of the banner.	CHARACTER(120)	5	No

orgchndm.txt

Business rules

- This interface file contains chains within a company.
- This interface file cannot contain duplicate records for a chain_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
CHAIN_IDNT	The unique identifier of the chain in the organizational hierarchy.	CHARACTER(4)	1	Yes
CMPY_IDNT	The unique identifier of the company in product and organization hierarchy.	CHARACTER(4)	2	Yes
CHAIN_DESC	The name of the chain in the organizational hierarchy.	CHARACTER(120)	3	No
CHAIN_MGR_NAME	The name of the manager for the chain.	CHARACTER(120)	4	No

orgdisdm.txt

Business rules

- This interface file contains districts within a region.

- This interface file cannot contain duplicate records for a regn_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
DISTT_IDNT	The unique identifier of a district in the organization hierarchy.	CHARACTER(4)	1	Yes
DISTT_DESC	The name of the district in the organization hierarchy.	CHARACTER(120)	2	No
DISTT_MGR_NAME	The name of the manager responsible for this district.	CHARACTER(120)	3	No
REGN_IDNT	The unique identifier of the region in the organization hierarchy.	CHARACTER(4)	4	Yes

orglmdm.txt

Business rules

- This interface file defines the associations between location and location list.
- This interface file cannot contain duplicate records for a loclst_idnt, loc_idnt combination.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
LOCLST_IDNT	The unique identifier of a location list.	CHARACTER(10)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
LOC_TYPE_CDE	The code that indicates whether the location is a store or warehouse.	CHARACTER(2)	3	Yes

orglocdm.txt

Business rules

- This interface file contains locations within a district.
- This interface file cannot contain duplicate records for a loc_idnt.

Appendix A – Application programming interface (API) flat file specifications

- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	1	Yes
LOC_TYPE_CDE	The code that indicates whether the location is a store or warehouse.	CHARACTER(2)	2	Yes
LOC_DESC	The description or name of the store or warehouse.	CHARACTER(120)	3	No
LOC_DESC_10	The 10 character abbreviation of the store name.	CHARACTER(10)	4	No
LOC_DESC_3	The 3 character abbreviation of the store name.	CHARACTER(3)	5	No
LOC_SECND_DESC	The secondary description or name of the store or warehouse.	CHARACTER(120)	6	No
LOC_TYPE_DESC	The description of the loc_type_cde that indicates whether the location is a store or warehouse. .	CHARACTER(120)	7	No
DISTT_IDNT	The unique identifier of a district in the organization hierarchy.	CHARACTER(4)	8	Yes
DISTT_DESC	The name of the district in the organization hierarchy.	CHARACTER(120)	9	No

Name	Description	Data Type/Bytes	Field order	Required field
CRNCY_CDE_IDNT	The unique identifier of the currency code.	CHARACTER(10)	10	No
CRNCY_CDE_DESC	The description of local currency code. E.g. description for USD = US Dollar.	CHARACTER(120)	11	No
PHY_WH_IDNT	The unique identifier of the physical warehouse that is assigned to the virtual warehouse.	CHARACTER(10)	12	No
VIRTUAL_WH_IDNT	The identifier of the virtual warehouse.	CHARACTER(10)	13	No
STOCKHOLD_IND	Indicates whether the location can hold stock. In a non-multichannel environment this will always be "Y."	CHARACTER(1)	14	No
CHANNEL_IDNT	The unique identifier of the channel in the organizational hierarchy.	CHARACTER(4)	15	No
CHANNEL_DESC	The name of the channel.	CHARACTER(120)	16	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
BANNER_IDNT	The unique identifier of a banner. Banner represents the name of a retail company's subsidiary that is recognizable to the consumer or the name of the store as it appears on the catalog, web channel or brick and mortar store.	CHARACTER(4)	17	No
BANNER_DESC	The name of the banner.	CHARACTER(120)	18	No
LOC_ADDR	The street address of the store or warehouse.	CHARACTER(255)	19	No
LOC_CITY_NAME	The city in which the store or warehouse is located.	CHARACTER(120)	20	No
LOC_ST_OR_PRVNC_CDE	The state or province code in which the store or warehouse is located.	CHARACTER(7)	21	No
LOC_CNTRY_CDE	The country code in which the store or warehouse is located.	CHARACTER(10)	22	No
LOC_CNTRY_DESC	The description or name of the country code in which the store or warehouse is located.	CHARACTER(120)	23	No
LOC_PSTL_CDE	The postal code of the store or warehouse.	CHARACTER(30)	24	No

Name	Description	Data Type/Bytes	Field order	Required field
LOC_MGR_NAME	The name of the manager responsible for this store. Only valid for the store Locations.	CHARACTER(120)	25	No
LOC_FMT_CDE	The code that indicates the type of format of the location. Only valid for store locations.	CHARACTER(5)	26	No
LOC_SELLING_AREA	The location's total selling area.	NUMBER(8)	27	No
LOC_TOT_LINEAR_DISTANCE	The total linear selling space of the location.	NUMBER(8)	28	No
LOC_PRMTN_ZNE_CDE	The code that indicates the promotion zone for which this location is a member . Only valid for the store Locations.	CHARACTER(5)	29	No
LOC_TRNSFR_ZNE_CDE	The code that indicates the transfer zone for which this location is a member. Only valid for the store locations.	CHARACTER(5)	30	No
LOC_VAT_REGN	The number of the Value Added Tax region in which this store or warehouse is contained.	NUMBER(4)	31	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
LOC_VAT_INCLUDE_IND	Indicates whether or not Value Added Tax will be included in the retail prices for the store. Valid values are 'Y' or 'N'.	CHARACTER(1)	32	No
LOC_MALL_NAME	The name of the mall in which the store is located.	CHARACTER(120)	33	No
LOC_DEFAULT_WH	The number of the warehouse that may be used as the default for creating cross-dock masks. This determines which stores are associated with or sourced from a warehouse.	CHARACTER(10)	34	No
LOC_BREAK_PAC_IND	Indicates whether or not the warehouse is capable of distributing less than the supplier case quantity. Valid values are 'Y' or 'N'.	CHARACTER(1)	35	No
LOC_REMODEL_DT	The date on which the store was last remodeled.	DATE	36	No
LOC_START_DT	The start date for location.	DATE	37	No
LOC_END_DT	The end date for a location.	DATE	38	No
LOC_TOT_AREA	The total area of the location.	NUMBER(8)	39	No

Name	Description	Data Type/Bytes	Field order	Required field
LOC_NO_LOAD DOCKS	This field is client specific. The definition and use of this field is customizable for each client.	CHARACTER(4)	40	No
LOC_NO_UNLOAD DOCKS	This field is client specific. The definition and use of this field is customizable for each client.	CHARACTER(4)	41	No
LOC_UPS_DISTT	The code that indicates the UPS district for which this location is a member. Only valid for the store locations.	NUMBER(2)	42	No
LOC_TIME_ZNE	The code that indicates the time zone for which this location is a member. Only valid for the store locations.	CHARACTER(10)	43	No
LOC_FASH_LINE_NO	This field is client specific. The definition and use of this field is customizable for each client.	CHARACTER(9)	44	No
LOC_COMP_CDE	This field is client specific. The definition and use of this field is customizable for each client.	CHARACTER(2)	45	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
LOC_STORE_VOL_CAT	This field is client specific. The definition and use of this field is customizable for each client.	CHARACTER(2)	46	No
LOC_PAY_CAT	This field is client specific. The definition and use of this field is customizable for each client.	CHARACTER(1)	47	No
LOC_ACCT_CLK_ID	This field is client specific. The definition and use of this field is customizable for each client.	CHARACTER(3)	48	No
LOC_FMT_DESC	The description or name of the location format code of this location. Only valid for the store locations.	CHARACTER(120)	49	No
LOC_ST_OR_PRVNC_DESC	The description or name of the state or province in which the store or warehouse is located.	CHARACTER(120)	50	No
LOC_TRNSFR_ZNE_DESC	The description or name of the transfer zone code of this location. Only valid for the store locations.	CHARACTER(120)	51	No

Name	Description	Data Type/Bytes	Field order	Required field
LOC_PRMTN_ZNE_DESC	The description or name of the promotion zone code of this location. Only valid for the store locations.	CHARACTER(120)	52	No
STORE_CLASS	This value is populated for RPAS only. Null if RPAS is not used.	CHARACTER(1)	53	No
START_ORDER_DAYS	This value is populated for RPAS only. Null if RPAS is not used.	CHARACTER(3)	54	No
FORECAST_WH_IND	This value is populated for RPAS only. Null if RPAS is not used.	CHARACTER(1)	55	No

orgloldm.txt

Business rules

- This interface file contains one record for each location list. A location list is normally used to group locations for reporting purposes.
- This interface file cannot contain duplicate records for a loclst_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
LOCLST_IDNT	The unique identifier of a location list.	CHARACTER(10)	1	Yes
CREATE_ID	The login ID of the person who created the location list.	CHARACTER(30)	2	Yes
LOCLST_DESC	The description or name of the location list unique identifier.	CHARACTER(120)	3	No

orgltmdm.txt

Business rules

- This interface file defines the associations between location and location traits.
- This interface file cannot contain duplicate records for a loc_trait_idnt, loc_idnt combination.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
LOC_TRAIT_IDNT	The location trait unique identifier. Only valid entries are for the store locations.	CHARACTER(10)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
LOC_TYPE_CDE	The code that indicates whether the location is a store or warehouse.	CHARACTER(2)	3	No

orgltrdm.txt

Business rules

- This interface file cannot contain duplicate records for a loc_trait_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
LOC_TRAIT_IDNT	The location trait unique identifier. Only valid entries are for the store locations.	CHARACTER(10)	1	Yes
LOC_TRAIT_DESC	The description or name of the location trait unique identifier.	CHARACTER(120)	2	No

orglwmdm.txt

Business rules

- This interface file cannot contain duplicate records for a loc_idnt, comp_ind combination. (Up to two rows are expected for each location.)

- This interface file contains the complete snapshot of active information.
- This interface file follows the dimension flat file interface layout standard.
- DAY_DT cannot be null.
- DAY_DT has to be less than or equal to the current date.
- Comp date (DAY_DT with a comp ind of 'Y') is not backpostable.

Name	Description	Data Type/Bytes	Field Order	Required Field
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	1	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	2	Yes
COMP_IND	Indicates if a location is comparable or not.	CHARACTER(1)	3	Yes

orgrgndm.txt

Business rules

- This interface file contains regions within an area.
- This interface file cannot contain duplicate records for a regn_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
REGN_IDNT	The unique identifier of the region in the organization hierarchy.	CHARACTER(4)	1	Yes
REGN_DESC	The description or name of the region in the organization hierarchy.	CHARACTER(120)	2	No
REGN_MGR_NAME	The name of the manager for the region.	CHARACTER(120)	3	No
AREA_IDNT	The unique identifier of an area in the organizational hierarchy.	CHARACTER(4)	4	Yes

phasdm.txt

Business rules

- This interface file contains phases. Phases are periods of time within a season. Each day should fall within no more than one phase.

- This interface file cannot contain duplicate records for a phase_idnt, seasn_idnt combination.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
SEASN_IDNT	The season identifier.	CHARACTER(3)	1	Yes
PHASE_IDNT	The unique identifier of the phase.	CHARACTER(3)	2	Yes
PHASE_START_DT	The beginning date of the phase.	DATE	3	Yes
PHASE_END_DT	The ending date of the phase.	DATE	4	Yes
PHASE_DESC	The description or name for the phase.	CHARACTER(120)	5	No

plcblwdm.txt

Business rules

- This interface file contains future and past current planning data for a department, class, subclass, and location for a given week.
- This interface file cannot contain duplicate transactions for a day_dt, dept_idnt, class_idnt, sbclass_idnt, and loc_idnt combination.
- All values are to be in primary currency.
- Percent values are expected to be decimals.
- This interface file follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
DAY_DT	The calendar day on which the transaction occurred.	DATE	1	Yes
DEPT_IDNT	The unique identifier of a department in the product hierarchy.	CHARACTER(4)	2	Yes
CLASS_IDNT	The unique identifier of the class in the product hierarchy.	CHARACTER(4)	3	Yes
SBCLASS_IDNT	The unique identifier of the subclass in the product hierarchy.	CHARACTER(4)	4	Yes

Name	Description	Data Type/Bytes	Field Order	Required Field
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	5	Yes
F_PLN_CURR_CLRC_SLS_QTY	The current plan clearance sales units minus customer returns.	NUMBER(12, 4)	6	No
F_PLN_CURR_PRMTN_SLS_QTY	The current plan promotional sales units minus customer returns.	NUMBER(12, 4)	7	No
F_PLN_CURR_RGLR_SLS_QTY	The current plan regular sales units minus customer returns.	NUMBER(12, 4)	8	No
F_PLN_CURR_CLRC_SLS_AMT	The current plan clearance sales amount minus customer returns.	NUMBER(18, 4)	9	No
F_PLN_CURR_PRMTN_SLS_AMT	The current plan promotional sales amount minus customer returns.	NUMBER(18, 4)	10	No
F_PLN_CURR_RGLR_SLS_AMT	The current plan regular sales amount minus customer returns.	NUMBER(18, 4)	11	No
F_PLN_CURR_GRS_PRFT_AMT	The current plan gross margin amount.	NUMBER(18, 4)	12	No
F_PLN_CURR_RGLR_MKDN_AMT	The current plan regular markdown amount.	NUMBER(18, 4)	13	No
F_PLN_CURR_CLRC_MKDN_AMT	The current plan clearance markdown amount.	NUMBER(18, 4)	14	No
F_PLN_CURR_PRMTN_MKDN_AMT	The current plan promotion markdown amount.	NUMBER(18, 4)	15	No
F_PLN_CURR_SHRK_QTY	The current plan shrinkage units, the total units of loss of inventory over time due to damage, misplacement, or theft.	NUMBER(12, 4)	16	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field Order	Required Field
F_PLN_CURR_SHRK_RTL_AMT	The current plan shrinkage retail value, the total retail value of loss of inventory over time due to damage, misplacement, or theft.	NUMBER(18, 4)	17	No
F_PLN_CURR_BOP_QTY	The current plan beginning inventory units	NUMBER(12, 4)	18	No
F_PLN_CURR_BOP_COST_AMT	The current plan beginning inventory cost amount	NUMBER(18, 4)	19	No
F_PLN_CURR_BOP_RTL_AMT	The current plan beginning inventory retail amount	NUMBER(18, 4)	20	No
F_PLN_CURR_OTB_QTY	The current plan quantity of goods that may be received in stock without exceeding planned inventory levels.	NUMBER(12, 4)	21	No
F_PLN_CURR_OTB_COST_AMT	The current plan cost of goods that may be received in stock without exceeding planned inventory levels.	NUMBER(18, 4)	22	No
F_PLN_CURR_OTB_RTL_AMT	The current plan retail of goods that may be received in stock without exceeding planned inventory levels.	NUMBER(18, 4)	23	No
F_PLN_CURR_RCPTS_QTY	The current plan quantity of goods to be received in stock.	NUMBER(12, 4)	24	No
F_PLN_CURR_RCPTS_COST_AMT	The current plan cost of planned quantity of goods to be received in stock.	NUMBER(18, 4)	21	25
F_PLN_CURR_RCPTS_RTL_AMT	The current plan retail of planned quantity of goods to be received in stock.	NUMBER(18, 4)	26	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_PLN_CURR_CMTS_RTL_AMT	The current plan retail amount of commitments made to suppliers.	NUMBER(18, 4)	27	No
F_PLN_CURR_ORD_CNCLD_RTL_AMT	The current plan on order cancel retail amount.	NUMBER(18, 4)	28	No
F_PLN_CURR_ORD_RT L_AMT	The current plan retail of goods that have been ordered but not received	NUMBER(18, 4)	29	No
F_PLN_CURR_RECL_IN_RTL_AMT	The current plan retail amount of inventory transferred in as a result of reclassification.	NUMBER(18, 4)	30	No
F_PLN_CURR_RECL_O UT_RTL_AMT	The current plan retail amount of inventory transferred out as a result of reclassification	NUMBER(18, 4)	31	No
F_PLN_CURR_RTV_RT L_AMT	The current plan goods returned to vendor expressed in retail amount.	NUMBER(18, 4)	32	No
F_PLN_CURR_CMTS_Q TY	The current plan units ordered but not approved.	NUMBER(12, 4)	33	No
F_PLN_CURR_ORD_CNCLD_QTY	The current plan cancelled orders expressed in units.	NUMBER(12, 4)	34	No
F_PLN_CURR_ORD_QT Y	The current plan quantity of goods that have been ordered but not received	NUMBER(12, 4)	35	No
F_PLN_CURR_RECL_IN_QTY	The current plan quantity of inventory transferred in as a result of reclassification .	NUMBER(12, 4)	36	No
F_PLN_CURR_RECL_O UT_QTY	The current plan quantity of inventory transferred out as a result of reclassification.	NUMBER(12, 4)	37	No
F_PLN_CURR_RTV_QT Y	The current plan goods returned to vendor expressed in units.	NUMBER(12, 4)	38	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_PLN_CURR_EOP_RTL_AMT	The current plan ending inventory retail amount.	NUMBER(18, 4)	39	No
F_PLN_CURR_WOS_AMT	The current plan weeks of supply: ratio of beginning inventory value to sales value on a weekly basis.	NUMBER(18, 4)	40	No
F_PLN_CURR_EOP_COST_AMT	The current plan ending inventory cost amount.	NUMBER(18, 4)	41	No
F_PLN_CURR_ORD_CNCLD_COST_AMT	The current plan on order cancel cost amount.	NUMBER(18, 4)	42	No
F_PLN_CURR_ORD_COST_AMT	The current plan cost of goods that have been ordered but not received.	NUMBER(18, 4)	43	No
F_PLN_CURR_CMTS_COST_AMT	The current plan cost amount of commitments made to suppliers.	NUMBER(18, 4)	44	No
F_PLN_CURR_CUM_MKUP_PCT	The current plan percentage difference between total delivered cost and total original retail value of merchandise handled within a stated time frame, inclusive of the accumulated inventory.	NUMBER(12, 4)	45	No
F_PLN_CURR_EOP_QTY	The current plan ending inventory units.	NUMBER(12, 4)	46	No
F_PLN_CURR_WOS_QTY	The current plan weeks of supply: ratio of beginning inventory units to sales units on a weekly basis.	NUMBER(12, 4)	47	No
F_PLN_CURR_COGS_AMT	The current plan cost of goods sold amount	NUMBER(18, 4)	48	No
F_PLN_CURR_EXCL_SLS_VAT_AMT	The current plan total sales value excluding value added tax amount. Sales value includes regular, clearance and promotional sales minus customer returns.	NUMBER(18, 4)	53	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_PLN_CURR_EMPLY_DISC_AMT	The current plan employee discount at retail.	NUMBER(18, 4)	50	No
F_PLN_CURR_FRGHT_COST_AMT	The current plan freight cost amount.	NUMBER(18, 4)	51	No
F_PLN_CURR_WRKRM_COST_AMT	The current plan workroom cost amount.	NUMBER(18, 4)	52	No
F_PLN_CURR_RTRNS_LS_AMT	The current plan customer sales return retail amount.	NUMBER(18, 4)	53	No

plnsendm.txt

Business rules

- This interface file contains plan seasons.
- This interface file cannot contain duplicate records for a pln_seasn_idnt.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
PLN_SEASN_IDNT	The unique identifier for a plan season.	CHARACTER (6)	1	Yes
PLN_SEASN_START_DT	The plan season start date	DATE	2	Yes
PLN_SEASN_END_DT	The plan season end date.	DATE	3	Yes
PLN_SEASN_DESC	The description of plan season	CHARACTER (30)	4	Yes

ploblwdm.txt

Business rules

- This interface file contains future and past original planning data for a department, class, subclass, and location for a given week.
- This interface file cannot contain duplicate transactions for a day_dt, dept_idnt, class_idnt, sbclass_idnt, and loc_idnt combination.
- This interface file follows the fact flat file interface layout standard.
- All values are to be in primary currency.
- Percent values are expected to be decimals.

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field Order	Required Field
DAY_DT	The calendar day on which the transaction occurred.	DATE	1	Yes
DEPT_IDNT	The unique identifier of a department in the product hierarchy.	CHARACTER(4)	2	Yes
CLASS_IDNT	The unique identifier of the class in the product hierarchy.	CHARACTER(4)	3	Yes
SBCLASS_IDNT	The unique identifier of the subclass in the product hierarchy.	CHARACTER(4)	4	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	5	Yes
F_PLN_ORIG_CLRC_SLS_QTY	The original plan clearance sales units minus customer returns.	NUMBER(12,4)	6	No
F_PLN_ORIG_PRMTN_SLS_QTY	The original plan promotional sales units minus customer returns.	NUMBER(12,4)	7	No
F_PLN_ORIG_RGLR_SLS_QTY	The original plan regular sales units minus customer returns.	NUMBER(12,4)	8	No
F_PLN_ORIG_CLRC_SLS_AMT	The original plan clearance sales amount minus customer returns.	NUMBER(18,4)	9	No
F_PLN_ORIG_PRMTN_SLS_AMT	The original plan promotional sales amount minus customer returns.	NUMBER(18,4)	10	No
F_PLN_ORIG_RGLR_SLS_AMT	The original plan regular sales amount minus customer returns.	NUMBER(18,4)	11	No
F_PLN_ORIG_GRS_PRFT_AMT	The original plan gross margin amount.	NUMBER(18,4)	12	No
F_PLN_ORIG_RGLR_MKDN_AMT	The original plan regular markdown amount.	NUMBER(18,4)	13	No
F_PLN_ORIG_CLRC_MKDN_AMT	The original plan clearance markdown amount.	NUMBER(18,4)	14	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_PLN_ORIG_PRMTN_MKDN_AMT	The original plan promotion markdown amount.	NUMBER(18,4)	15	No
F_PLN_ORIG_SHRK_QTY	The original plan shrinkage units, the total units of loss of inventory over time due to damage, misplacement, or theft.	NUMBER(12,4)	16	No
F_PLN_ORIG_SHRK_RT_L_AMT	The original plan shrinkage retail value, the total retail value of loss of inventory over time due to damage, misplacement, or theft.	NUMBER(18,4)	17	No
F_PLN_ORIG_BOP_QTY	The original plan beginning inventory units.	NUMBER(12,4)	18	No
F_PLN_ORIG_BOP_COST_AMT	The original plan beginning inventory cost amount	NUMBER(18,4)	19	No
F_PLN_ORIG_BOP_RT_L_AMT	The original plan beginning inventory retail amount.	NUMBER(18,4)	20	No
F_PLN_ORIG_RCPTS_QTY	The original plan quantity of goods that may be received in stock without exceeding planned inventory levels.	NUMBER(12,4)	21	No
F_PLN_ORIG_RCPTS_COST_AMT	The original plan cost of goods that may be received in stock without exceeding planned inventory levels.	NUMBER(18,4)	22	No
F_PLN_ORIG_RCPTS_RT_L_AMT	The original plan retail of goods that may be received in stock without exceeding planned inventory levels.	NUMBER(18,4)	23	No
F_PLN_ORIG_CMTS_RT_L_AMT	The original plan retail amount of commitments made to suppliers.	NUMBER(18,4)	24	No
F_PLN_ORIG_ORD_CNCLD_RT_L_AMT	The original plan on order cancel retail amount.	NUMBER(18,4)	25	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field Order	Required Field
F_PLN_ORIG_ORD_RTL_AMT	The original plan retail of goods that have been ordered but not received.	NUMBER(18,4)	26	No
F_PLN_ORIG_RECL_IN_RTL_AMT	The original plan retail amount of inventory transferred in as a result of reclassification.	NUMBER(18,4)	27	No
F_PLN_ORIG_RECL_OUT_RTL_AMT	The original plan retail amount of inventory transferred out as a result of reclassification.	NUMBER(18,4)	28	No
F_PLN_ORIG_RTV_RTL_AMT	The original plan goods returned to vendor expressed in retail amount.	NUMBER(18,4)	29	No
F_PLN_ORIG_CMTS_QTY	The original plan units ordered but not approved.	NUMBER(12,4)	30	No
F_PLN_ORIG_ORD_CNCLD_QTY	The original plan cancelled orders expressed in units.	NUMBER(12,4)	31	No
F_PLN_ORIG_ORD_QTY	The original plan goods unit that have been ordered but not received.	NUMBER(12,4)	32	No
F_PLN_ORIG_RECL_IN_QTY	The original plan quantity of inventory transferred in as a result of reclassification.	NUMBER(12,4)	33	No
F_PLN_ORIG_RECL_OUT_QTY	The original plan quantity of inventory transferred out as a result of reclassification.	NUMBER(12,4)	34	No
F_PLN_ORIG_RTV_QTY	The original plan goods returned to vendor expressed in units.	NUMBER(12,4)	35	No
F_PLN_ORIG_EOP_RTL_AMT	The original plan ending inventory retail amount.	NUMBER(18,4)	36	No
F_PLN_ORIG_EOP_QTY	The original plan ending inventory units.	NUMBER(12,4)	37	No
F_PLN_ORIG_ORD_COST_AMT	The original plan cost of goods that have been ordered but not received.	NUMBER(18,4)	38	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_PLN_ORIG_ORD_CN CLLD_COST_AMT	The original plan on order cancel cost amount.	NUMBER(18,4)	39	No
F_PLN_ORIG_CMTS_C OST_AMT	The original plan cost amount of commitments made to suppliers.	NUMBER(18,4)	40	No
F_PLN_ORIG_CUM_MK UP_PCT	The original plan percentage difference between total delivered cost and total original retail value of merchandise handled within a stated time frame, inclusive of the accumulated inventory.	NUMBER(12,4)	41	No
F_PLN_ORIG_COGS_A MT	The original plan cost of goods sold amount.	NUMBER(18,4)	42	No
F_PLN_ORIG_EXCL_ SLS_VAT_AMT	The original plan total sales value excluding value added tax amount. Sales value includes regular, clearance and promotional sales minus customer returns.	NUMBER(18,4)	43	No
F_PLN_ORIG_EMPTY_ DISC_AMT	The original plan employee discount at retail.	NUMBER(18,4)	44	No
F_PLN_ORIG_FRGHT_C OST_AMT	The original plan freight cost amount.	NUMBER(18,4)	45	No
F_PLN_ORIG_WRKRM_ COST_AMT	The original plan workroom cost amount.	NUMBER(18,4)	46	No
F_PLN_ORIG_RTRNS_S LS_AMT	The original plan customer sales return retail amount.	NUMBER(18,4)	47	No
F_PLN_ORIG_EOP_COS T_AMT	The original plan ending cost amount.	NUMBER(18,4)	48	No

prcilddm.txt

Business rules

- This interface file contains prices by item and location combination on a given day.
- This interface file cannot contain duplicate transactions for an item_idnt, loc_idnt, day_dt combination.

Appendix A – Application programming interface (API) flat file specifications

- This interface file follows the fact flat file interface layout standard.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	3	Yes
LOC_TYPE_CDE	The code that indicates whether the location is a store or warehouse.	CHARACTER(2)	4	Yes
CHNG_CDE	The reason code for price change.	CHARACTER(2)	5	No
F_MULTI_UNIT_QTY	The number of units that comprise a multi-unit transaction.	NUMBER(12,4)	6	No
F_UNIT_RTL_AMT	The unit value of new retail valuation/price in primary currency.	NUMBER(18,4)	7	No
F_UNIT_RTL_AMT_LCL	The unit value of new retail valuation/price in local currency.	NUMBER(18,4)	8	No
F_MULTI_UNIT_RTL_AMT	The unit dollar value of new retail multi unit valuation/price.	NUMBER(18,4)	9	No

Name	Description	Data Type/Bytes	Field order	Required field
F_MULTI_UNIT_RTL_AMT_LCL	The unit dollar value of new retail multi unit valuation/price in local currency.	NUMBER(18,4)	10	No
SELLING_UOM_CDE	The selling unit of measure code for an item's single-unit retail. This is a non-aggregatable value.	CHARACTER(4)	11	No
MULTI_SELLING_UOM_CDE	The selling unit of measure code for an item's multi-unit retail. This is a non-aggregatable value.	CHARACTER(4)	12	No

prdcldsm.txt

Business rules

- This interface file contains classes within a department.
- This interface file cannot contain duplicate records for a dept_idnt, class_idnt combination.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
CLASS_IDNT	The unique identifier of the class in the product hierarchy.	CHARACTER(4)	1	Yes
DEPT_IDNT	The unique identifier of a department in the product hierarchy.	CHARACTER(4)	2	Yes

Name	Description	Data Type/Bytes	Field order	Required field
CLASS_DESC	The name of the class in the product hierarchy.	CHARACTER(120)	3	No
CLASS_BUYR_IDNT	The unique identifier for the buyer of the class.	CHARACTER(4)	4	No
CLASS_BUYR_NAME	The name of the buyer for this class of products	CHARACTER(120)	5	No
CLASS_MRCH_IDNT	The unique identifier of the merchandiser for this department.	CHARACTER(4)	6	No
CLASS_MRCH_NAME	The name of the merchandiser for this class of products.	CHARACTER(120)	7	No

prdcmpdm.txt

Business rules:

- This interface file contains company information.
- This interface file cannot contain duplicate records for a cmpy_idnt.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field order	Required field
CMPY_IDNT	The unique identifier of the company in product and organization hierarchy.	CHARACTER(4)	1	Yes
CMPY_DESC	The name of the company in product and organization hierarchy.	CHARACTER(120)	2	No

prddepdm.txt

Business rules

- This interface file contains departments within a group.
- This interface file cannot contain duplicate records for a dept_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
DEPT_IDNT	The unique identifier of a department in the product hierarchy.	CHARACTER(4)	1	Yes
GRP_IDNT	The unique identifier of the group in the product hierarchy.	CHARACTER(4)	2	Yes
DEPT_DESC	The name of the department in the product hierarchy.	CHARACTER(120)	3	No
DEPT_BUYR_IDNT	The unique identifier of the buyer for the department.	CHARACTER(4)	4	No
DEPT_BUYR_NAME	The name of the buyer which corresponds to the dept_buyr_idnt for the department.	CHARACTER(120)	5	No
DEPT_MRCH_IDNT	The unique character representation of the merchandiser for the department.	CHARACTER(4)	6	No
DEPT_MRCH_NAME	The name of the merchandiser that corresponds to the dept_mrch_idnt for the department.	CHARACTER(120)	7	No
PRFT_CALC_TYPE_CDE	The unique code which determines whether profit will be calculated based on cost or retail for the department.	CHARACTER(1)	8	No

Name	Description	Data Type/Bytes	Field order	Required field
PRFT_CALC_TYPE_DESC	The description of the what method the profit was calculated for the department. Typically, it would be cost or retail.	CHARACTER(120)	9	No
PURCH_TYPE_CDE	The code that determines which type of stock the items are within this department (i.e. normal stock vs. consignment stock).	CHARACTER(1)	10	No
PURCH_TYPE_DESC	The description of the type of merchandise within the department (i.e. normal stock, consignment stock, etc.).	CHARACTER(120)	11	No
BUD_INT	The budgeted intake percentage. The term is synonymous with markup percent of retail.	NUMBER(12,4)	12	No
BUD_MKUP	The budgeted markup percentage. This term is synonymous with markup percent of cost.	NUMBER(12,4)	13	No
TOTL_MKT_AMT	The total market amount expected for this department.	NUMBER(18,4)	14	No
MKUP_CALC_TYPE_CDE	The code which determines how markup is calculated for the department.	CHARACTER(1)	15	No

Name	Description	Data Type/Bytes	Field order	Required field
MKUP_CALC_TYPE_DESC	The description of the how the markup is calculated for the department.	CHARACTER(120)	16	No
OTB_CALC_TYPE_CDE	The code that determines if Open To Buy (OTB) is based on cost or retail for the department.	CHARACTER(1)	17	No
OTB_CALC_TYPE_DESC	The description of the whether the OTB is calculated based on cost or retail.	CHARACTER(120)	18	No

prddiffdm.txt

Business rules

- This interface file contains all item differentiator identifiers, along with their associated NRF industry codes.
- This interface file cannot contain duplicate records for a diff_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
DIFF_IDNT	The uniquely identifier of a differentiator. (for example, diff_type = 'S' might have these differentiators: 1, 50, 1000; then diff_type = 'C' cannot use the same numbers)	CHARACTER(10)	1	Yes
DIFF_TYPE	The unique identifier of a differentiator type. (for example, 'S' - size, 'C' - color, 'F' - flavor, 'E' - scent, 'P' - pattern).	CHARACTER(6)	2	No

Name	Description	Data Type/Bytes	Field order	Required field
DIFF_DESC	The description of the differentiator	CHARACTER(120)	3	No
INDUSTRY_CDE	A unique number that represents all possible combinations of sizes.	CHARACTER(10)	4	No
INDUSTRY_SUBGROUP	A unique number that represents all different color range group.	CHARACTER(10)	5	No

prddivdm.txt

Business rules

- This interface file contains divisions within a company.
- This interface file cannot contain duplicate records for a div_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
DIV_IDNT	The unique identifier of a division in the product hierarchy.	CHARACTER(4)	1	Yes
CMPY_IDNT	The unique identifier of the company in product and organization hierarchy.	CHARACTER(4)	2	Yes
DIV_DESC	The name of the division in the product hierarchy.	CHARACTER(120)	3	No
DIV_BUYR_IDNT	The unique character representation of the buyer for the division.	CHARACTER(4)	4	No
DIV_BUYR_NAME	The name of the buyer for the division.	CHARACTER(120)	5	No
DIV_MRCH_IDNT	The unique identifier of the merchandiser for the division.	CHARACTER(4)	6	No
DIV_MRCH_NAME	The name of the merchandiser for the division.	CHARACTER(120)	7	No

prddtypdm.txt

Business rules

- This interface file contains differentiator (diff) types.
- This interface file cannot contain duplicate records for a diff_type.
- The maximum number of diff types allowed in RDW is 30. If new diff types (inserts via the text file) plus existing diff types (on the prod_diff_type_dm table) exceeds 30, data processing errors occur.
- For more information regarding the impact of diff type dimension changes to RDW's front end, see the RDW Middle Tier Installation Guide.
- This data is loaded during installation.

Name	Description	Data Type/Bytes	Field order	Required field
DIFF_TYPE	The unique identifier of a differentiator type. (for example, 'S' - size, 'C' - color, 'F' - flavor, 'E' - scent, 'P' - pattern).	CHARACTER(6)	1	Yes
DIFF_TYPE_DESC	The description of the differentiator type.	CHARACTER(120)	2	Yes

prdgrpdm.txt

Business rules

- This interface file contains groups within a division.
- This interface file cannot contain duplicate records for a grp_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
GRP_IDNT	The unique identifier of the group in the product hierarchy.	CHARACTER(4)	1	Yes
DIV_IDNT	The unique identifier of a division in the product hierarchy.	CHARACTER(4)	2	Yes
GRP_DESC	The name of the group in the product hierarchy.	CHARACTER(120)	3	No

Name	Description	Data Type/Bytes	Field order	Required field
GRP_BUYR_IDNT	The unique character representation of the buyer for the group.	CHARACTER(4)	4	No
GRP_BUYR_NAME	The name of the buyer that corresponds with the buyr_idnt for the group.	CHARACTER(120)	5	No
GRP_MRCH_IDNT	The unique identifier of the merchandiser for the group.	CHARACTER(4)	6	No
GRP_MRCH_NAME	The name of the merchandiser that corresponds to the grp_mrch_idnt for the group.	CHARACTER(120)	7	No

prdisldm.txt

Business rules

- This interface file contains records associating tracking level items with locations and suppliers.
- This interface file cannot contain duplicate records for a supp_idnt, item_idnt, loc_idnt combination.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
SUPP_IDNT	The unique identifier of a supplier.	CHARACTER(10)	2	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	3	Yes

Name	Description	Data Type/Bytes	Field order	Required field
SUPP_PRT_NBR	The corresponding suppliers part number.	CHARACTER(30)	4	No
PRMY_SUPP_IND	Indicator to maintain and track the primary supplier for an item. Y indicates this is a primary supplier for the item at the location.	CHARACTER(1)	5	No
PRESENTATION_METHOD	The description of the packaging (if any) being taken into consideration in the specified dimensions. Valid values are 'JHOOK', 'STACK'.	CHARACTER(6)	6	No
F_SUPP_CASE_QTY	The quantity of the item in an orderable case pack from the primary supplier.	NUMBER(12,4)	7	No

prditmclstrdm.txt

Business rules

- This interface file contains item clusters.
- This interface file cannot contain duplicate records for an item_clstr_key.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field Order	Required Field
ITEM_CLSTR_KEY	Surrogate key used to identify an item cluster. This column is used for Behavior Profiling.	NUMBER(4)	1	Yes
ITEM_CLSTR_DESC	The reference name for this cluster of item's.	CHARACTER(30)	2	No
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	3	Yes

prditmdm.txt

Business rules

- This interface file contains items within a subclass, class, and department. The combination of subclass, class and department makes an item unique. For example, item 100 cannot be identified by subclass 10, because subclass 10 can belong to different classes, and represent 2 different subclasses. Item 100 belongs to a combination of subclass, class and department.
- This interface file cannot contain duplicate records for an item_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
LEVEL1_IDNT	The unique identifier of the first level item of the family.	CHARACTER(25)	2	No
LEVEL2_IDNT	The unique identifier of the second level item of the family.	CHARACTER(25)	3	No
LEVEL3_IDNT	The unique identifier of the third level item of the family.	CHARACTER(25)	4	No

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_LEVEL	The number indicating which of the three levels the item resides. Valid values are 1, 2 and 3.	NUMBER(1)	5	Yes
TRAN_LEVEL	The number indicating which of the three levels transactions occur for the item's group. Valid values are 1, 2 and 3.	NUMBER(1)	6	Yes
DIFF_1	One of the four differentiator identifier available from the source system.	CHARACTER(10)	7	No
DIFF_2	One of the four differentiator identifier available from the source system.	CHARACTER(10)	8	No
DIFF_3	One of the four differentiator identifier available from the source system.	CHARACTER(10)	9	No
DIFF_4	One of the four differentiator identifier available from the source system.	CHARACTER(10)	10	No
ITEM_AGGREGATE_IND	This value is populated for RPAS only. Null if RPAS is not used.	CHARACTER(1)	11	No
DIFF_1_AGGREGATE_IND	This value is populated for RPAS only. Null if RPAS is not used.	CHARACTER(1)	12	No

Name	Description	Data Type/Bytes	Field order	Required field
DIFF_2_AGGREGATE_IND	This value is populated for RPAS only. Null if RPAS is not used.	CHARACTER(1)	13	No
DIFF_3_AGGREGATE_IND	This value is populated for RPAS only. Null if RPAS is not used.	CHARACTER(1)	14	No
DIFF_4_AGGREGATE_IND	This value is populated for RPAS only. Null if RPAS is not used.	CHARACTER(1)	15	No
PACK_IND	Indicates if the item is a pack.	CHARACTER(1)	16	No
PACK_SELLABLE_CDE	Indicates whether the pack is sellable. A sellable pack is a group of items that is to be sold as one item, whether the pack arrived as orderable or if the retailer took it upon themselves to package and sell the items together.	CHARACTER(6)	17	No
PACK_SELLABLE_DESC	The pack sellable description. Valid descriptions are: Sellable, Non-sellable.	CHARACTER(120)	18	No
PACK_SIMPLE_CDE	Indicates whether the pack is simple. A simple pack is the grouping of multiples of one particular item to be sold as one item. An example would be a twelve pack of cola.	CHARACTER(6)	19	No

Name	Description	Data Type/Bytes	Field order	Required field
PACK_SIMPLE_DESC	The pack simple description. Valid descriptions are: Simple, complex.	CHARACTER(120)	20	No
PACK_ORDERABLE_CDE	The abbreviated code for the pack order type: vendor or buyer. An orderable pack is a pack whose contents are specified by the buyer. A vendor pack is a pack that is packaged by the vendor and can only be ordered that way.	CHARACTER(6)	21	No
PACK_ORDERABLE_DESC	The pack order type description.	CHARACTER(120)	22	No
PACK_IND	Indicates if the item is a pack.	CHARACTER(1)	16	No
PACKAGE_UOM	The unit of measure associated with the package size.	CHARACTER(4)	23	No
PACKAGE_SIZE	The size of the product printed on any packaging.	NUMBER(12,4)	24	No
SBCLASS_IDNT	The unique identifier of the subclass in the product hierarchy.	CHARACTER(4)	25	Yes
CLASS_IDNT	The unique identifier of the class in the product hierarchy.	CHARACTER(4)	26	Yes
DEPT_IDNT	The unique identifier of a department in the product hierarchy.	CHARACTER(4)	27	Yes

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_DESC	The long description of the item. This description is used through out the system to help online users identify the item.	CHARACTER(255)	28	No
ITEM_SECND_DESC	The secondary description of the item.	CHARACTER(255)	29	No
ITEM_SHRT_DESC	The shortened description of the item. This description may be the default for downloading to the point of sale system.	CHARACTER(120)	30	No
ITEM_NBR_TYPE_CDE	The code specifying what type the item is. Some valid values for this field are ITEM, UPC-A, EAN13, ISBN, etc.	CHARACTER(6)	31	No
ITEM_NBR_TYPE_DESC	The description of the item number type.	CHARACTER(120)	32	No
STND_UOM_CDE	The string that uniquely identifies the unit of measure.	CHARACTER(6)	33	No
STND_UOM_DESC	The description of the UOM_CDE for clarity.	CHARACTER(120)	34,	No
FORECAST_IND	This value is populated for RPAS only. Null if RPAS is not used.	CHARACTER(1)	35	Yes

Name	Description	Data Type/Bytes	Field order	Required field
SELLABLE_IND	Indicates whether the item can be sold. If 'N', then the only analysis available is on customer order lines of type partial within Customer Order Management	CHARACTER(1)	36	No
INV_IND	Indicates whether an item is an inventory item or a non-inventory item (such as gift certificates, labor)	CHARACTER(1)	37	No
MRCH_IND	Indicates whether the item's sales are financially tracked in the stock ledger.	CHARACTER(1)	38	No
RECIPE_CARD_IND	Indicates whether a recipe card is available for the item.	CHARACTER(1)	39	No
PRSH_IND	Indicates whether the item is perishable.	CHARACTER(1)	40	No
ITEM_TYPE_IDNT	The unique identifier for the item type. Example item types include Swatch, Component, Raw, etc.	CHARACTER(6)	41	No
CONV_TYPE_IDNT	The unique identifier for the conveyable type. Conveyable type indicates whether the product needs to be hand carried or can be placed on the conveyer belt to be moved.	CHARACTER(6)	42	No

Name	Description	Data Type/Bytes	Field order	Required field
CLLCTN_IDNT	The unique identifier for the collection to which this item belongs. A collection may be a line of leather furniture, including an armchair, ottoman, sofa, etc. which are all part of the Leather Collection.	CHARACTER(6)	43	No

prditmldm.txt

Business rules

- This interface file contains one row for each item list. An item list is normally used to group items for reporting purpose.
- This interface file cannot contain duplicate records for an itemlst_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
ITEMLIST_IDNT	The unique identifier of an item list.	CHARACTER(10)	1	Yes
CREATE_ID	The login ID of the person who created the Item List.	CHARACTER(30)	2	Yes
ITEMLIST_DESC	The description or name of the item list.	CHARACTER(120)	3	No

prditmldm.txt

Business rules

- This interface file contains the associations between item list and tracking level item identifiers.
- This interface file cannot contain duplicate records for an itemlst_idnt and item_idnt combination.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEMLST_IDNT	The unique identifier of an item list.	CHARACTER(10)	1	Yes
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	2	Yes

prditmltmdm.txt

Business rules

- This interface file contains associations among locations, tracking level items, and their location traits.
- This interface file cannot contain duplicate records for an item_idnt, loc_idnt combination.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
LAUNCH_DT	The date that the item should first be sold at the location.	DATE	3	No
DEPOSIT_CDE	The code which indicates whether a deposit is associated with this item at the location	CHARACTER(6)	4	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
FOOD_STAMP_IND	Indicates whether the item is approved for food stamps at the location.	CHARACTER(1)	5	No
REWARD_ELIGIBLE_IND	Indicates whether the item is legally valid for various types of bonus point/award programs at the location.	CHARACTER(1)	6	No
NATL_BRAND_COMP_ITEM	The nationally branded item to which you would like to compare the current item.	CHARACTER(25)	7	No
STOP_SALE_IND	Indicates that sale of the item should be stopped immediately at the location.	CHARACTER(1)	8	No
ELECT_MKT_CLUBS	The code that represents the electronic marketing clubs to which the item belongs at the location.	CHARACTER(6)	9	No
STORE_REORDERABLE_IND	Indicates whether the store may re-order the item.	CHARACTER(1)	10	No
FULL_PALLET_ITEM_IND	Indicates whether a store must reorder an item in full pallets only.	CHARACTER(1)	11	No

Name	Description	Data Type/Bytes	Field order	Required field
DEPOSIT_CDE_DESC	The deposit code description which indicates whether a deposit is associated with this item at the location.	CHARACTER(120)	12	No

prditmsmdm.txt

Business rules

- This interface file contains associations between a tracking level or above item, and a product season/phase.
- This interface file cannot contain duplicate records for an item.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
PROD_SEASN_IDNT	The unique identifier of a product season.	CHARACTER(3)	2	Yes
PROD_PHASE_IDNT	The unique identifier of the product phase.	CHARACTER(3)	3	Yes

prditmuddm.txt

Business rules

- This interface file contains the associations between user defined attributes (UDA) at the detail level.
- This interface file cannot contain duplicate records for an item_uda_dtl_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_UDA_HEAD_IDNT	The unique identifier of the UDA.	CHARACTER(5)	1	Yes
ITEM_UDA_DTL_IDNT	The unique identifier of the text or date or lov values for a uda.	CHARACTER(256)	2	Yes
ITEM_UDA_DTL_DESC	The description of UDA value, text, or date.	CHARACTER(255)	3	No

prditmuhdm.txt

Business rules

- This interface file contains distinct user defined attribute (UDA) values.
- This interface file cannot contain duplicate records for an item_uda_head_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_UDA_HEAD_IDNT	The unique identifier of the UDA.	CHARACTER(5)	1	Yes
ITEM_UDA_TYPE_CDE	The code designating the uda type: DT=date, LV=list of values, FF=Free form text.	CHARACTER(3)	2	Yes
ITEM_UDA_HEAD_DESC	The description of the UDA.	CHARACTER(120)	3	Yes

prditmumdm.txt

Business rules

- This interface file contains the associations between UDA (User Defined Attributes) at the detail level and item identifiers at the tracking level.
- This interface file cannot contain duplicate records for an item_uda_dtl_idnt and item_idnt combination.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_UDA_HEAD_IDNT	The unique identifier of the UDA.	CHARACTER(5)	1	Yes
ITEM_UDA_DTL_IDNT	The unique identifier of the text or date or lov values for a uda.	CHARACTER(256)	2	Yes
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	3	Yes

prdpimdm.txt

Business rules

- This interface file contains the associations between packs and their component tracking-level item identifiers.
- This interface file cannot contain duplicate records for a pack_idnt and item_idnt combination.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
PACK_IDNT	The unique identifier of pack.	CHARACTER(25)	1	Yes
PACK_ITEM_QTY	Total quantity of a unique item within a pack.	NUMBER(12,4)	2	No
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	3	Yes

prdsbcdm.txt

Business rules:

- This interface file contains a subclass within a class and a department.
- This interface file cannot contain duplicate records for a dept_idnt, class_idnt, subclass_idnt combination.
- This interface file follows the dimension flat file interface layout standard.

- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
SBCLASS_IDNT	The unique identifier of the subclass in the product hierarchy.	CHARACTER(4)	1	Yes
CLASS_IDNT	The unique identifier of the class in the product hierarchy.	CHARACTER(4)	2	Yes
DEPT_IDNT	The unique identifier of a department in the product hierarchy.	CHARACTER(4)	3	Yes
SBCLASS_DESC	The name of the subclass in the product hierarchy.	CHARACTER(120)	4	No
SBCLASS_BUYR_IDNT	The unique identifier of the buyer for this subclass of products.	CHARACTER(4)	5	No
SBCLASS_BUYR_NAME	The name of the buyer for this subclass of products.	CHARACTER(120)	6	No
SBCLASS_MRCH_IDNT	The unique identifier for the merchandiser of this subclass of products.	CHARACTER(4)	7	No
SBCLASS_MRCH_NAME	The name of the merchandiser for this subclass of products.	CHARACTER(120)	8	No

prmdtldm.txt

Business rules

- This interface file cannot contain duplicate records for an event_idnt, head_idnt, prmtn_dtl_idnt combination.
- This interface file contains the complete snapshot of active information.
- If a dimension identifier is required but is not available, a value of -1 is needed.
- This interface file follows the dimension flat file interface layout standard.
- event_idnt will be -3 if the promotion detail comes from DTC.

Name	Description	Data Type/Bytes	Field order	Required field
PRMTN_DTL_IDNT	The unique identifier of a promotion detail.	CHARACTER(10)	1	Yes
HEAD_IDNT	The unique identifier of a promotion head.	CHARACTER(10)	2	Yes
EVENT_IDNT	The unique identifier of a promotion event.	CHARACTER(10)	3	Yes
PRMTN_TRIG_TYPE_IDNT	The unique identifier of the promotion trigger type. Valid values can be 'offer code', 'media code', and so on.	NUMBER(10)	4	Yes
PRMTN_SRC_CDE	The unique identifier of a promotion source. Valid values can be 'DTC', 'RPM' or any other promotion source chosen by client.	CHARACTER(6)	5	Yes
PRMTN_SVC_TYPE_IDNT	The unique identifier of a promotion service type.	CHARACTER(10)	6	Yes
PRMTN_FMT_IDNT	The unique identifier of a promotion format.	CHARACTER(10)	7	Yes
BEG_DT	The date the promotion begins.	DATE	8	Yes
PRMTN_DTL_DESC	Description for the promotion detail identifier.	CHARACTER(160)	9	No
PRMTN_SVC_TYPE_DESC	Description for the promotion service type.	CHARACTER(120)	10	No
PRMTN_FMT_DESC	Description for the promotion format.	CHARACTER(120)	11	No

Name	Description	Data Type/Bytes	Field order	Required field
END_DT	The date the promotion ends.	DATE	12	No

prmevtdm.txt

Business rules

- This interface file contains promotion events and related attributes. Events are time periods used to group promotions for analysis.
- This interface file cannot contain duplicate records for an event_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
EVENT_IDNT	The unique identifier of a promotion event.	CHARACTER(10)	1	Yes
EVENT_DESC	Description for the promotion event.	CHARACTER(255)	2	No
THEME_DESC	Description for the promotion theme.	CHARACTER(120)	3	No

prmhdrdm.txt

Business rules

- This interface file contains promotion headers and their attributes. Headers define a promotion and its start/end dates.
- This interface file cannot contain duplicate records for a head_idnt.
- All promotion head_idnt records require a beginning date, even if they are 'dummy' values such as 4444-04-04.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
HEAD_IDNT	The unique identifier of a promotion head.	CHARACTER(10)	1	Yes
EVENT_IDNT	The unique identifier of a promotion event.	CHARACTER(10)	2	Yes

Name	Description	Data Type/Bytes	Field order	Required field
HEAD_NAME	Name for the promotion head.	CHARACTER(120)	3	No
HEAD_DESC	Description for the promotion head.	CHARACTER(160)	4	No
BEG_DT	The date the promotion begins.	DATE	5	Yes
END_DT	The date the promotion ends.	DATE	6	No

regngrpdm.txt

Business rules

- This interface file contains regionality group information.
- This interface file cannot contain duplicate records for a regionality_grp_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
REGIONALITY_GRP_IDNT	The unique identifier of the regionality group.	CHARACTER(4)	1	Yes
REGIONALITY_GRP_DESC	The name of the regionality group.	CHARACTER(120)	2	No
REGIONALITY_GRP_ROLE_CDE	The role that a client wants to assign to this group. This field is referenced in the code type 'ROLE'.	CHARACTER(6)	3	No
REGIONALITY_GRP_ROLE_DESC	The description for a role.	CHARACTER(120)	4	No

regnmtxdm.txt

Business rules

- This interface file contains the associations among regionality groups, departments, locations and suppliers.
- This interface file cannot contain duplicate records for a regionality_grp_idnt, loc_idnt, supp_idnt, dept_idnt combination.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
REGIONALITY_GRP_IDNT	The unique identifier of the regionality group.	CHARACTER(4)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
LOC_TYPE_CDE	The code that indicates whether the location is a store or warehouse.	CHARACTER(2)	3	Yes
SUPP_IDNT	The unique identifier of a supplier.	CHARACTER(10)	4	Yes
DEPT_IDNT	The unique identifier of a department in the product hierarchy.	CHARACTER(4)	5	Yes

rgstrdm.txt

Business rules

- This interface file contains register information.
- This interface file cannot contain duplicate records for a rgstr_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field Order	Required Field
RGSTR_IDNT	The unique identifier of the register.	CHARACTER(10)	1	Yes

Name	Description	Data Type/Bytes	Field Order	Required Field
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes

rplcilddm.txt

Business rules

- If a dimension identifier is required but is not available, a value of -1 is needed.
- This interface file follows the fact flat file interface layout standard.
- The banner_idnt corresponding to the hdr_media_idnt and line_media_idnt must be the same.
- This interface file cannot contain duplicate transactions for an item_idnt, loc_idnt, hdr_media_idnt, line_media_idnt, banner_idnt, and day_dt combination.
- This interface file contains the replacement data for an item, location, order header media, and order line media combination on a given day.

Name	Description	Data Type/Bytes	Field Order	Required Field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
DAY_DT	The transaction date when the customer order line was created or modified.	DATE	3	Yes
HDR_MEDIA_IDNT	The unique identifier of the customer order header level media.	CHARACTER(10)	4	Yes
LINE_MEDIA_IDNT	The unique identifier of the customer order line level media.	CHARACTER(10)	5	Yes
BANNER_IDNT	The unique identifier of a banner.	CHARACTER(4)	6	Yes
F_RPLC_IN_QTY	The number of units that have been received from the customer for a replacement in transaction.	NUMBER(12,4)	7	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_RPLC_OUT_QTY	The number of units that have been sent to the customer for a replacement out transaction.	NUMBER(12,4)	8	No
F_RPLC_COST_IN_AMT	The total cost, in primary currency, of the units received from the customer for a replacement in transaction.	NUMBER(18,4)	9	No
F_RPLC_COST_IN_AMT_LCL	The total cost, in local currency, of the units received from the customer for a replacement in transaction.	NUMBER(18,4)	10	No
F_RPLC_COST_OUT_AMT	The total cost, in primary currency, of the units sent to the customer for a replacement out transaction.	NUMBER(18,4)	11	No
F_RPLC_COST_OUT_AMT_LCL	The total cost, in local currency, of the units sent to the customer for a replacement out transaction.	NUMBER(18,4)	12	No

rqstactvdmdm.txt

Business rules

- This interface file cannot contain duplicate records for a rqst_actv_idnt.
- This data contains only the current day's newly created transactions.
- This data must be extracted from the source system after midnight, and only data created in the system before midnight must be extracted.
- This interface file follows the fact flat file interface layout standard.
- The format of the min_idnt field is the hour (in format HH24) followed by a number 01-60, which indicates the minute of that hour.

Name	Description	Data Type/Bytes	Field Order	Required Field
ACTV_RQST_IDNT	The unique identifier of the activity request.	CHARACTER (12)	1	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	2	Yes
MIN_IDNT	The unique identifier of the minute.	NUMBER(4)	3	Yes
ACTV_RQST_TYPE_IDNT	The unique identifier of the activity request type.	CHARACTER (120)	4	Yes
CUST_IDNT	The unique identifier of the customer.	CHARACTER (15)	5	Yes
CSR_IDNT	The unique identifier of a customer service representative.	CHARACTER (30)	6	Yes
BANNER_IDNT	The unique identifier of a banner. Banner represents the name of a retail company's subsidiary that is recognizable to the consumer or the name of the store as it appears on the catalog, web channel or brick and mortar store.	CHARACTER (4)	7	Yes
SELLING_ITEM_IDNT	The unique identifier of a selling item.	CHARACTER (25)	8	Yes
F_ACTV_RQST_COUNT	The number of activity requests. In this request, day, minute-level table, the count value can only be 1.	NUMBER(16, 4)	9	No

rqstctlgddm.txt

Business rules

- This interface file cannot contain duplicate records for a rqst_ctlg_idnt.
- This interface file follows the fact flat file interface layout standard.
- This data must be extracted from the source system after midnight, and only data created in the system before midnight must be extracted.
- This data contains only the current day's newly created transactions.

Name	Description	Data Type/Bytes	Field Order	Required Field
CTLG_RQST_IDNT	The unique identifier of the catalog request.	CHARACTER(12)	1	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	2	Yes
CTLG_TYPE_IDNT	The unique identifier of the catalog type requested.	CHARACTER(30)	3	Yes
CTLG_RQST_TYPE_IDNT	The unique identifier of the catalog request type.	CHARACTER(120)	4	Yes
RQST_ORGN_IDNT	The unique identifier of the request origin.	CHARACTER(30)	5	Yes
CUST_IDNT	The unique identifier of the customer.	CHARACTER(15)	6	Yes
CSR_IDNT	The unique identifier of a customer service representative.	CHARACTER(30)	7	Yes
F_CTLG_RQST_COUNT	The number of catalog requests. In this request, day-level table, the count value can only be 1.	NUMBER(16, 4)	8	No

rsndm.txt

Business rules

- This interface file contains the reason class, types, and codes for the reason dimension. The file can hold various kinds of transaction reasons/codes such as inventory adjustment, return-to-vendor, voids, sales, and so on. The reason class allows definition of the reason, and the corresponding types and codes can also be defined under the class.
- This interface file cannot contain duplicate records for a reasn_code_idnt, reasn_type_idnt, reasn_class_idnt combination.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
REASN_CODE_IDNT	The unique identifier of the reason code.	CHARACTER(6)	1	Yes

Name	Description	Data Type/Bytes	Field order	Required field
REASN_TYPE_IDNT	The unique identifier of the reason type.	CHARACTER(6)	2	Yes
REASN_CLASS_IDNT	The unique identifier of the reason class.	CHARACTER(6)	3	Yes
REASN_CODE_DESC	The description of the reason code	CHARACTER(120)	4	No
REASN_TYPE_DESC	The description of the reason type.	CHARACTER(120)	5	No
REASN_CLASS_DESC	The description of the reason class	CHARACTER(120)	6	No

saviddm.txt

Business rules

- This interface file contains summarized item availability quantities for a supplier, item on a given day.
- This interface file cannot contain duplicate transactions for an item_idnt, supp_idnt, and day_dt combination.
- This interface file contains only the current day's new or changed information.
- This interface file follows the fact flat file interface layout standard.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
SUPP_IDNT	The unique identifier of a supplier.	CHARACTER(10)	2	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	3	Yes
F_AVAIL_QTY	The quantity of stock available to be ordered from the supplier.	NUMBER(12,4)	4	No

scmialddm.txt

Business rules:

- Contains data pertaining to a supplier's missed shipments by location and day.
- Cannot contain duplicate transactions for a supp_idnt, loc_idnt, day_dt.
- Follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
SUPP_IDNT	The unique identifier of a supplier.	CHARACTER(10)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	3	Yes
F_MISSED_ASN_COUNT	The total number of ASN (advanced ship notice) shipments that were expected and not received.	NUMBER(16,4)	4	No

scmidlddm.txt

- Cannot contain duplicate transactions for a supp_idnt, loc_idnt, day_dt.
- Contains data pertaining to a supplier's missed deliveries by location and day.
- Follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
SUPP_IDNT	The unique identifier of a supplier.	CHARACTER(10)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	3	Yes

Name	Description	Data Type/Bytes	Field Order	Required Field
F_MISSED_SCHED_COUNT	The total number of scheduled shipments that have not been received.	NUMBER(16,4)	4	No

scmiolddm.txt

Business rules:

- Cannot contain duplicate transactions for a supp_idnt, loc_idnt, day_dt.
- Contains data pertaining to a supplier's missed purchase orders by location and day.
- Follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
SUPP_IDNT	The unique identifier of a supplier.	CHARACTER(10)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	3	Yes
F_MISSED_ORDER_COUNT	The total number of purchase order shipments that were expected and not received.	NUMBER(16,4)	4	No

scqcdm.txt

Business rules:

- Cannot contain duplicate transactions for an item_idnt, supp_idnt, ship_idnt, loc_idnt, day_dt, po_idnt.
- Contains shipment information about which items requiring QC (quality control) failed or passed the QC test.
- Follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes

Name	Description	Data Type/Bytes	Field Order	Required Field
SHIP_IDNT	The unique identifier of the shipment.	CHARACTER(10)	2	Yes
SUPP_IDNT	The unique identifier of a supplier.	CHARACTER(10)	3	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	4	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	5	Yes
PO_IDNT	The unique identifier of a purchase order.	CHARACTER(8)	6	Yes
F_QC_FLAG	Indicates whether or not quality control checking was required on the receipt.	CHARACTER(1)	7	No
F_QC_FAILED_QTY	The total quantity of items that failed quality control checks.	NUMBER(12,4)	8	No
F_QC_PASSED_QTY	The total quantity of items that passed quality control checks.	NUMBER(12,4)	9	No

scrqtlddm.txt

Business rules:

- Contains shipment information about quantity of items received.
- Cannot contain duplicate transactions for an item_idnt, supp_idnt, ship_idnt, loc_idnt, day_dt, po_idnt.
- Follows the fact flat file interface layout standard.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field Order	Required Field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
SUPP_IDNT	The unique identifier of a supplier.	CHARACTER(10)	2	Yes
SHIP_IDNT	The unique identifier of the shipment.	CHARACTER(10)	3	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	4	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	5	Yes
PO_IDNT	The unique identifier of a purchase order.	CHARACTER(8)	6	Yes
F_ASN_EXPECTED_QTY	The total advanced shipment notice (ASN) quantity expected.	NUMBER(12,4)	7	No
F_RECEIVED_QTY	The total quantity received.	NUMBER(12,4)	8	No
F_ORDERED_QTY	The total quantity ordered.	NUMBER(12,4)	9	No
F_ASN_EXPECTED_COUNT	The number of advance shipping notice (ASN) deliveries where the quantity received equaled the quantity expected. The count value can only be 0 or 1.	NUMBER(16,4)	10	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_ASN_UNDER_COUNT	The number of advanced shipping notice (ASN) deliveries where the quantity received were less than the number expected. In this day-level table, the count value can only be 0 or 1.	NUMBER(16,4)	11	No
F_ASN_OVER_COUNT	The number of advanced shipping notice (ASN) deliveries where the quantity received exceeded the number expected. In this day-level table, the count value can only be 0 or 1.	NUMBER(16,4)	12	No
F_MISMATCHED_COUNT	The number of deliveries where quantity was received for an item that was not expected. In this day-level table, the count value can only be 0 or 1.	NUMBER(16,4)	13	No
F_FULL_PO_COUNT	The number of purchase orders where all expected quantity was received. In this day-level table, the count value can only be 0 or 1.	NUMBER(16,4)	14	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_PART_PO_COUNT	The number of purchase orders where only part of the expected quantity was received. In this day-level table, the count value can only be 0 or 1.	NUMBER(16,4)	15	No
F_OVER_PO_COUNT	The number of purchase orders where more than the expected quantity was received. In this day-level table, the count value can only be 0 or 1.	NUMBER(16,4)	16	No
PICKUP_LOC	The user-entered location of shipment for client to pick up.	CHARACTER(45)	17	No
PICKUP_NBR	The user-entered identifier of a shipment.	CHARACTER(25)	18	No
PICKUP_DT	The user entered date of the pickup.	DATE	19	No

scrtllddm.txt

Business rules:

- This interface file contains shipment information about quantity of items received. This data is only associated with scrqtllddm.txt.
- This interface file contains shipment information about timeliness of receipt. This data is only associated with scrtllddm.txt.
- This interface file contains shipment information about which items requiring QC (quality control) failed or passed the QC test. This data is only associated with scqcdm.txt.
- This interface file cannot contain duplicate transactions for item_idnt, ship_idnt, supp_idnt, loc_idnt, day_dt, po_idnt. This interface file is also applied to the scrqtllddm.txt and scrtllddm.txt interface files.
- This interface file follows the fact flat file interface layout standard.

Appendix A – Application programming interface (API) flat file specifications

- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field Order	Required Field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
SUPP_IDNT	The unique identifier of a supplier.	CHARACTER(10)	2	Yes
SHIP_IDNT	The unique identifier of the shipment.	CHARACTER(10)	3	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	4	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	5	Yes
PO_IDNT	The unique identifier of a purchase order.	CHARACTER(8)	6	Yes
F_ON_TIME_COUNT	The number of deliveries where the quantity received equaled the number expected. In this day-level table, the count value can only be 0 or 1.	NUMBER(16,4)	7	No
F_EARLY_COUNT	The number of deliveries that arrived before the scheduled time. In this day-level table, the count value can only be 0 or 1.	NUMBER(16,4)	8	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_LATE_COUNT	The number of deliveries that arrived after the scheduled time. In this day-level table, the count value can only be 0 or 1.	NUMBER(16,4)	9	No
F_UNSCHEDED_COUNT	The number of deliveries that arrived on days other than the scheduled date. In this day-level table, the count value can only be 0 or 1.	NUMBER(16,4)	10	No
F_DAYS_EARLY_COUNT	The total number of days a shipment arrived before the scheduled date.	NUMBER(16,4)	11	No
F_DAYS_LATE_COUNT	The total number of days a shipment arrived after the scheduled date.	NUMBER(16,4)	12	No

sctiddm.txt

Business rules

- This interface file contains supplier contract information.
- This interface file cannot contain duplicate transactions for an item_idnt, cntct_idnt, day_dt combination.
- This interface file contains only the current day's new or changed information.
- This interface file follows the fact flat file interface layout standard.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
CNTRCT_IDNT	The unique identifier of a contract.	CHARACTER(6)	2	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	3	Yes
F_CNTRCT_QTY	The total contracted quantity to be ordered from the supplier.	NUMBER(12,4)	4	No
F_CNTRCT_COST_AMT	The unit purchase cost negotiated for this contract.	NUMBER(18,4)	5	No
F_CNTRCT_ORD_QTY	The total ordered quantity from the contract to date for all locations.	NUMBER(12,4)	6	No
F_CNTRCT_ORD_COST_AMT	The total cost value for the ordered quantity from the contract to date for all locations.	NUMBER(18,4)	7	No
F_CNTRCT_ORD_CNCLLD_QTY	The total cancelled quantities from the contract to date, for all locations and orders.	NUMBER(12,4)	8	No

Name	Description	Data Type/Bytes	Field order	Required field
F_CNTRCT_ORD_CNCLLD_COST_AMT	The total cost value for the cancelled quantities from the contract to date, for all locations and orders.	NUMBER(18,4)	9	No

seasndm.txt

Business rules

- This interface file contains seasons. Seasons are arbitrary periods of time around which some retailers organize their buying and selling patterns. Each day should fall within no more than one season.
- This interface file cannot contain duplicate records for a seasn_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
SEASN_IDNT	The unique identifier of a season.	CHARACTER(3)	1	Yes
SEASN_START_DT	The beginning date for the season.	DATE	2	Yes
SEASN_END_DT	The ending date for the season.	DATE	3	Yes
SEASN_DESC	The description or name for the season.	CHARACTER(120)	4	No

selitmdm.txt

Business rules

- This interface file cannot contain duplicate records for a selling_item_idnt.
- This interface file contains the complete snapshot of active information.
- This interface file follows the dimension flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
SELLING_ITEM_IDNT	The unique identifier of a selling item. A selling item represents a grouping of inventory items within a media.	CHARACTER(25)	1	Yes
SELLING_ITEM_DESC	The description of the selling item.	CHARACTER(255)	2	No

sfcilwdm.txt

Business rules

- This interface file contains sales forecast information for an item and location combination on a given week.
- This interface file cannot contain duplicate transactions for an item_idnt, loc_idnt, and day_dt.
- This interface file follows the fact flat file interface layout standard.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	3	Yes
F_FCST_SLS_QTY	The forecast sales quantity.	NUMBER(12,4)	4	No

sincilddm.txt

Business rules

- This interface file contains invoice and order cost information for each item on a matched invoice.
- This interface file cannot contain duplicate transactions for an item_idnt, po_idnt, invc_idnt, supp_idnt, day_dt, and loc_idnt combination.
- This interface file follows the fact flat file interface layout standard.

- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field Order	Required Field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
PO_IDNT	The unique identifier of a purchase order.	CHARACTER(8)	2	Yes
INVC_IDNT	The unique identifier of an invoice.	CHARACTER(10)	3	Yes
SUPP_IDNT	The unique identifier of a supplier.	CHARACTER(10)	4	Yes
SHIP_IDNT	The unique identifier of the shipment.	CHARACTER(10)	5	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	6	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	7	Yes
INVC_LINE_NBR	The number that differentiates invoice lines where item-PO-supp-day-ship-loc are all the same	NUMBER(18,4)	8	Yes
F_SUPP_INVC_COST_AMT	The invoice cost, in the system primary currency.	NUMBER(18,4)	9	No
F_SUPP_INVC_COST_AMT_LCL	The invoice cost, in the local currency	NUMBER(18,4)	10	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_SUPP_INVC_QTY	The quantity of an item shown on the invoice	NUMBER(12,4)	11	No
SUPP_INVC_STATUS_CDE	Status of the invoice line item. Valid values are 'U' for unmatched, 'R' for partially matched and 'M' for matched.	CHARACTER(2)	12	No

slsildmdm.txt

Business rules

- This interface file contains sales and returns for an item, location, day, minute, voucher, and transaction.
- RDW assumes that tran_idnts received from the source system are unique across media-location-register-employee-minute-day. In an example from brick and mortar, two items, sold at the same location, by the same employee in the same minute, but at two different cash registers to two different customers in two different transactions, will result in two separate and distinct tran_idnts; similarly, the same item/loc/day/minute/register but different employees, ringing up two separate transactions will result in two distinct tran_idnts.
- tran_idnt is unique across all locations.
- The format of the min_idnt field is the hour (in format HH24) followed by a number 01-60, which indicates the minute of that hour.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
TRAN_IDNT	The unique identifier of the transaction.	CHARACTER(30)	2	Yes

Name	Description	Data Type/Bytes	Field order	Required field
VCHR_IDNT	Voucher number. If the Item is a gift certificate, then the corresponding Item Number will represent a VCHR_IDNT. This attribute is not a dimensional attribute but is used to uniquely identify a record.	CHARACTER(16)	3	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	4	Yes
MIN_IDNT	The unique identifier of the minute.	NUMBER(4)	5	Yes
OVERRIDE_REASN_CODE_IDNT	The unique identifier for a reason code.	CHARACTER(6)	6	Yes
OVERRIDE_REASN_TYPE_IDNT	The unique identifier for a reason type.	CHARACTER(6)	7	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	8	Yes
RTRN_REASN_IDNT	The unique identifier used to identify a return reason code. These codes should exist in the RMS CODE_DETAIL table under 'SARR' code type.	CHARACTER(6)	9	Yes

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
CUST_REF	The customer identifier associated with the transaction.	CHARACTER(20)	10	Yes
CUST_REF_TYPE	The type of the identifier number used by a customer.	CHARACTER(6)	11	Yes
EMPLY_IDNT	The unique identifier of the employee.	CHARACTER(10)	12	Yes
SLSPRSN_IDNT	The unique identifier for a salesperson.	CHARACTER(10)	13	Yes
CSHR_IDNT	The unique identifier for a cashier.	CHARACTER(10)	14	Yes
RGSTR_IDNT	The unique identifier of the register.	CHARACTER(10)	15	Yes
REASN_CODE_IDNT	The unique identifier of the reason code.	CHARACTER(6)	16	Yes
REASN_TYPE_IDNT	The unique identifier of the reason type.	CHARACTER(6)	17	Yes
SUB_TRAN_TYPE_IDNT	The unique identifier of the sub-transaction type.	CHARACTER(6)	18	Yes
LINE_MEDIA_IDNT	The identifier of a customer order line media.	CHARACTER(10)	19	Yes
BANNER_IDNT	The unique identifier of a banner.	CHARACTER(4)	20	Yes
SELLING_ITEM_IDNT	The unique identifier of a selling item.	CHARACTER(25)	21	Yes

Name	Description	Data Type/Bytes	Field order	Required field
CO_HDR_IDNT	The unique identifier of a customer order.	CHARACTER(30)	22	Yes
CO_LINE_IDNT	The unique identifier of a customer order line.	CHARACTER(30)	23	Yes
DROP_SHIP_IND	An indicator to identify if an item is shipped directly to the customer.	CHARACTER(1)	24	No
RTL_TYPE_CDE	The price type ('R'egular, 'P'romotion, 'C'learance).	CHARACTER(2)	25	Yes
F_SLS_AMT	The value of the sale in primary currency	NUMBER(18,4)	26	No
F_SLS_AMT_LCL	The value of the sale in local currency	NUMBER(18,4)	27	No
F_SLS_QTY	The number of items involved in the sale	NUMBER(12,4)	28	No
F_SLS_PRFT_AMT	The profit amount realized on the sale in primary currency.	NUMBER(18,4)	29	No
F_SLS_PRFT_AMT_LCL	The profit amount realized on the sale in local currency.	NUMBER(18,4)	30	No
F_RTRN_AMT	The value of the return in primary currency	NUMBER(18,4)	31	No
F_RTRN_AMT_LCL	The value of the return in local currency	NUMBER(18,4)	32	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_RTRN_QTY	The number of items involved in the return	NUMBER(12,4)	33	No
F_RTRN_PRFT_AMT	The profit amount realized on the return in primary currency	NUMBER(18,4)	34	No
F_RTRN_PRFT_AMT_LCL	The profit amount realized on the return in local currency	NUMBER(18,4)	35	No
F_SLS_ENTER_ITEM_COUNT	The number of times the item is manually entered by cashier for sale	NUMBER(16,4)	36	No
F_SLS_SCAN_ITEM_COUNT	The number of times the item is scanned by cashier for sale	NUMBER(16,4)	37	No
F_RTRN_ENTER_ITEM_COUNT	The number of times the item is manually entered by cashier for return	NUMBER(16,4)	38	No
F_RTRN_SCAN_ITEM_COUNT	Number of times the item is scanned by cashier for return	NUMBER(16,4)	39	No
F_SLS_IS_MKUP_COUNT	The count of the number of in store markup sales transactions	NUMBER(16,4)	40	No
F_SLS_IS_MKDN_COUNT	The count of the number of in store markdown sales transactions	NUMBER(16,4)	41	No
F_RTRN_IS_MKUP_COUNT	The count of the number of in store markup return transactions	NUMBER(16,4)	42	No

Name	Description	Data Type/Bytes	Field order	Required field
F_RTRN_IS_MKDN_COUNT	The count of the number of in store markdown return transactions	NUMBER(16,4)	43	No
F_SLS_IS_MKUP_AMT	The total in store markup amount in primary currency for sales transactions	NUMBER(18,4)	44	No
F_SLS_IS_MKUP_AMT_LCL	The total in store markup amount in local currency for sales transactions	NUMBER(18,4)	45	No
F_RTRN_IS_MKUP_AMT	The total in store markup amount in primary currency for return transactions	NUMBER(18,4)	46	No
F_RTRN_IS_MKUP_AMT_LCL	The total in store markup amount in local currency for return transactions	NUMBER(18,4)	47	No
F_SLS_IS_MKDN_AMT	The total in store markdown amount in primary currency for sales transactions	NUMBER(18,4)	48	No
F_SLS_IS_MKDN_AMT_LCL	The total in store markdown amount in local currency for sales transactions	NUMBER(18,4)	49	No
F_RTRN_IS_MKDN_AMT	The total in store markdown amount in primary currency for return transactions	NUMBER(18,4)	50	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_RTRN_IS_MKDN_AMT_LCL	The total in store markdown amount in local currency for return transactions	NUMBER(18,4)	51	No
F_SLS_EMPLY_DISC_AMT	The total employee retail discount amount in primary currency for sales transactions	NUMBER(18,4)	52	No
F_SLS_EMPLY_DISC_AMT_LCL	The total employee retail discount amount in local currency for sales transactions	NUMBER(18,4)	53	No
F_RTRN_EMPLY_DISC_AMT	The total employee retail discount amount in primary currency for return transactions	NUMBER(18,4)	54	No
F_RTRN_EMPLY_DISC_AMT_LCL	The total employee retail discount amount in local currency for return transactions	NUMBER(18,4)	55	No
F_SLS_ACCOM_AMT	The total customer order accommodations, associated with items, in primary currency for sales transactions.	NUMBER(18,4)	56	No

Name	Description	Data Type/Bytes	Field order	Required field
F_SLS_ACCOM_AMT_LCL	The total customer order accommodations, associated with items, in local currency for sales transactions.	NUMBER(18,4)	57	No
F_SLS_VAT_AMT	The value of the sales value added tax in primary currency.	NUMBER(18,4)	58	No
F_SLS_VAT_AMT_LCL	The value of the sales value added tax in local currency	NUMBER(18,4)	59	No
F_RTRN_VAT_AMT	The value of the return value added tax in primary currency	NUMBER(18,4)	60	No
F_RTRN_VAT_AMT_LCL	The value of the return value added tax in local currency	NUMBER(18,4)	61	No

sismkdnliddm.txt

Business rules

- This interface file contains point of sale, permanent, and clearance markdown and markup information for an item, location, and retail type on a given day.
- This interface file cannot contain duplicate transactions for a item_idnt, loc_idnt, rtl_type_cde, day_dt combination.
- This interface file follows the fact flat file interface layout standard.
- This interface file contains neither break-to-sell items nor packs that contain break-to-sell component items.
- Typical markdowns, markups, markdown cancels, and markup cancels should be positive values in their respective fields. Any reversals of the transactions that use the same tran data codes contain negative values in those applicable fields.

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
RTL_TYPE_CDE	The price type ('R'egular, 'P'romotion, 'C'learance).	CHARACTER(2)	3	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	4	Yes
F_MKDN_AMT	The value of the markdown, in primary currency.	NUMBER(18,4)	5	No
F_MKDN_AMT_LCL	The value of the markdown, in local currency.	NUMBER(18,4)	6	No
F_MKDN_QTY	The quantity of the markdown	NUMBER(12,4)	7	No
F_MKUP_AMT	The value of the markup, in primary currency.	NUMBER(18,4)	8	No
F_MKUP_AMT_LCL	The value of the markup, in local currency.	NUMBER(18,4)	9	No
F_MKUP_QTY	The quantity of the markup.	NUMBER(12,4)	10	No
F_MKDN_CNCL_AMT	The value of the markdown cancel, in primary currency.	NUMBER(18,4)	11	No
F_MKDN_CNCL_AMT_LCL	The value of the markdown cancel, in local currency.	NUMBER(18,4)	12	No
F_MKDN_CNCL_QTY	The quantity of the markdown cancel.	NUMBER(12,4)	13	No

Name	Description	Data Type/Bytes	Field order	Required field
F_MKUP_CNCL_AMT	The value of the markup cancel, in primary currency.	NUMBER(18,4)	14	No
F_MKUP_CNCL_AMT_LCL	The value of the markup cancel, in local currency.	NUMBER(18,4)	15	No
F_MKUP_CNCL_QTY	The quantity of the markup cancel.	NUMBER(12,4)	16	No

slsprmilmdm.txt

Business rules

- This interface file cannot contain duplicate records for a tran_idnt, day_dt, min_idnt, prmtn_dtl_idnt, head_idnt, prmtn_src_cde and item_idnt combination.
- If a dimension identifier is required but is not available, a value of -1 is needed.
- This interface file follows the fact flat file interface layout standard.
- tran_idnt is unique across all locations.

Name	Description	Data Type/Bytes	Field order	Required field
TRAN_IDNT	The unique identifier of a sales transaction.	CHARACTER(30)	1	Yes
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	2	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	3	Yes
MIN_IDNT	The unique identifier of the minute. This is the minute the sales transaction was created.	NUMBER(4)	4	Yes

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
PRMTN_DTL_IDNT	The identifier of the promotion detail.	CHARACTER(10)	5	Yes
HEAD_IDNT	The unique identifier of the promotion.	CHARACTER(10)	6	Yes
PRMTN_SRC_CDE	The unique identifier of the promotion source. The valid value can be 'DTC', 'RMS' or others.	CHARACTER(6)	7	Yes
SELLING_ITEM_IDNT	The unique identifier of a selling item.	CHARACTER(25)	8	Yes
LINE_MEDIA_IDNT	The unique identifier of the customer order line level media.	CHARACTER(10)	9	Yes
BANNER_IDNT	The unique identifier of a banner.	CHARACTER(4)	10	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	11	Yes
CUST_REF	The customer identifier associated with the transaction.	CHARACTER(20)	12	Yes
CUST_REF_TYPE	The type of the identifier number used by a customer.	CHARACTER(6)	13	Yes

Name	Description	Data Type/Bytes	Field order	Required field
CO_LINE_IDNT	The unique identifier of a customer order line.	CHARACTER(30)	14	Yes
CO_HDR_IDNT	The unique identifier of a customer order header.	CHARACTER(30)	15	Yes
F_PRMTN_MKDN_AMT	The promotional markdown amount in primary currency.	NUMBER(18,4)	16	No
F_PRMTN_MKDN_AMT_LCL	The promotional markdown amount in local currency.	NUMBER(18,4)	17	No

spaldlddm.txt

Business rules

- This interface file contains information about the amount of space allocated for each department at a particular location on a particular day. The space is measured in one, two, or three-dimensional space (linear, square, or cubic).
- This interface file cannot contain duplicate transactions for a dept_idnt, loc_idnt and day_dt
- This interface file contains only the current day's new or changed information.
- This interface file follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
DEPT_IDNT	The unique identifier of a department in the product hierarchy.	CHARACTER(4)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	3	Yes

Name	Description	Data Type/Bytes	Field Order	Required Field
F_SA_LINEAR_AMT	The amount of linear space allotted to the item at the location, expressed in the customer's preferred unit of measure.	NUMBER(18,4)	4	No
F_SA_SQUARE_AMT	The amount of two-dimensional space allotted to the item (such as square feet or square centimeters) at the location, expressed in the customer's preferred unit of measure).	NUMBER(18,4)	5	No
F_SA_CUBIC_AMT	The amount of three-dimension space allotted to the item (such as cubic feet or cubic centimeters) at the location, expressed in the customer's preferred unit of measure.	NUMBER(18,4)	6	No
F_SA_LINEAR_MAX_AMT	The max amount of linear space allotted to the item at the location, expressed in the customer's preferred unit of measure.	NUMBER(18,4)	7	No
F_SA_SQUARE_MAX_AMT	The max amount of two-dimensional space allotted to the item (such as square feet or square centimeters) at the location, expressed in the customer's preferred unit of measure)	NUMBER(18,4)	8	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_SA_CUBIC_MAX_AMT	The max amount of three-dimension space allotted to the item (such as cubic feet or cubic centimeters) at the location, expressed in the customer's preferred unit of measure	NUMBER(18,4)	9	No
F_SA_LINEAR_MIN_AMT	The minimum amount of linear space allotted to the item at the location, expressed in the customer's preferred unit of measure.	NUMBER(18,4)	10	No
F_SA_SQUARE_MIN_AMT	The minimum amount of two-dimensional space allotted to the item (such as square feet or square centimeters) at the location, expressed in the customer's preferred unit of measure..	NUMBER(18,4)	11	No
F_SA_CUBIC_MIN_AMT	The minimum amount of three-dimension space allotted to the item (such as cubic feet or cubic centimeters) at the location, expressed in the customer's preferred unit of measure.	NUMBER(18,4)	12	No
F_SA_FACINGS	The number of facings for a display.	NUMBER(18,4)	13	No
F_SA_ON_DISP_IND	Indicates whether an item is on display or not.	CHARACTER(1)	14	No
F_SA_ON_FEAT_IND	Indicates whether an item is on feature or not.	CHARACTER(1)	15	No

spaliddm.txt

Business rules

- This interface file contains information about the amount of space allocated for each item at a particular location on a particular day. The space is measured in one, two or three-dimensional space (linear, square, or cubic).
- This interface file cannot contain duplicate transactions for a item_idnt, loc_idnt, day_dt combination.
- This interface file contains only the current day's new or changed information.
- This interface file follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
ITEM_IDNT	The unique identifier of an item.	CHARACTER(25)	1	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	2	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	3	Yes
F_SA_LINEAR_AMT	The amount of linear space allotted to the item at the location, expressed in the customer's preferred unit of measure.	NUMBER(18,4)	4	No
F_SA_SQUARE_AMT	The amount of two-dimensional space allotted to the item (such as square feet or square centimeters) at the location, expressed in the customer's preferred unit of measure).	NUMBER(18,4)	5	No
F_SA_CUBIC_AMT	The amount of three-dimension space allotted to the item (such as cubic feet or cubic centimeters) at the location, expressed in the customer's preferred unit of measure.	NUMBER(18,4)	6	No
F_SA_LINEAR_MAX_AMT	The max amount of linear space allotted to the item at the location, expressed in the customer's preferred unit of measure.	NUMBER(18,4)	7	No

Name	Description	Data Type/Bytes	Field Order	Required Field
F_SA_SQUARE_MAX_A MT	The max amount of two-dimensional space allotted to the item (such as square feet or square centimeters) at the location, expressed in the customer's preferred unit of measure)	NUMBER(18,4)	8	No
F_SA_CUBIC_MAX_AM T	The max amount of three-dimension space allotted to the item (such as cubic feet or cubic centimeters) at the location, expressed in the customer's preferred unit of measure	NUMBER(18,4)	9	No
F_SA_LINEAR_MIN_A MT	The minimum amount of linear space allotted to the item at the location, expressed in the customer's preferred unit of measure.	NUMBER(18,4)	10	No
F_SA_SQUARE_MIN_A MT	The minimum amount of two-dimensional space allotted to the time (such as square feet or square centimeters) at the location, expressed in the customer's preferred unit of measure..	NUMBER(18,4)	11	No
F_SA_CUBIC_MIN_AM T	The minimum amount of three-dimension space allotted to the item (such as cubic feet or cubic centimeters) at the location, expressed in the customer's preferred unit of measure.	NUMBER(18,4)	12	No
F_SA_FACINGS	The number of facings for a display.	NUMBER(18,4)	13	No
F_SA_ON_DISP_IND	Indicates whether an item is on display or not.	CHARACTER(1)	14	No
F_SA_ON_FEAT_IND	Indicates whether an item is on feature or not.	CHARACTER(1)	15	No

stlblmthdm.txt

Business rules:

Appendix A – Application programming interface (API) flat file specifications

- This interface file contains stock ledger values for a department, class, subclass, and location on a given month.
- This interface file cannot contain duplicate transactions for a dept_idnt, class_idnt, subclass_idnt, loc_idnt, and day_dt combination.
- This interface file can only be populated for one time, either Gregorian time or 454 time.
- This interface file is populated with DAY_DT = -1 and G_DAY_DT = end of month date Gregorian when the stock ledger uses Gregorian time; or G_DAY_DT = -1 and DAY_DT = end of month date when the stock ledger uses 454 time.
- This interface file follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field order	Required field
SBCLASS_IDNT	The unique identifier of the subclass in the product hierarchy.	CHARACTER(4)	1	Yes
CLASS_IDNT	The unique identifier of the class in the product hierarchy.	CHARACTER(4)	2	Yes
DEPT_IDNT	The unique identifier of a department in the product hierarchy.	CHARACTER(4)	3	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	4	Yes
LOC_TYPE_CDE	The code that indicates whether the location is a store or warehouse.	CHARACTER(2)	5	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	6	Yes
F_IVL_BEG_SOH_COST_AMT	The beginning of period stock on hand total cost, in primary currency	NUMBER(18,4)	7	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_BEG_SOH_COST_AMT_LCL	The beginning of period stock on hand total cost, in local currency	NUMBER(18,4)	8	N
F_IVL_BEG_SOH_RTL_AMT	The beginning of period stock on hand total retail, in primary currency	NUMBER(18,4)	9	N
F_IVL_BEG_SOH_RTL_AMT_LCL	The beginning of period stock on hand total retail, in local currency	NUMBER(18,4)	10	N
F_IVL_SOH_ADJ_COST_AMT	The value at cost of stock on hand adjustments, in primary currency.	NUMBER(18,4)	11	N
F_IVL_SOH_ADJ_COST_AMT_LCL	The value at cost of stock on hand adjustments, in local currency.	NUMBER(18,4)	12	N
F_IVL_SOH_ADJ_RTL_AMT	The value at retail of stock on hand adjustments, in primary currency	NUMBER(18,4)	13	N
F_IVL_SOH_ADJ_RTL_AMT_LCL	The value at retail of stock on hand adjustments, in local currency	NUMBER(18,4)	14	N
F_IVL_RCPTS_COST_AMT	The value at cost of inventory received, in primary currency	NUMBER(18,4)	15	N
F_IVL_RCPTS_COST_AMT_LCL	The value at cost of inventory received, in local currency	NUMBER(18,4)	16	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_RCPTS_RTL_AMT	The value at retail of inventory received, in primary currency	NUMBER(18,4)	17	N
F_IVL_RCPTS_RTL_AMT_LCL	The value at retail of inventory received, in local currency	NUMBER(18,4)	18	N
F_IVL_RTV_COST_AMT	The value at cost of inventory returned to a vendor, in primary currency	NUMBER(18,4)	19	N
F_IVL_RTV_COST_AMT_LCL	The value at cost of inventory returned to a vendor, in local currency.	NUMBER(18,4)	20	N
F_IVL_RTV_RTL_AMT	The value at retail of inventory returned to a vendor, in primary currency.	NUMBER(18,4)	21	N
F_IVL_RTV_RTL_AMT_LCL	The value at retail of inventory returned to a vendor, in local currency.	NUMBER(18,4)	22	N
F_IVL_TSF_IN_COST_AMT	The value at cost of inventory transferred in, in primary currency	NUMBER(18,4)	23	N
F_IVL_TSF_IN_COST_AMT_LCL	The value at cost of inventory transferred in, in local currency	NUMBER(18,4)	24	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_TSF_IN_RTL_AMT	The value at retail of inventory transferred in, in primary currency	NUMBER(18,4)	25	N
F_IVL_TSF_IN_RTL_AMT_LCL	The value at retail of inventory transferred in, in local currency.	NUMBER(18,4)	26	N
F_IVL_TSF_OUT_COST_AMT	The value at cost of inventory transferred out, in primary currency	NUMBER(18,4)	27	N
F_IVL_TSF_OUT_COST_AMT_LCL	The value at cost of inventory transferred out, in local currency	NUMBER(18,4)	28	N
F_IVL_TSF_OUT_RTL_AMT	The value at retail of inventory transferred out, in primary currency	NUMBER(18,4)	29	N
F_IVL_TSF_OUT_RTL_AMT_LCL	The value at retail of inventory transferred out, in local currency	NUMBER(18,4)	30	N
F_IVL_SHRK_COST_AMT	The value at cost of the difference between actual and ending inventory, in primary currency.	NUMBER(18,4)	31	N
F_IVL_SHRK_COST_AMT_LCL	The value at cost of the difference between actual and ending inventory, in local currency.	NUMBER(18,4)	32	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_SHRK_RTL_AMT	The value at retail of the difference between actual and ending inventory, in primary currency.	NUMBER(18,4)	33	N
F_IVL_SHRK_RTL_AMT_LCL	The value at retail of the difference between actual and ending inventory, in local currency.	NUMBER(18,4)	34	N
F_IVL_RTRNS_COST_AMT	The value at cost of inventory returned from sales, in primary currency	NUMBER(18,4)	35	N
F_IVL_RTRNS_COST_AMT_LCL	The value at cost of inventory returned from sales, in local currency	NUMBER(18,4)	36	N
F_IVL_RTRNS_RTL_AMT	The value at retail of inventory returned from sales, in primary currency	NUMBER(18,4)	37	N
F_IVL_RTRNS_RTL_AMT_LCL	The value at retail of inventory returned from sales, in local currency	NUMBER(18,4)	38	N
F_IVL_RECLASS_IN_COST_AMT	The value at cost of inventory reclassified to this location, in primary currency	NUMBER(18,4)	39	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_RECLASS_IN_COST_AMT_LCL	The value at cost of inventory reclassified to this location, in local currency	NUMBER(18,4)	40	N
F_IVL_RECLASS_IN_RTL_AMT	The value at retail of inventory reclassified to this location, in primary currency	NUMBER(18,4)	41	N
F_IVL_RECLASS_IN_RTL_AMT_LCL	The value at retail of inventory reclassified to this location, in local currency	NUMBER(18,4)	42	N
F_IVL_RECLASS_OUT_COST_AMT	The value at cost of inventory reclassified from this location, in primary currency	NUMBER(18,4)	43	N
F_IVL_RECLASS_OUT_COST_AMT_LCL	The value at cost of inventory reclassified from this location, in local currency	NUMBER(18,4)	44	N
F_IVL_RECLASS_OUT_RTL_AMT	The value at retail of inventory reclassified from this location, in primary currency	NUMBER(18,4)	45	N
F_IVL_RECLASS_OUT_RTL_AMT_LCL	The value at retail of inventory reclassified from this location, in local currency	NUMBER(18,4)	46	N
F_IVL_SLS_COST_AMT	The value at cost of inventory sold, in primary currency	NUMBER(18,4)	47	N

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_SLS_COST_AMT_LCL	The value at cost of inventory sold, in local currency.	NUMBER(18,4)	48	N
F_IVL_SLS_RTL_AMT	The value at retail of inventory sold, in primary currency	NUMBER(18,4)	49	N
F_IVL_SLS_RTL_AMT_LCL	The value at retail of inventory sold, in local currency.	NUMBER(18,4)	50	N
F_IVL_END_SOH_COST_AMT	The end of period stock on hand total cost, in primary currency.	NUMBER(18,4)	51	N
F_IVL_END_SOH_COST_AMT_LCL	The end of period stock on hand total cost, in local currency	NUMBER(18,4)	52	N
F_IVL_END_SOH_RTL_AMT	The end of period stock on hand total retail, in primary currency.	NUMBER(18,4)	53	N
F_IVL_END_SOH_RTL_AMT_LCL	The end of period stock on hand total retail, in local currency	NUMBER(18,4)	54	N
F_IVL_GRS_PRFT_AMT	The total gross profit amount, in primary currency	NUMBER(18,4)	55	N
F_IVL_GRS_PRFT_AMT_LCL	The total gross profit amount, in local currency.	NUMBER(18,4)	56	N
F_IVL_CUM_MKON_PC T	The cumulative markon percent.	NUMBER(12,4)	57	N
F_IVL_MKUP_AMT	The value of upward revisions in price, in primary currency.	NUMBER(18,4)	58	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_MKUP_AMT_LCL	The value of upward revisions in price, in local currency.	NUMBER(18,4)	59	N
F_IVL_MKUP_CNCLLD_AMT	The value of corrections to a upward revisions in price, in primary currency.	NUMBER(18,4)	60	N
F_IVL_MKUP_CNCLLD_AMT_LCL	The value of corrections to a upward revisions in price, in local currency.	NUMBER(18,4)	61	N
F_IVL_MKDN_CNCLLD_AMT	The value of markdown cancellation to correct an unintentional error in a previous markup, in local currency.	NUMBER(18,4)	62	N
F_IVL_MKDN_CNCLLD_AMT_LCL	The value of markdown cancellation to correct an unintentional error in a previous markup, in primary currency.	NUMBER(18,4)	63	N
F_IVL_PERM_MKDN_AMT	The value of permanent reduction in price, in primary currency.	NUMBER(18,4)	64	N
F_IVL_PERM_MKDN_AMT_LCL	The value of permanent reduction in price, in local currency.	NUMBER(18,4)	65	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_PRMTN_MKDN_AMT	The value of promotion reductions of the price, in primary currency.	NUMBER(18,4)	66	N
F_IVL_PRMTN_MKDN_AMT_LCL	The value of promotion reductions of the price, in local currency.	NUMBER(18,4)	67	N
F_IVL_CLRC_MKDN_AMT	The value of clearance reductions of the price, in primary currency.	NUMBER(18,4)	68	N
F_IVL_CLRC_MKDN_AMT_LCL	The value of clearance reductions of the price, in local currency.	NUMBER(18,4)	69	N
F_IVL_EMPTY_DISC_AMT	The value of employee discounts, in primary currency.	NUMBER(18,4)	70	N
F_IVL_EMPTY_DISC_AMT_LCL	The value of employee discounts, in local currency.	NUMBER(18,4)	71	N
F_IVL_CASH_DISC_AMT	The value of cash discounts, in primary currency.	NUMBER(18,4)	72	N
F_IVL_CASH_DISC_AMT_LCL	The value of cash discounts, in local currency.	NUMBER(18,4)	73	N
F_IVL_FRGHT_COST_AMT	The value of freight expenses, in primary currency.	NUMBER(18,4)	74	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_FRGHT_COST_AMT_LCL	The value of freight expenses, in local currency.	NUMBER(18,4)	75	N
F_IVL_WRKRM_COST_AMT	The value of workroom expenses, in primary currency.	NUMBER(18,4)	76	N
F_IVL_WRKRM_COST_AMT_LCL	The value of workroom expenses, in local currency	NUMBER(18,4)	77	N
F_IVL_GAFS_COST_AMT	The goods available for sale valued at cost, in primary currency.	NUMBER(18,4)	78	N
F_IVL_GAFS_COST_AMT_LCL	The goods available for sale valued at cost, in local currency.	NUMBER(18,4)	79	N
F_IVL_GAFS_RTL_AMT	The goods available for sale valued at retail, in primary currency.	NUMBER(18,4)	80	N
F_IVL_GAFS_RTL_AMT_LCL	The goods available for sale valued at retail, in local currency.	NUMBER(18,4)	81	N
F_IVL_SLS_QTY	The number of net units of merchandise sold.	NUMBER(12,4)	82	N
F_IVL_SLS_RTL_EX_VAT_AMT	The value at retail, excluding VAT, of net merchandise sold, in primary currency.	NUMBER(18,4)	83	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_SLS_RTL_EX_VAT_AMT_LCL	The value at retail, excluding VAT, of net merchandise sold, in local currency.	NUMBER(18,4)	84	N
F_IVL_FRGHT_CLAIM_RTL_AMT	The value at retail of freight claim, in primary currency.	NUMBER(18,4)	85	N
F_IVL_FRGHT_CLAIM_RTL_AMT_LCL	The value at retail of freight claim, in local currency.	NUMBER(18,4)	86	N
F_IVL_FRGHT_CLAIM_COST_AMT	The value at cost of freight claim, in primary currency.	NUMBER(18,4)	87	N
F_IVL_FRGHT_CLAIM_COST_AMT_LCL	The value at cost of freight claim, in local currency.	NUMBER(18,4)	88	N
F_IVL_IC_TSF_IN_COST_AMT	The value at cost of inventory transferred in for intercompany transfers, in primary currency.	NUMBER(18,4)	89	N
F_IVL_IC_TSF_IN_COST_AMT_LCL	The value at cost of inventory transferred in for intercompany transfers, in local currency.	NUMBER(18,4)	90	N
F_IVL_IC_TSF_IN_RTL_AMT	The value at retail of inventory transferred in for intercompany transfers, in primary currency.	NUMBER(18,4)	91	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_IC_TSF_IN_RTL_AMT_LCL	The value at retail of inventory transferred in for intercompany transfers, in local currency.	NUMBER(18,4)	92	N
F_IVL_IC_TSF_OUT_COST_AMT	The value at cost of inventory transferred out for intercompany transfers, in primary currency.	NUMBER(18,4)	93	N
F_IVL_IC_TSF_OUT_COST_AMT_LCL	The value at cost of inventory transferred out for intercompany transfers, in local currency.	NUMBER(18,4)	94	N
F_IVL_IC_TSF_OUT_RTL_AMT	The value at retail of inventory transferred out for intercompany transfers, in primary currency.	NUMBER(18,4)	95	N
F_IVL_IC_TSF_OUT_RTL_AMT_LCL	The value at retail of inventory transferred out for intercompany transfers, in local currency.	NUMBER(18,4)	96	N
F_IVL_IC_MARGIN_AMT	The margin value of intercompany transfers, in primary currency.	NUMBER(18,4)	97	N
F_IVL_IC_MARGIN_AMT_LCL	The margin value of intercompany transfers, in local currency.	NUMBER(18,4)	98	N

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_IC_MKDN_RTL_AMT	The markdown at retail of merchandise transferred out for intercompany transfers, in primary currency.	NUMBER(18,4)	99	N
F_IVL_IC_MKDN_RTL_AMT_LCL	The markdown at retail of merchandise transferred out for intercompany transfers, in local currency.	NUMBER(18,4)	100	N
F_IVL_IC_MKUP_RTL_AMT	The markup at retail of merchandise transferred out for intercompany transfers, in primary currency.	NUMBER(18,4)	101	N
F_IVL_IC_MKUP_RTL_AMT_LCL	The markup at retail of merchandise transferred out for intercompany transfers, in local currency.	NUMBER(18,4)	102	N
F_IVL_WO_UPD_INV_COST_AMT	The value at cost of merchandise required work order activity, update inventory, for intercompany transfers, in primary currency.	NUMBER(18,4)	103	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_WO_UPD_INV_COST_AMT_LCL	The value at cost of merchandise required work order activity, update inventory, for intercompany transfers, in local currency.	NUMBER(18,4)	104	N
F_IVL_WO_POST_FIN_COST_AMT	The value at cost of merchandise required work order activity, post to financial, for intercompany transfers, in primary currency.	NUMBER(18,4)	105	N
F_IVL_WO_POST_FIN_COST_AMT_LCL	The value at cost of merchandise required work order activity, post to financial, for intercompany transfers, in local currency.	NUMBER(18,4)	106	N
F_IVL_ADJ_COGS_COST_AMT	The value at cost of stock adjustments that affect COGS, in primary currency.	NUMBER(18,4)	107	N
F_IVL_ADJ_COGS_COST_AMT_LCL	The value at cost of stock adjustments that affect COGS, in local currency.	NUMBER(18,4)	108	N
F_IVL_ADJ_COGS_RTL_AMT	The value at retail of stock adjustments that affect COGS, in primary currency.	NUMBER(18,4)	109	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_ADJ_COGS_RTL_AMT_LCL	The value at retail of stock adjustments that affect COGS, in local currency.	NUMBER(18,4)	110	N
F_IVL_RESTOCK_FEE_AMT	The value at cost of restocking fees received, in primary currency.	NUMBER(18,4)	111	N
F_IVL_RESTOCK_FEE_AMT_LCL	The value at cost of restocking fees received, in local currency.	NUMBER(18,4)	112	N
F_IVL_DEAL_INCM_SL S_AMT	The value of deal incomes sales received, in primary currency.	NUMBER(18,4)	113	N
F_IVL_DEAL_INCM_SL S_AMT_LCL	The value of deal incomes sales received, in local currency.	NUMBER(18,4)	114	N
F_IVL_DEAL_INCM_PU RCH_AMT	The value of deal incomes purchases received, in primary currency.	NUMBER(18,4)	115	N
F_IVL_DEAL_INCM_PU RCH_AMT_LCL	The value of deal incomes purchases received, in local currency.	NUMBER(18,4)	116	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_COST_VAR_AMT	The standard cost change as well as the cost difference between standard cost and transaction cost for transactions such as receiving, RTV and transfers using the standard cost method of accounting, in primary currency.	NUMBER(18,4)	117	N
F_IVL_COST_VAR_AMT_LCL	The standard cost change as well as the cost difference between standard cost and transaction cost for transactions such as receiving, RTV and transfers using the standard cost method of accounting, in local currency.	NUMBER(18,4)	118	N
F_IVL_RTL_COST_VAR_AMT	The cost variance using retail based accounting, in primary currency.	NUMBER(18,4)	119	N
F_IVL_RTL_COST_VAR_AMT_LCL	The cost variance using retail based accounting, in local currency.	NUMBER(18,4)	120	N
F_IVL_MARGIN_COST_VAR_AMT	The cost variance using cost based accounting, in primary currency.	NUMBER(18,4)	121	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_MARGIN_COST_VAR_AMT_LCL	The cost variance using cost based accounting, in local currency.	NUMBER(18,4)	122	N
F_IVL_UP_CHRG_PRFT_AMT	The value of profit up charge costs incurred, in primary currency.	NUMBER(18,4)	123	N
F_IVL_UP_CHRG_PRFT_AMT_LCL	The value of expense up charge costs incurred, in primary currency.	NUMBER(18,4)	124	N
F_IVL_UP_CHRG_EXP_AMT	The value of expense up charge costs incurred, in primary currency.	NUMBER(18,4)	125	N
F_IVL_UP_CHRG_EXP_AMT_LCL	The value of expense up charge costs incurred, in local currency.	NUMBER(18,4)	126	N
F_IVL_TSF_IN_BK_COST_AMT	The value at cost of inventory transferred in through a book transfer, in primary currency.	NUMBER(18,4)	127	N
F_IVL_TSF_IN_BK_COST_AMT_LCL	The value at cost of inventory transferred in through a book transfer, in local currency.	NUMBER(18,4)	128	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_TSF_IN_BK_RTL_AMT	The value at retail of inventory transferred in through a book transfer, in primary currency.	NUMBER(18,4)	129	N
F_IVL_TSF_IN_BK_RTL_AMT_LCL	The value at retail of inventory transferred in through a book transfer, in local currency.	NUMBER(18,4)	130	N
F_IVL_TSF_OUT_BK_COST_AMT	The value at cost of inventory transferred out through a book transfer, in primary currency.	NUMBER(18,4)	131	N
F_IVL_TSF_OUT_BK_COST_AMT_LCL	The value at cost of inventory transferred out through a book transfer, in local currency.	NUMBER(18,4)	132	N
F_IVL_TSF_OUT_BK_RTL_AMT	The value at retail of inventory transferred out through a book transfer, in primary currency.	NUMBER(18,4)	133	N
F_IVL_TSF_OUT_BK_RTL_AMT_LCL	The value at retail of inventory transferred out through a book transfer, in local currency.	NUMBER(18,4)	134	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_INTER_STK_SLS_AMT	The value of cumulative net sales since the last time a physical inventory was taken, in primary currency. It is valued at cost for the cost department and at retail for the retail department.	NUMBER(18,4)	135	N
F_IVL_INTER_STK_SLS_AMT_LCL	The cumulative net sales value since the last time a physical inventory was taken, in local currency. It is valued at cost for the cost department and at retail for the retail department.	NUMBER(18,4)	136	N
F_IVL_INTER_STK_SHRK_AMT	The cumulative estimated (or budgeted) shrinkage value since the last time a physical inventory was taken, in primary currency. It is valued at cost for the cost department and at retail for the retail department.	NUMBER(18,4)	137	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_INTER_STK_SHRK_AMT_LCL	The cumulative estimated (or budgeted) shrinkage value since the last time a physical inventory was taken, in local currency. It is valued at cost for the cost department and at retail for the retail department.	NUMBER(18,4)	138	N
F_IVL_STK_MTD_SLS_AMT	The month-to-date net sales value, in primary currency. It is valued at cost for the cost department and at retail for the retail department.	NUMBER(18,4)	139	N
F_IVL_STK_MTD_SLS_AMT_LCL	The month-to-date net sales value, in local currency. It is valued at cost for the cost department and at retail for the retail department.	NUMBER(18,4)	140	N
F_IVL_STK_MTD_SHRK_AMT	The month-to-date estimated (or budgeted) shrinkage value, in primary currency. It is valued at cost for the cost department and at retail for the retail department.	NUMBER(18,4)	141	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_STK_MTD_SHR K_AMT_LCL	The month-to-date estimated (or budgeted) shrinkage value, in local currency. It is valued at cost for the cost department and at retail for the retail department.	NUMBER(18,4)	142	N
F_IVL_BK_STOCK_RTL _AMT	The value at retail of book stock, in primary currency.	NUMBER(18,4)	143	N
F_IVL_BK_STOCK_RTL _AMT_LCL	The value at retail of book stock, in local currency.	NUMBER(18,4)	144	N
F_IVL_BK_STOCK_COST T_AMT	The value at cost of book stock, in primary currency.	NUMBER(18,4)	145	N
F_IVL_BK_STOCK_COST T_AMT_LCL	The value at cost of book stock, in local currency.	NUMBER(18,4)	146	N
F_IVL_ACTL_STOCK_COST OST_AMT	The value at cost of actual stock, when the physical inventory is taken, in primary currency.	NUMBER(18,4)	147	N
F_IVL_ACTL_STOCK_COST OST_AMT_LCL	The value at cost of actual stock, when the physical inventory is taken, in local currency.	NUMBER(18,4)	148	N

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_ACTL_STOCK_R TL_AMT	The value at retail of actual stock, when the physical inventory is taken, in primary currency.	NUMBER(18,4)	149	N
F_IVL_ACTL_STOCK_R TL_AMT_LCL	The value at retail of actual stock, when the physical inventory is taken, in local currency.	NUMBER(18,4)	150	N

stlblwdm.txt

Business rules:

- This interface file contains stock ledger values for a department, class, subclass and location on a given week.
- This interface file cannot contain duplicate transactions for a dept_idnt, class_idnt, subclass_idnt, loc_idnt and day_dt combination.
- This interface file follows the fact flat file interface layout standard.
- For this interface file, the day_dt represents the end day of a week.
- This interface file does not need to be provided when the stock ledger uses Gregorian time (because this table is not populated).

Name	Description	Data Type/Bytes	Field order	Required field
SBCLASS_IDNT	The unique identifier of the subclass in the product hierarchy.	CHARACTER(4)	1	Yes
CLASS_IDNT	The unique identifier of the class in the product hierarchy.	CHARACTER(4)	2	Yes
DEPT_IDNT	The unique identifier of a department in the product hierarchy.	CHARACTER(4)	3	Yes

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	4	Yes
LOC_TYPE_CDE	The code that indicates whether the location is a store or warehouse.	CHARACTER(2)	5	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	6	Yes
F_IVL_BEG_SOH_COST_AMT	The beginning of period stock on hand total cost, in primary currency.	NUMBER(18,4)	7	No
F_IVL_BEG_SOH_COST_AMT_LCL	The beginning of period stock on hand total cost, in local currency.	NUMBER(18,4)	8	No
F_IVL_BEG_SOH_RTL_AMT	The beginning of period stock on hand total retail, in primary currency.	NUMBER(18,4)	9	No
F_IVL_BEG_SOH_RTL_AMT_LCL	The beginning of period stock on hand total retail, in local currency.	NUMBER(18,4)	10	No
F_IVL_SOH_ADJ_COST_AMT	The value at cost of stock on hand adjustments, in primary currency.	NUMBER(18,4)	11	No
F_IVL_SOH_ADJ_COST_AMT_LCL	The value at cost of stock on hand adjustments, in local currency.	NUMBER(18,4)	12	No
F_IVL_SOH_ADJ_RTL_AMT	The value at retail of stock on hand adjustments, in primary currency.	NUMBER(18,4)	13	No

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_SOH_ADJ_RTL_AMT_LCL	The value at retail of stock on hand adjustments, in local currency.	NUMBER(18,4)	14	No
F_IVL_RCPTS_COST_AMT	The value at cost of inventory received, in primary currency.	NUMBER(18,4)	15	No
F_IVL_RCPTS_COST_AMT_LCL	The value at cost of inventory received, in local currency.	NUMBER(18,4)	16	No
F_IVL_RCPTS_RTL_AMT	The value at retail of inventory received, in primary currency.	NUMBER(18,4)	17	No
F_IVL_RCPTS_RTL_AMT_LCL	The value at retail of inventory received, in local currency.	NUMBER(18,4)	18	No
F_IVL_RTV_COST_AMT	The value at cost of inventory returned to a vendor, in primary currency.	NUMBER(18,4)	19	No
F_IVL_RTV_COST_AMT_LCL	The value at cost of inventory returned to a vendor, in local currency.	NUMBER(18,4)	20	No
F_IVL_RTV_RTL_AMT	The value at retail of inventory returned to a vendor, in primary currency.	NUMBER(18,4)	21	No
F_IVL_RTV_RTL_AMT_LCL	The value at retail of inventory returned to a vendor, in local currency.	NUMBER(18,4)	22	No
F_IVL_TSF_IN_COST_AMT	The value at cost of inventory transferred in, in primary currency.	NUMBER(18,4)	23	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_TSF_IN_COST_AMT_LCL	The value at cost of inventory transferred in, in local currency.	NUMBER(18,4)	24	No
F_IVL_TSF_IN_RTL_AMT	The value at retail of inventory transferred in, in primary currency.	NUMBER(18,4)	25	No
F_IVL_TSF_IN_RTL_AMT_LCL	The value at retail of inventory transferred in, in local currency.	NUMBER(18,4)	26	No
F_IVL_TSF_OUT_COST_AMT	The value at cost of inventory transferred out, in primary currency.	NUMBER(18,4)	27	No
F_IVL_TSF_OUT_COST_AMT_LCL	The value at cost of inventory transferred out, in local currency.	NUMBER(18,4)	28	No
F_IVL_TSF_OUT_RTL_AMT	The value at retail of inventory transferred out, in primary currency.	NUMBER(18,4)	29	No
F_IVL_TSF_OUT_RTL_AMT_LCL	The value at retail of inventory transferred out, in local currency.	NUMBER(18,4)	30	No
F_IVL_SHRK_COST_AMT	The value at cost of the difference between actual and ending inventory, in primary currency.	NUMBER(18,4)	31	No
F_IVL_SHRK_COST_AMT_LCL	The value at cost of the difference between actual and ending inventory, in local currency.	NUMBER(18,4)	32	No

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_SHRK_RTL_AMT	The value at retail of the difference between actual and ending inventory, in primary currency.	NUMBER(18,4)	33	No
F_IVL_SHRK_RTL_AMT_LCL	The value at retail of the difference between actual and ending inventory, in local currency.	NUMBER(18,4)	34	No
F_IVL_RTRNS_COST_AMT	The value at cost of inventory returned from sales, in primary currency.	NUMBER(18,4)	35	No
F_IVL_RTRNS_COST_AMT_LCL	The value at cost of inventory returned from sales, in local currency.	NUMBER(18,4)	36	No
F_IVL_RTRNS_RTL_AMT	The value at retail of inventory returned from sales, in primary currency.	NUMBER(18,4)	37	No
F_IVL_RTRNS_RTL_AMT_LCL	The value at retail of inventory returned from sales, in local currency.	NUMBER(18,4)	38	No
F_IVL_RECLASS_IN_COST_AMT	The value at cost of inventory reclassified to this location, in primary currency.	NUMBER(18,4)	39	No
F_IVL_RECLASS_IN_COST_AMT_LCL	The value at cost of inventory reclassified to this location, in local currency.	NUMBER(18,4)	40	No
F_IVL_RECLASS_IN_RTL_AMT	The value at retail of inventory reclassified to this location, in primary currency.	NUMBER(18,4)	41	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_RECLASS_IN_RTL_AMT_LCL	The value at retail of inventory reclassified to this location, in local currency.	NUMBER(18,4)	42	No
F_IVL_RECLASS_OUT_COST_AMT	The value at cost of inventory reclassified from this location, in primary currency.	NUMBER(18,4)	43	No
F_IVL_RECLASS_OUT_COST_AMT_LCL	The value at cost of inventory reclassified from this location, in local currency.	NUMBER(18,4)	44	No
F_IVL_RECLASS_OUT_RTL_AMT	The value at retail of inventory reclassified from this location, in primary currency.	NUMBER(18,4)	45	No
F_IVL_RECLASS_OUT_RTL_AMT_LCL	The value at retail of inventory reclassified from this location, in local currency.	NUMBER(18,4)	46	No
F_IVL_SLS_COST_AMT	The value at cost of inventory sold, in primary currency.	NUMBER(18,4)	47	No
F_IVL_SLS_COST_AMT_LCL	The value at cost of inventory sold, in local currency.	NUMBER(18,4)	48	No
F_IVL_SLS_RTL_AMT	The value at retail of inventory sold, in primary currency.	NUMBER(18,4)	49	No
F_IVL_SLS_RTL_AMT_LCL	The value at retail of inventory sold, in local currency.	NUMBER(18,4)	50	No
F_IVL_END_SOH_COST_AMT	The end of period stock on hand total cost, in primary currency.	NUMBER(18,4)	51	No

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_END_SOH_COST_AMT_LCL	The end of period stock on hand total cost, in local currency.	NUMBER(18,4)	52	No
F_IVL_END_SOH_RTL_AMT	The end of period stock on hand total retail, in primary currency.	NUMBER(18,4)	53	No
F_IVL_END_SOH_RTL_AMT_LCL	The end of period stock on hand total retail, in local currency.	NUMBER(18,4)	54	No
F_IVL_GRS_PRFT_AMT	The total gross profit amount, in primary currency.	NUMBER(18,4)	55	No
F_IVL_GRS_PRFT_AMT_LCL	The total gross profit amount, in local currency.	NUMBER(18,4)	56	No
F_IVL_CUM_MKON_PCT	The cumulative markon percent.	NUMBER(12,4)	57	No
F_IVL_ADJ_STOCK_COST_AMT	The value at cost of adjusted stock when the physical inventory is taken, in primary currency.	NUMBER(18,4)	58	No
F_IVL_ADJ_STOCK_COST_AMT_LCL	The value at cost of adjusted stock when the physical inventory is taken, in local currency.	NUMBER(18,4)	59	No
F_IVL_ADJ_STOCK_RTL_AMT	The value at retail of adjusted stock when the physical inventory is taken, in primary currency.	NUMBER(18,4)	60	No
F_IVL_ADJ_STOCK_RTL_AMT_LCL	The value at retail of adjusted stock when the physical inventory is taken, in local currency.	NUMBER(18,4)	61	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_MKUP_AMT	The value of upward revisions in price, in primary currency.	NUMBER(18,4)	62	No
F_IVL_MKUP_AMT_LCL	The value of upward revisions in price, in local currency.	NUMBER(18,4)	63	No
F_IVL_MKUP_CNCLLD_AMT	The value of corrections to upward revisions in price, in primary currency.	NUMBER(18,4)	64	No
F_IVL_MKUP_CNCLLD_AMT_LCL	The value of corrections to upward revisions in price, in local currency.	NUMBER(18,4)	65	No
F_IVL_MKDN_CNCLLD_AMT	The value of markdown cancellation to correct an unintentional error in a previous markup, in primary currency.	NUMBER(18,4)	66	No
F_IVL_MKDN_CNCLLD_AMT_LCL	The value of markdown cancellation to correct an unintentional error in a previous markup, in local currency.	NUMBER(18,4)	67	No
F_IVL_PERM_MKDN_AMT	The value of permanent reductions of the price, in primary currency.	NUMBER(18,4)	68	No

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_PERM_MKDN_AMT_LCL	The value of permanent reductions of the price, in local currency.	NUMBER(18,4)	69	No
F_IVL_PRMTN_MKDN_AMT	The value of promotion reductions of the price, in primary currency.	NUMBER(18,4)	70	No
F_IVL_PRMTN_MKDN_AMT_LCL	The value of promotion reductions of the price, in local currency.	NUMBER(18,4)	71	No
F_IVL_CLRC_MKDN_AMT	The value of clearance reductions of the price, in primary currency.	NUMBER(18,4)	72	No
F_IVL_CLRC_MKDN_AMT_LCL	The value of clearance reductions of the price, in local currency.	NUMBER(18,4)	73	No
F_IVL_EMPTY_DISC_AMT	The value of employee discounts, in primary currency.	NUMBER(18,4)	74	No
F_IVL_EMPTY_DISC_AMT_LCL	The value of employee discounts, in local currency.	NUMBER(18,4)	75	No
F_IVL_CASH_DISC_AMT	The value of cash discounts, in primary currency.	NUMBER(18,4)	76	No
F_IVL_CASH_DISC_AMT_LCL	The value of cash discounts, in local currency.	NUMBER(18,4)	77	No
F_IVL_FRGHT_COST_AMT	The value of freight expenses, in primary currency.	NUMBER(18,4)	78	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_FRGHT_COST_AMT_LCL	The value of freight expenses, in local currency.	NUMBER(18,4)	79	No
F_IVL_WRKRM_COST_AMT	The value of workroom expenses, in primary currency.	NUMBER(18,4)	80	No
F_IVL_WRKRM_COST_AMT_LCL	The value of workroom expenses, in local currency.	NUMBER(18,4)	81	No
F_IVL_GAFS_COST_AMT	The goods available for sale valued at cost, in primary currency.	NUMBER(18,4)	82	No
F_IVL_GAFS_COST_AMT_LCL	The goods available for sale valued at cost, in local currency.	NUMBER(18,4)	83	No
F_IVL_GAFS_RTL_AMT	The goods available for sale valued at retail, in primary currency.	NUMBER(18,4)	84	No
F_IVL_GAFS_RTL_AMT_LCL	The goods available for sale valued at retail, in local currency.	NUMBER(18,4)	85	No
F_IVL_SLS_QTY	The number of net units of merchandise sold.	NUMBER(12,4)	86	No
F_IVL_SLS_RTL_EX_VAT_AMT	The value at retail, excluding VAT, of net merchandise sold, in primary currency.	NUMBER(18,4)	87	No
F_IVL_SLS_RTL_EX_VAT_AMT_LCL	The value at retail, excluding VAT, of net merchandise sold, in local currency.	NUMBER(18,4)	88	No

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_FRGHT_CLAIM_RTL_AMT	The value at retail of freight claim, in primary currency.	NUMBER(18,4)	89	No
F_IVL_FRGHT_CLAIM_RTL_AMT_LCL	The value at retail of freight claim, in local currency.	NUMBER(18,4)	90	No
F_IVL_FRGHT_CLAIM_COST_AMT	The value at cost of freight claim, in primary currency.	NUMBER(18,4)	91	No
F_IVL_FRGHT_CLAIM_COST_AMT_LCL	The value at cost of freight claim, in local currency.	NUMBER(18,4)	92	No
F_IVL_IC_TSF_IN_COST_AMT	The value at cost of inventory transferred in for intercompany transfers, in primary currency.	NUMBER(18,4)	93	No
F_IVL_IC_TSF_IN_COST_AMT_LCL	The value at cost of inventory transferred in for intercompany transfers, in local currency.	NUMBER(18,4)	94	No
F_IVL_IC_TSF_IN_RTL_AMT	The value at retail of inventory transferred in for intercompany transfers, in primary currency.	NUMBER(18,4)	95	No
F_IVL_IC_TSF_IN_RTL_AMT_LCL	The value at retail of inventory transferred in for intercompany transfers, in local currency.	NUMBER(18,4)	96	No
F_IVL_IC_TSF_OUT_COST_AMT	The value at cost of inventory transferred out for intercompany transfers, in primary currency.	NUMBER(18,4)	97	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_IC_TSF_OUT_COST_AMT_LCL	The value at cost of inventory transferred out for intercompany transfers, in local currency.	NUMBER(18,4)	98	No
F_IVL_IC_TSF_OUT_RTL_AMT	The value at retail of inventory transferred out for intercompany transfers, in primary currency.	NUMBER(18,4)	99	No
F_IVL_IC_TSF_OUT_RTL_AMT_LCL	The value at retail of inventory transferred out for intercompany transfers, in local currency.	NUMBER(18,4)	100	No
F_IVL_IC_MARGIN_AMT	The margin value of intercompany transfers, in primary currency.	NUMBER(18,4)	101	No
F_IVL_IC_MARGIN_AMT_LCL	The margin value of intercompany transfers, in local currency.	NUMBER(18,4)	102	No
F_IVL_IC_MKDN_RTL_AMT	The markdown at retail of merchandise transferred out for intercompany transfers, in primary currency.	NUMBER(18,4)	103	No
F_IVL_IC_MKDN_RTL_AMT_LCL	The markdown at retail of merchandise transferred out for intercompany transfers, in local currency.	NUMBER(18,4)	104	No

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_IC_MKUP_RTL_AMT	The markup at retail of merchandise transferred out for intercompany transfers, in primary currency.	NUMBER(18,4)	105	No
F_IVL_IC_MKUP_RTL_AMT_LCL	The markup at retail of merchandise transferred out for intercompany transfers, in local currency.	NUMBER(18,4)	106	No
F_IVL_WO_UPD_INV_COST_AMT	The value at cost of merchandise required work order activity, update inventory, for intercompany transfers, in primary currency.	NUMBER(18,4)	107	No
F_IVL_WO_UPD_INV_COST_AMT_LCL	The value at cost of merchandise required work order activity, update inventory, for intercompany transfers, in local currency.	NUMBER(18,4)	108	No
F_IVL_WO_POST_FIN_COST_AMT	The value at cost of merchandise required work order activity, post to financial, for intercompany transfers, in primary currency.	NUMBER(18,4)	109	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_WO_POST_FIN_COST_AMT_LCL	The value at cost of merchandise required work order activity, post to financial, for intercompany transfers, in local currency.	NUMBER(18,4)	110	No
F_IVL_ADJ_COGS_COST_AMT	The value at cost of stock adjustments that affect COGS, in primary currency.	NUMBER(18,4)	111	No
F_IVL_ADJ_COGS_COST_AMT_LCL	The value at cost of stock adjustments that affect COGS, in local currency.	NUMBER(18,4)	112	No
F_IVL_ADJ_COGS_RTL_AMT	The value at retail of stock adjustments that affect COGS, in primary currency	NUMBER(18,4)	113	No
F_IVL_ADJ_COGS_RTL_AMT_LCL	The value at retail of stock adjustments that affect COGS, in local currency	NUMBER(18,4)	114	No
F_IVL_RESTOCK_FEE_AMT	The value at cost of restocking fees received, in primary currency.	NUMBER(18,4)	115	No
F_IVL_RESTOCK_FEE_AMT_LCL	The value at cost of restocking fees received, in local currency.	NUMBER(18,4)	116	No
F_IVL_DEAL_INCM_SLS_AMT	The value of deal incomes sales received, in primary currency.	NUMBER(18,4)	117	No
F_IVL_DEAL_INCM_SLS_AMT_LCL	The value of deal incomes sales received, in local currency.	NUMBER(18,4)	118	No

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_DEAL_INCM_PURCH_AMT	The value of deal incomes purchases received, in primary currency.	NUMBER(18,4)	119	No
F_IVL_DEAL_INCM_PURCH_AMT_LCL	The value of deal incomes purchases received, in local currency.	NUMBER(18,4)	120	No
F_IVL_COST_VAR_AMT	The standard cost change as well as the cost difference between standard cost and transaction cost for transactions such as receiving, RTV and transfers using the standard cost method of accounting, in primary currency.	NUMBER(18,4)	121	No
F_IVL_COST_VAR_AMT_LCL	The standard cost change as well as the cost difference between standard cost and transaction cost for transactions such as receiving, RTV and transfers using the standard cost method of accounting, in local currency.	NUMBER(18,4)	122	No
F_IVL_RTL_COST_VAR_AMT	The cost variance using retail based accounting, in primary currency.	NUMBER(18,4)	123	No
F_IVL_RTL_COST_VAR_AMT_LCL	The cost variance using retail based accounting, in local currency.	NUMBER(18,4)	124	No
F_IVL_MARGIN_COST_VAR_AMT	The cost variance using cost based accounting, in primary currency.	NUMBER(18,4)	125	No

Appendix A – Application programming interface (API) flat file specifications

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_MARGIN_COST_VAR_AMT_LCL	The cost variance using cost based accounting, in local currency.	NUMBER(18,4)	126	No
F_IVL_UP_CHRG_PRFT_AMT	The value of profit up charge costs incurred, in primary currency.	NUMBER(18,4)	127	No
F_IVL_UP_CHRG_PRFT_AMT_LCL	The value of profit up charge costs incurred, in local currency.	NUMBER(18,4)	128	No
F_IVL_UP_CHRG_EXP_AMT	The value of expense up charge costs incurred, in primary currency.	NUMBER(18,4)	129	No
F_IVL_UP_CHRG_EXP_AMT_LCL	The value of expense up charge costs incurred, in local currency.	NUMBER(18,4)	130	No
F_IVL_TSF_IN_BK_COST_AMT	The value at cost of inventory transferred in through a book transfer, in primary currency.	NUMBER(18,4)	131	No
F_IVL_TSF_IN_BK_COST_AMT_LCL	The value at cost of inventory transferred in through a book transfer, in local currency.	NUMBER(18,4)	132	No
F_IVL_TSF_IN_BK_RTL_AMT	The value at retail of inventory transferred in through a book transfer, in primary currency.	NUMBER(18,4)	133	No

Name	Description	Data Type/Bytes	Field order	Required field
F_IVL_TSF_IN_BK_RTL_AMT_LCL	The value at retail of inventory transferred in through a book transfer, in local currency.	NUMBER(18,4)	134	No
F_IVL_TSF_OUT_BK_COST_AMT	The value at cost of inventory transferred out through a book transfer, in primary currency.	NUMBER(18,4)	135	No
F_IVL_TSF_OUT_BK_COST_AMT_LCL	The value at cost of inventory transferred out through a book transfer, in local currency.	NUMBER(18,4)	136	No
F_IVL_TSF_OUT_BK_RTL_AMT	The value at retail of inventory transferred out through a book transfer, in primary currency.	NUMBER(18,4)	137	No
F_IVL_TSF_OUT_BK_RTL_AMT_LCL	The value at retail of inventory transferred out through a book transfer, in local currency.	NUMBER(18,4)	138	No

sttflddm.txt

Business rules

- This interface file contains store traffic information.
- This interface file cannot contain duplicate transactions for a loc_idnt, day_dt combination.
- This interface file follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	1	Yes

DAY_DT	The calendar day on which the transaction occurred.	DATE	2	Yes
F_STORE_TRAFFIC	The number of visitors to a particular store on a certain day.	NUMBER(16,4)	3	No

subtrantypedm.txt

Business rules

- This interface file contains sub-transaction type records.
- This interface file cannot contain duplicate records for a sub_tran_type_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
SUB_TRAN_TYPE_IDNT	The unique identifier of the sub-transaction type.	CHARACTER(6)	1	Yes
SUB_TRAN_TYPE_DESC	The description of the sub-transaction type.	CHARACTER(120)	2	No

supctrdm.txt

Business rules

- This interface file contains supplier contract information.
- This interface file cannot contain duplicate records for a cntrect_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
CNTRCT_IDNT	The unique identifier of a contract.	CHARACTER(6)	1	Yes
SUPP_IDNT	The unique identifier of a supplier.	CHARACTER(10)	2	Yes
STATUS_CDE	The code representing the status for this contract.	CHARACTER(1)	3	Yes

Name	Description	Data Type/Bytes	Field order	Required field
CNTRCT_BEG_DT	The starting date for the contract.	DATE	4	No
CNTRCT_END_DT	The ending date for the contract.	DATE	5	No
CNTRCT_DIST	The distributor name who collects the merchandise from the supplier and delivers to the retailer.	CHARACTER(40)	6	No
CNTRCT_SHIP_MTHD_CDE	The code representing the method of shipment associated with the contract.	CHARACTER(2)	7	No
CNTRCT_SHIP_MTHD_DESC	The description of the method of shipment associated with the contract.	CHARACTER(120)	8	No
STATUS_DESC	The description of the contract status.	CHARACTER(120)	9	No

supsupdm.txt

Business rules

- This interface file contains a record for each supplier, and it holds details of supplier related attributes.
- This interface file cannot contain duplicate records for a supp_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
SUPP_IDNT	The unique identifier of a supplier.	CHARACTER(10)	1	Yes
SUPP_DESC	The supplier's name.	CHARACTER(120)	2	Yes

Name	Description	Data Type/Bytes	Field order	Required field
SUPP_QC_RQRD_IND	Indicates if this supplier's receipts should be checked for quality control.	CHARACTER(1)	3	No
SUPP_PRE_MARK_IND	Indicates whether the items supplied by this supplier will be pre-marked.	CHARACTER(1)	4	No
SUPP_PRE_TICKET_IND	Indicates if the supplier pre-marks or pre-prices his goods.	CHARACTER(1)	5	No
SUPP_STTS_CDE	The code that indicates if the supplier is currently active.	CHARACTER(2)	6	No
SUPP_STTS_DESC	The description of the status code.	CHARACTER(120)	7	No
SUPP_EDT_IND	This column indicates if the supplier has EDI capabilities.	CHARACTER(1)	8	No
SUPP_DOMESTIC_CDE	Supplier's domestic code.	CHARACTER(1)	9	No
SUPP_DOMESTIC_DESC	The description of the supplier's domestic code.	CHARACTER(120)	10	No
SUPP_CRNCY_CDE	The code representing the currency that the supplier operates under.	CHARACTER(3)	11	No
SUPP_CRNCY_DESC	The description of the supplier's currency code.	CHARACTER(120)	12	No
SUPP_VMI_IND	Indicates whether a supplier is vendor managed inventory supplier.	CHARACTER(1)	13	No

suptrmdm.txt

Business rules

- This interface file defines the associations between supplier and supplier trait.

- This interface file cannot contain duplicate records for a supp_trait_idnt, supp_idnt combination.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
SUPP_TRAIT_IDNT	The unique identifier of the supplier trait.	CHARACTER(10)	1	Yes
SUPP_IDNT	The unique identifier of a supplier.	CHARACTER(10)	2	Yes

suptrtdm.txt

Business rules

- This interface file contains supplier trait information.
- This interface file cannot contain duplicate records for a supp_trait_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
SUPP_TRAIT_IDNT	The unique identifier of the supplier trait.	CHARACTER(10)	1	Yes
MAST_SUPP_FLAG	Flag which indicates if this trait is a master supplier trait. Valid values are 'Y' or 'N'.	CHARACTER(1)	2	Yes
SUPP_TRAIT_DESC	The supplier trait description.	CHARACTER(120)	3	No
MAST_SUPP_CDE	The number of the master supplier.	CHARACTER(10)	4	No

time_13.txt

Business rules:

- This text file contains one row for one month of a fiscal calendar year.
- This text file cannot contain duplicate records for the same year, quarter, and month.

Name	Description	Data Type/Bytes	Field Order	Required Field
Year	13-period fiscal calendar year.	NUMBER(4)	1	Yes
Qtr	13-period fiscal quarter; valid values are 1-4.	NUMBER(1)	2	Yes
Month (period)	13-period fiscal period; valid values are 1-13.	NUMBER(2)	3	Yes
First day of the month	The Gregorian date; for example, 20020101 for January 1st 2002.	DATE	4	Yes
Number of weeks	Contains either the number 4 or 5 depending upon whether it is a 4-week or 5-week period.	NUMBER(1)	5	Yes

time_454.txt

Business rules:

- This text file contains one row for one month of a fiscal calendar year.
- This text file cannot contain duplicate records for the same year and month.

Name	Description	Data Type/Bytes	Field Order	Required Field
Year	454 fiscal calendar year.	NUMBER(4)	1	Yes
Month	Month for the fiscal year in the calendar; for example, 1 for January, 12 for December, and so on.	NUMBER(2)	2	Yes
First day of the month	The Gregorian date; for example, 20020101 for January 1st 2002.	DATE	3	Yes
Number of weeks	Contains either the number 4 or 5 depending upon whether it is a 4-week or a 5-week month.	NUMBER(1)	4	Yes
Month description	Calendar month description (January, February, and so on).	CHARACTER(30)	5	Yes

tndrtypdm.txt

Business rules

- This interface file contains tender types and their parent tender type groups.
- This interface file cannot contain duplicate records for a tndr_type_id_idnt, tndr_type_grp_idnt combination.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
TNDR_TYPE_GRP_IDNT	The unique identifier for the tender type group. . An example of a tender type group is cash, check, or credit card.	CHARACTER(6)	1	Yes
TNDR_TYPE_ID_IDNT	The unique identifier for the tender type ID within a tender type group. An example of a tender type ID is Discover Card, Master Card, or Visa	CHARACTER(6)	2	Yes
TNDR_TYPE_GRP_DESC	The description of the tender type group. An example of the description may be "Credit Cards", "Cash", or "Check".	CHARACTER(120)	3	No
TNDR_TYPE_ID_DESC	The description of the tender type ID. An example of the ID description may be "Master Card", "Visa Gold", or American Express Corporate".	CHARACTER(120)	4	No
CASH_EQUIV_FLAG	The indicator of the cash equivalence.	CHARACTER(1)	5	No

ttldmdm.txt

Business rules

- This interface file contains tender type transaction information.

Appendix A – Application programming interface (API) flat file specifications

- This interface file cannot contain duplicate records for tndr_type_group_idnt, tndr_type_id_idnt, tran_idnt, loc_idnt, day_dt, min_idnt, rgstr_idnt, and cshr_idnt combination.
- This interface file follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field order	Required field
TNDR_TYPE_ID_IDNT	The unique identifier for the tender type ID. An example of a tender type ID is Discover Card, Master Card, or Visa.	CHARACTER(6)	1	Yes
TRAN_IDNT	The unique identifier of the transaction.	CHARACTER(30)	2	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	3	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	4	Yes
MIN_IDNT	The unique identifier of the minute.	NUMBER(4)	5	Yes
RGSTR_IDNT	The unique identifier of the register.	CHARACTER(10)	6	Yes
CSHR_IDNT	The unique identifier for a cashier.	CHARACTER(10)	7	Yes

Name	Description	Data Type/Bytes	Field order	Required field
F_CC_SCAN_FLAG	Indicates whether the credit card was scanned or manually entered. Valid values are 'Y' for scanned, or 'N' or Null for manually entered.	CHARACTER(1)	8	No
F_TNDR_COUPON_COUNT	The total count of tender coupons used per transaction. Tender coupons are issues by the manufacturer as opposed to the store.	NUMBER(16,4)	9	No
F_TNDR_COUPON_AMT	The total amount of tender coupons used per transaction. Tender coupons are issues by the manufacturer as opposed to the store.	NUMBER(18,4)	10	No

Name	Description	Data Type/Bytes	Field order	Required field
F_TNDR_COUPON_AMT_LCL	The total amount of tender coupons used per transaction, in local currency. Tender coupons are issued by the manufacturer as opposed to the store.	NUMBER(18,4)	11	No
F_TNDR_SLS_AMT	The sales amount paid for with a particular tender type in primary currency.	NUMBER(18,4)	12	No
F_TNDR_SLS_AMT_LCL	The sales amount paid for with a particular tender type in local currency	NUMBER(18,4)	13	No
F_TNDR_RTRNS_SLS_AMT	The return amount credited to a particular tender type in primary currency.	NUMBER(18,4)	14	No
F_TNDR_RTRNS_SLS_AMT_LCL	The return amount credited to a particular tender type in local currency.	NUMBER(18,4)	15	No

ttltypdm.txt

Business rules

- This interface file contains user-defined totals.
- This interface file cannot contain duplicate records for a total_type_idnt.
- This interface file follows the dimension flat file interface layout standard.
- This interface file contains the complete snapshot of active information.

Name	Description	Data Type/Bytes	Field order	Required field
TOTAL_TYPE_IDNT	The original identifier for the total to be reconciled.	CHARACTER(10)	1	Yes
TOTAL_TYPE_DESC	The description of the total type.	CHARACTER(255)	2	Yes

vchr_age_band_dm.txt

Business rules

- This interface file contains one row for every voucher age band. The voucher age dimension provides a static age band dimension that is used to categorize gift certificates and other vouchers based on their age upon redemption. Each age band is a client-defined range of age, expressed in calendar days. The age of a voucher is used to determine the age band into which it falls.
- Voucher age bands cannot overlap. For example, if voucher age band 1 has a min of 12 and a max of 20, the next age band must have a min of 21 and a max greater than or equal to 21.
- This interface file cannot contain duplicate records for a vchr_age_band_key.
- This data is loaded during installation.

Name	Description	Data Type/Bytes	Field Order	Required Field
VCHR_AGE_BAND_KEY	Surrogate key for the age range into which a voucher falls.	NUMBER(6)	1	Yes
VCHR_AGE_BAND_MIN	The minimum age for a band. The limits to the age band are inclusive. For example, if the age band min is 12 and the max is 20, then all vouchers of age 12 to 20, inclusive of the limits, belong to this age band.	NUMBER(6)	2	Yes

VCHR_AGE_BAND_MAX	The maximum age for a band. The limits to the age band are inclusive. For example, if the age band min is 12 and the max is 20, then all vouchers of age 12 to 20, inclusive of the limits, belong to this age band.	NUMBER(6)	3	Yes
VCHR_AGE_BAND_DESC	The description of the voucher age band. This description can be whatever the client prefers to use to identify an age band. For example, the description could be "Age band 4", "Seven weeks old", or "12 to 20 days".	CHARACTER(30)	4	No

vchreschddm.txt

Business rules

- This interface file contains the date and count of escheated vouchers. When a voucher escheats, the retailer releases all liability of the voucher to the state government. The quantity of escheated vouchers and the dates on which they are escheated are captured from this text file.
- This interface file cannot contain duplicate transactions for a day_dt.
- This interface file follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
DAY_DT	The calendar day on which the transaction occurred.	DATE	1	Yes
F_ESCH_COUNT	The total count of the escheated vouchers on a particular day.	NUMBER(16,4)	2	No
F_ESCH_AMT	The monetary amount of the escheated vouchers. If the voucher was never issued, the escheat amount is 0. If it was issued, the escheat amount is the issue amount.	NUMBER(18,4)	3	No

vchreschddm.txt

Business rules:

- This interface file contains the date and count of escheated vouchers. When a voucher escheats, the retailer releases all liability of the voucher to the state government. The quantity of escheated vouchers and the dates on which they are escheated are captured from this text file.
- This interface file cannot contain duplicate transactions for a day_dt.
- This interface file follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field order	Required field
DAY_DT	The calendar day on which the transaction occurred.	DATE	1	Yes
F_ESCH_COUNT	The total count of the escheated vouchers on a particular day.	NUMBER(16,4)	2	No
F_ESCH_AMT	The monetary amount of the escheated vouchers. If the voucher was never issued, the escheat amount is 0. If it was issued, the escheat amount is the issue amount.	NUMBER(18,4)	3	No

vchrmoveldsgdm.txt

Business rules

- This interface file contains issued and redeemed voucher information at the individual voucher level.
- This interface file cannot contain duplicate transactions for a vchr_line_no, vchr_status_cde combination.
- This interface file follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
VCHR_LINE_NO	The unique identifier for an entry on this table. Corresponds to the unique identifier for a voucher in the source system.	CHARACTER (20)	1	Yes

Name	Description	Data Type/Bytes	Field Order	Required Field
VCHR_STATUS_CDE	Indicates whether this is an issue (I) or redemption (R) record for this voucher.	CHARACTER (1)	2	Yes
LOC_IDNT	The unique identifier of the location.	CHARACTER (10)	3	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	4	Yes
VCHR_AGE	The age of the voucher in days.	NUMBER(6)	5	Yes
TNDR_TYPE_ID_IDNT	The unique identifier for the tender type ID. An example of a tender type ID is Discover Card, Master Card, or Visa.	CHARACTER (6)	6	Yes
RGSTR_IDNT	The unique identifier of the register.	CHARACTER (10)	7	Yes
CSHR_IDNT	The unique identifier for a cashier.	CHARACTER (10)	8	Yes
F_AMT	Amount for which this voucher was issued/redeemed in primary currency.	NUMBER(18, 4)	9	No
F_AMT_LCL	Amount for which this voucher was issued/redeemed in the issue/redemption location's local currency.	NUMBER(18, 4)	10	No

vchroutlwdm.txt

Business rules

- This interface file contains outstanding voucher information 'as of' the day_dt. A voucher is outstanding if it has been issued but not yet redeemed or escheated (that is, fully outstanding).
- This interface file cannot contain duplicate transactions for loc_idnt, week, vchr_age, tn timer_type_id_idnt, rgstr_idnt, cshr_idnt combination.
- This interface file follows the fact flat file interface layout standard.

Name	Description	Data Type/Bytes	Field Order	Required Field
LOC_IDNT	The unique identifier of the location.	CHARACTER(10)	1	Yes
DAY_DT	The calendar day on which the transaction occurred.	DATE	2	Yes
VCHR_AGE	The age of the voucher in days.	NUMBER(6)	3	Yes
TNDR_TYPE_ID_IDNT	The unique identifier for the tender type ID. An example of a tender type ID is Discover Card, Master Card, or Visa.	CHARACTER(6)	4	Yes
RGSTR_IDNT	The unique identifier of the register.	CHARACTER(10)	5	Yes
CSHR_IDNT	The unique identifier for a cashier.	CHARACTER(10)	6	Yes
F_OUT_COUNT	The number of outstanding vouchers in this age band.	NUMBER(16,4)	7	No
F_OUT_AMT	The monetary amount of the outstanding vouchers, in primary currency.	NUMBER(18,4)	8	No
F_OUT_AMT_LCL	The monetary amount of the outstanding vouchers, in local currency.	NUMBER(18,4)	9	No

wkday.txt

Business rules

- This text file contains only one record. That record displays the day description of the first day of the week.

Name	Description	Data Type/Bytes	Field Order	Required Field
WKDAY_DESC	The description of the weekday number, calendar weekday.	CHARACTER (120)	1	Yes