SAS Publishing

SAS® Detail Data Store for Retail 2.2

Implementation and Administration Guide

The Power to Know®

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2006. *SAS® Detail Data Store for Retail 2.2: Implementation and Administration Guide, First Edition*. Cary, NC: SAS Institute Inc.

SAS® Detail Data Store for Retail 2.2: Implementation and Administration Guide, First Edition

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, April 2006

Contents

Chapter 1 Introduction to the SAS Detail Data Store for Retail 1
Introduction to This Guide 1
What Is the SAS Detail Data Store for Retail? 2
Benefits of Implementing the SAS Detail Data Store for Retail 3
Customizing the Retail Detail Data Store 3

Chapter 2 Physical Design of the SAS Detail Data Store for Retail 5
Overview 5
Populating the Retained Key 5
Tracking Historical Data over Time 6
Using Effective Dates 6
Using Processed Date/Time 7
Using Natural/Business Keys 7

Chapter 3 Organization of Tables in the SAS Detail Data Store for Retail 9
Overview 9
More Information 11

Chapter 4 Defining Role-Based Users and Groups for the Retail Detail Data Store 13
Overview 13
Planning User Groups and Roles 13
Sample User Groups and Roles 14

Chapter 5 Creating and Registering Tables in the SAS Detail Data Store for Retail 17
Overview 17
Prerequisites 17
Installing the Metadata Import-Export Tool 17
Verification of Installation 18
Creating the Retail DDS 18
Registering the Metadata 19
Editing the Library Path 24
Creating a Subset Version of the Retail DDS Physical Model 26
Verifying the Metadata Registration 27

Chapter 6 Customizing the SAS Detail Data Store for Retail 29
Overview 29
Modifying the Retail DDS 30
Using Caution 30
Avoiding Trouble 30
Things to Keep in Mind 31
Modifying SAS Data Integration Studio Jobs for Retail DDS Changes 31

Chapter 7 Logical Model Report 33
Overview 33
Viewing the Logical Model Report 33
Chapter 8 Support Resources 35
Appendix 1 Table Loading Sequence 37
Appendix 2 Table Types in the SAS Detail Data Store for Retail 43
Transactional Tables 43
Master Tables 43
Reference Tables 44
Intersection Tables 44
Association Tables 44
Appendix 3 Standard Data Types and Naming Conventions 45
Appendix 4 Data Model Notation 47
Reading the Relationship Notation 47
Understanding a One-to-One Relationship 47
Understanding a One-to-Many Relationship 48
Glossary 51

CHAPTER 1

Introduction to the SAS Detail Data Store for Retail

Introduction to This Guide *1*
Prerequisite Knowledge *1*
What Is the SAS Detail Data Store for Retail? *2*
Benefits of Implementing the SAS Detail Data Store for Retail *3*
Customizing the Retail Detail Data Store *3*

Introduction to This Guide

This guide provides the following information to help the SAS Support Consultant and customer system administrator install, configure, and administer the SAS Detail Data Store for Retail:

- a high-level introduction to the retail detail data store (DDS)
- a description of the physical design and table structure of the retail DDS
- information about deploying and loading the retail DDS
- information about maintaining the retail DDS
- information about using the retail DDS with various SAS solutions

Prerequisite Knowledge

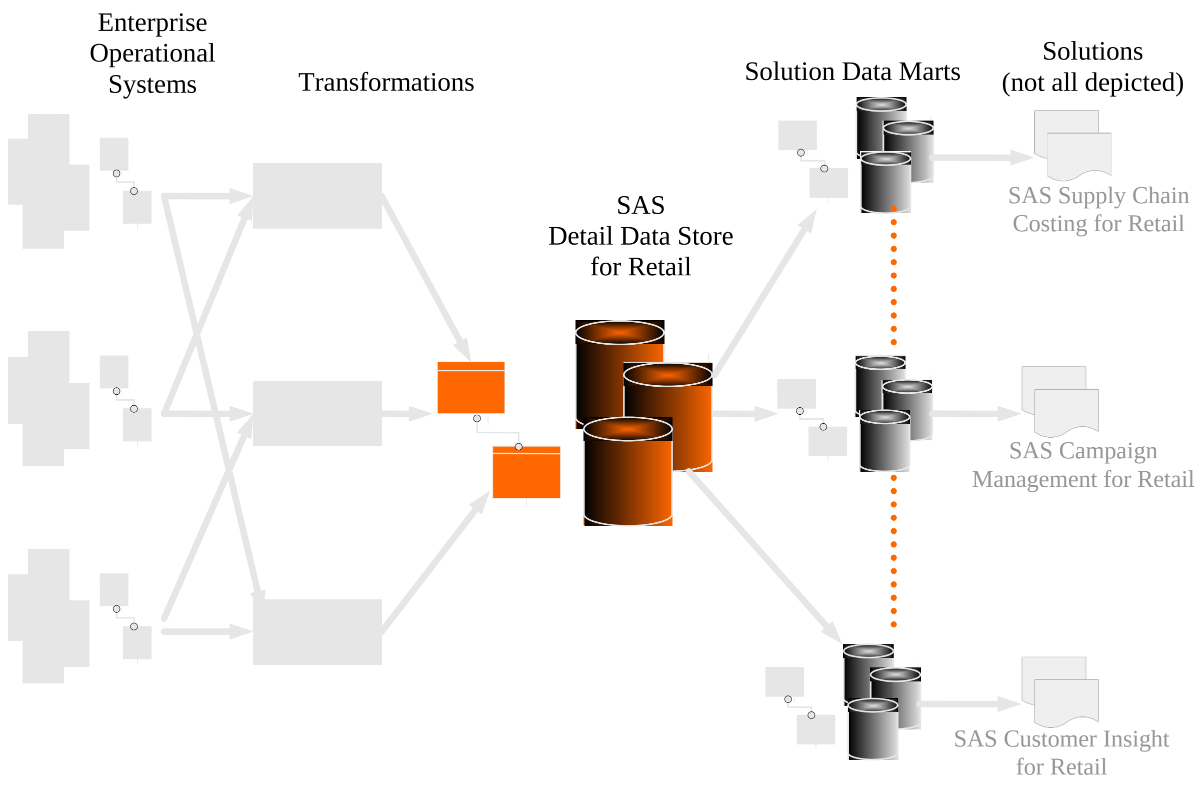
Users of this guide should be familiar with general database technology, data warehousing, and data modeling concepts. The DDS implementers should have administrative and programming experience with Base SAS software and SAS Data Integration Studio (formerly named SAS ETL Studio).

Note: To implement the SAS Detail Data Store for Retail, SAS 9.1.3 or later is required.

What Is the SAS Detail Data Store for Retail?

The retail DDS is a data store that serves as the *single version of the truth* for the SAS Retail Intelligence Solutions, as well as the gateway to SAS Retail solutions. As such, it contains the atomic-level data that is needed to populate the solution data marts. Figure 1.1 illustrates the high-level role of the SAS Detail Data Store for Retail within the Integrated Solutions Data Architecture.

Figure 1.1 Integrated Solutions Data Architecture


Enterprise Operational Systems
Transformations
Solution Data Marts
Solutions (not all depicted)
SAS Detail Data Store for Retail
SAS Supply Chain Costing for Retail
SAS Campaign Management for Retail
SAS Customer Insight for Retail

In the previous figure, data from multiple source systems gets merged into the SAS Detail Data Store for Retail. Then, from the SAS Detail Data Store for Retail, various SAS Retail solutions work off this common data layer.

Benefits of Implementing the SAS Detail Data Store for Retail

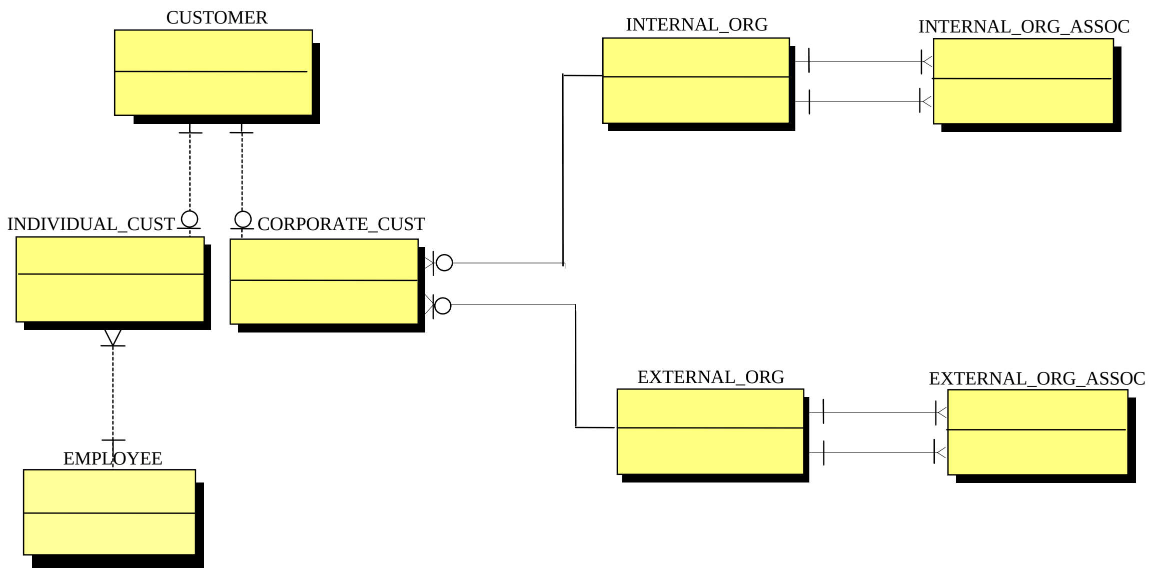
Implementing the retail DDS at a customer site provides several benefits:

- The retail DDS provides a single data target for loading data. For example, a customer first implements SAS Customer Insight for Retail. During the implementation, the customer populates the Item table. Following the SAS Customer Insight for Retail implementation, the customer implements SAS Transaction Insight for Retail. Because the Item table has already been populated with cleansed data, the customer can use this same table when loading the SAS Transaction Insight for Retail data mart. Therefore, populating a single definition of an Item table ensures that both SAS solutions have a single version of the truth.
- Because the definition of the retail DDS tables is already known, the extract, transform, and load (ETL) process from the retail DDS to the solution data marts will be pre-built.
- Data that is created from SAS Retail solutions can be stored in a central location and shared with other solutions. For example, the retail outlet segment scores from SAS Transaction Insight for Retail are written back to the retail DDS and shared with other SAS Retail solutions.

Customizing the Retail Detail Data Store

No predesigned data model will meet all of an organization's needs. Customizations will need to be applied to the retail DDS in almost every implementation. For example, Figure 1.2 shows how the emphasis on the **EMPLOYEE** table has been minimized because the existing solutions do not require much information about employees. However, in the future, if a customer wants to implement effective labor-management retail solutions that require detailed modeling of a retailer's employees, the customer can add tables and attributes to the existing data model. Chapter 6, "Customizing the SAS Detail Data Store for Retail" provides more information about customizing the retail DDS.

Figure 1.2 Customization of the Retail DDS



CHAPTER 2

Physical Design of the SAS Detail Data Store for Retail

Overview 5
Populating the Retained Key 5
Tracking Historical Data over Time 6
Using Effective Dates 6
Using Processed Date/Time 7
Using Natural/Business Keys 7

Overview

The retail DDS is designed as an integration and storage layer for operational (or source) systems. As such, the retail DDS is a lightly denormalized, relational data model that is flexible for storage. A key difference between a source system and the retail DDS is that the retail DDS captures current and historical data. Many of the retail DDS design decisions are based on the need for this historical data.

Populating the Retained Key

A retained key is a generated surrogate key that uniquely identifies a record at that point in time. Because data is coming from multiple source systems, the business key is not sufficient to identify the records. The retained key (_RK) field, which is part of the primary key, is populated with a retained surrogate (generated) key. This numeric key does not change for the different versions of the record.

For example, in the INTERNAL_ORG table in Figure 2.1, the **ORGANIZATION_NM** column changes from `Marketing` to `World Wide Marketing`, but the retained key (shown in the first column) stays as `100`. The only change in the old record is in the **VALID_TO_DTTM** column, which now correctly shows the changed date.

Figure 2.1 Example of Using a Retained Key

Old record

INTERNAL_ORG_RK	VALID_FROM_DTTM	VALID_TO_DTTM	ORGANIZATION_NM
100	01JAN1999:12:00:00	31DEC4747:00:00:00	Marketing

Old record updated, and new record populated with retained key

INTERNAL_ORG_RK	VALID_FROM_DTTM	VALID_TO_DTTM	ORGANIZATION_NM
100	01JAN1999:12:00:00	31DEC2000:23:59:59	Marketing
100	01JAN2001:00:00:00	31-DEC-4747 00:00:00	World Wide Marketing

Tracking Historical Data over Time

In Figure 2.1, the VALID_FROM_DTTM and VALID_TO_DTTM values are used with the _RK value to track historical data over time. These values define the time period during which the contents of the row are valid. You should set the VALID_TO_DTTM value to a date far into the future for ease of joins. If the source system does not track historical data for records, the VALID_FROM_DTTM and VALID_TO_DTTM values would correspond to the date and time that the DDS was loaded. However, if the source system did track historical data (for example, multiple changes of a row between load times of the DDS), there could be more than one row for the same retained key for a given load of the DDS.

The design decision about how the date and time values are managed is related to the deployment of the system. The date and time values can be tied to business system dates if they are provided by the business system, and if they do not conflict with their primary purpose in tracking the different versions of the system.

VALID_FROM values can be created from the following:

- retail DDS load date and time
- ETL date and time
- business system record timestamps

The primary purpose of the VALID_FROM and VALID_TO values is tracking versions, which is the only use of these values that is guaranteed. If the dates are used for another purpose (such as data extract date or business system entry date), then this usage must not compromise the primary purpose.

Using Effective Dates

Some types of data have EFFECTIVE_FROM and EFFECTIVE_TO dates that indicate when a particular piece of business information is in effect. These date values are different from the date values that are used to track versions. For example, a retailer could load all of the planned promotions into the PROMO table with its EFFECTIVE_FROM date time and EFFECTIVE_TO date time in one load.

In Figure 2.2, a retailer plans periodic promotions and loads them into the PROMO table.

Figure 2.2 Example of EFFECTIVE_FROM and EFFECTIVE_TO Dates

During first load of PROMO table

PROMO_RK	VALID_FROM_DTTM	VALID_TO_DTTM	EFFECTIVE_FROM_DTTM	EFFECTIVE_TO_DTTM
100	01JAN2005:00:00:00	31DEC4747:00:00:00	01MAR2005:00:00:00	31MAR2005:00:00:00

PROMO business validity changes subsequently

PROMO_RK	VALID_FROM_DTTM	VALID_TO_DTTM	EFFECTIVE_FROM	EFFECTIVE_TO
100	01JAN2005:00:00:00	31JAN2005:23:59:59	01MAR2005:00:00:00	31MAR2005:00:00:00
100	01FEB2005:00:00:00	31DEC4747:00:00:00	01MAR2005:00:00:00	15MAR2005:00:00:00

The EFFECTIVE_FROM_DTTM and EFFECTIVE_TO_DTTM date values provide effective business periods for the data in this version of the row. If these date values do not offer sufficient historical data with the VALID_FROM and VALID_TO date values, then you can add additional business-system-related dates, subject to naming and other standards.

Using Processed Date/Time

Knowing the last time that a row was processed in the retail DDS is useful. Processing can include initially creating a row or updating a row, such as adding the VALID_TO_DTTM value to an existing row. The PROCESSED_DTTM value determines which rows have changed since they were loaded into the data mart by determining the last time that the record was touched by an ETL process or by the data administrator, which includes unusual updates that do not change the row (such as error-correction data-patching).

You can populate the PROCESSED_DTTM value by using the **Load Time Column** option that is available in the Loader Transformations (SCD Type 2 Loader) of SAS Data Integration Studio.

Using Natural/Business Keys

In the retail DDS, retaining the primary source system identifier (also known as the natural key or business key) and the retained keys in the rows of the tables is useful. The standard for capturing the natural/business key in the retail DDS is <*TABLE_NAME*>_ID. Figure 2.3 shows the retained key for a SUPPLIER table and the natural/business key. These natural/business keys originate from the source systems and typically contain long alphanumeric strings. Retaining all keys is useful if you need to go back to the source systems to investigate data.

Figure 2.3 Retaining the Natural/Business Key

SUPPLIER_RK	VALID_FROM_DTTM	VALID_TO_DTTM	SUPPLIER_ID
1001	01JAN2005:00:00:00	31DEC4747:00:00:00	S0000000109

CHAPTER 3

Organization of Tables in the SAS Detail Data Store for Retail

Overview ... *9*
More Information ... *11*

Overview

At a high level, the retail DDS can be grouped into subject areas. In Figure 3.1, the data model consists of the following subject areas:

Customer

This subject area includes the customers who are involved in retail, such as the supertype Customer table, the Individual_Customer table, the Corporate_Customer table, and the associated reference tables. Customer information can include details of individual and corporate customers, associated addresses and contact information, household information for individuals, organization information for corporate customers, and information about the segments to which a household or customer belongs. Customer information is obtained from the operational management systems and the marketing or customer relationship management systems. Customer information can include orders that are placed by a customer via mail order systems. Additionally, customer information can have information pertaining to retailer-provided loyalty card transactions and customer-related transactions.

Facility

This subject area includes the different facilities used by a retailer or by a third party, including facility processes under a facility. A facility can be a distribution center for a retailer where a retailer maintains its inventory and services other retail outlets by supplying goods.

Census

This subject area includes external market data sources, such as Nielsen. Census information includes customer counts in different groups, such as industry, education, age, household, occupation, and so on.

Customer Relationship Management

This subject area includes information from customer relationship management (CRM) systems, such as SAS Marketing Automation. CRM information includes:

- campaigns and communications storage
  (available from marketing or CRM systems and loaded into the retail DDS if SAS Campaign Management for Retail is implemented)
- customer contacts and responses
- surveys that are conducted by the retailer
  (available from proprietary systems or data feeds)
- customer-segmentation information
- event information (such as address changes, marriages, job changes) that is actively tracked for a customer
  (available from account management and transactional systems)

Item

This subject area includes the corporate item master and item information, such as price history for an item, promotion for the item for a specific time period, product hierarchy for an item, and so on.

Location

This subject area includes the various retail locations. A location could be a physical geographical location or a virtual selling location. Location information includes tables that maintain different hierarchies for different types of retail locations.

Order

This subject area includes information about the different orders that are placed by customers through the various ordering channels, such as the phone, internet, and catalog. Additionally, order information includes the address information where the customer wants the order delivered.

Organization

This subject area includes information about retail organization. Organization could be internal or external to the retailer. Organization information is stored hierarchically.

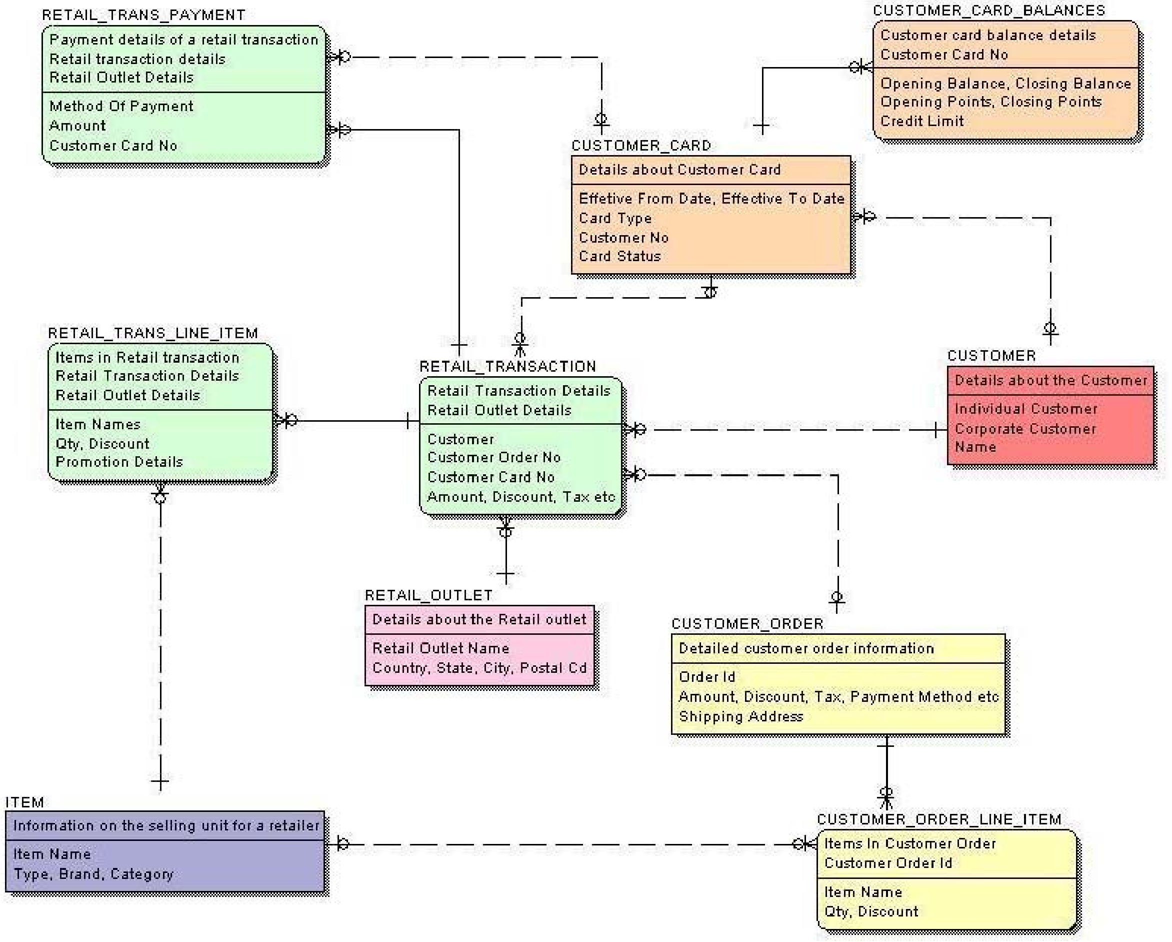
Supply Chain

This subject area includes information about item supply chain for a retailer. The supply chain process has been modeled from supplier to delivery through different delivery routes. This supply chain group of tables is used for deriving direct and indirect costs that are incurred for an item through different delivery routes at various time periods.

Retail Transaction

This subject area includes information about items that are transacted through different sales channels, such as cash registers in a retail outlet, bulk orders for corporate deals, transactions on the internet, and transactions at customer service points (for unsatisfactory reasons, such as returns, faulty goods, and exchanges).

Figure 3.1 High-Level View of the SAS Detail Data Store for Retail



Note: The figure is for illustration purposes only. Refer to the retail DDS data model for the most accurate information about the data model.

More Information

The following appendices include more information about retail DDS table organization:

- Appendix 2, “Table Types in the SAS Detail Data Store for Retail” describes the five types of tables used in the retail DDS.
- Appendix 3, “Standard Data Types and Naming Conventions” describes the standard data types and naming conventions used in the retail DDS.
- Appendix 4, “Data Model Notation” describes the special notations used in the retail DDS data model.

CHAPTER 4

Defining Role-Based Users and Groups for the Retail Detail Data Store

Overview 13
Planning User Groups and Roles 13
Sample User Groups and Roles 14
Metadata Administrator 15
DDS Administrator 15
ETL Administrator 15
Information Architect 15
Power Users 16

Overview

The tasks in this chapter describe how to define metadata user roles, set up appropriate user IDs, and assign the IDs to groups. This chapter provides a recommended role-based setup guide for retail DDS metadata users. Although potential roles are recommended, each business should define its own roles based on the specialized functions that are required.

Planning User Groups and Roles

The steps for planning user groups and roles are the following:

1 Define user roles based on business and functional needs.

2 Determine which user accounts you must establish in the metadata for the business and functional needs.

3 Decide how to organize users into the appropriate groups.

When setting up user roles, you should analyze the business and functional needs that pertain to the retail DDS setup and usage. Administrative user roles should be robust, and power user roles should be limited, but not limiting. User roles should have the correct balance between deployability, usability, and maintainability for your metadata.

Consider the various business and functional tasks that need to be accomplished to set up the proper user environment for the retail DDS. For example, the following are some of the required business and functional tasks:

- setting up the SAS Metadata Server environment
- registering and administering the metadata
- maintaining, scheduling, and running the ETL jobs
- creating information maps
- creating, running, and viewing reports

After you define the business and functional tasks, create the user IDs and permissions that are necessary to perform the tasks.

It is a best practice to organize users into functional groups because it will simplify the process of establishing and managing the users. Assigning permissions to users on an individual basis can be cumbersome.

After you define the user groups, assign permissions to groups.

There are two user groups that are automatically created in the foundation repository: PUBLIC and SASUSERS. Membership in the PUBLIC and SASUSERS groups is implicit. If you can access the SAS Metadata Server, then you are automatically a member of the PUBLIC group. If you can access the SAS Metadata Server and you have your own metadata identity, then you are automatically a member of both the PUBLIC group and the SASUSERS group.

Now, define the user roles and create the groups that are necessary to assist with the administration of the retail DDS.

Sample User Groups and Roles

The information in this section is a suggested method of implementing roles and groups for your retail DDS. This is not the only way to establish metadata user IDs.

The sample groups, which are derived from the perspective of a production environment, are listed:

- Metadata Administrator – This role acts as overall administrator for the metadata repository. It is not given access to the directories containing data. It is the unrestricted user.
- DDS Administrator – This role deploys retail DDS and ETL jobs and provides production support.
- ETL Administrator – This role develops, schedules, and manages ETL jobs and provides production support.
- Information Architect – This role creates information maps using the retail DDS data files. It coordinates with the DDS Administrator to set up privileges for Power Users.
- Power Users – This role queries the information maps that have been set up. Access for this role is granted by the DDS Administrator in consultation with the Information Architect.

These are sample groups. In reality, there could be a developer group and many types of power users. For the purpose of this section, users and groups are defined based on these functional tasks and the defined roles.

Metadata Administrator

The Metadata Administrator is the unrestricted user for your SAS installation. For a default SAS installation, this user is sasadm. This user has unrestricted privileges to the metadata repository and has the ability to make changes to the SAS installation.

The Metadata Administrator is responsible for the following:

- installing the SAS server and other SAS software
- setting up scripts or services to start and stop the servers
- creating the foundation repository
- setting up and registering the various application servers within the foundation repository
- creating administrative users (for example, the DDS Administrator)

DDS Administrator

The DDS Administrator represents all of the administrative users. This role is created by the Metadata Administrator and deploys the retail DDS and ETL jobs.

The DDS Administrator is responsible for the following:

- creating retail DDS physical data files
- creating the library and registering retail DDS metadata
- creating any other libraries that are needed by the deployment
- importing ETL jobs and External File Interface (EFI) objects
- creating other users and groups, such as ETL Administrator and Power Users
- assigning appropriate permissions to the user accounts
- creating deployment directories

ETL Administrator

The ETL Administrator develops, schedules, and manages ETL jobs.

The ETL Administrator is responsible for the following:

- verifying that deployment has worked properly
- deploying ETL jobs
- making changes to the ETL jobs using SAS Data Integration Studio
- scheduling ETL jobs using LSF Scheduler or any other third-party schedulers
- providing production support to resolve any problems in the data
- reporting on job status by directly accessing the data sets that are created for this purpose

Information Architect

The Information Architect is involved in the design, creation, and deployment of information maps for Power Users using SAS Information Map Studio.

The Information Architect is responsible for the following:

- creating and maintaining information maps
- working with DDS Administrators to set up privileges for Power Users

Power Users

The Power Users are users who consume the information maps that are generated by the Information Architects. Power Users query the retail DDS based on the information maps.

In reality, there could be different types of Power Users. You might need different Power Users to focus on different business areas. For example, you might set up a customer Power User to focus on customer accounts, and a logistics Power User to focus on supply chain-related information. Access for these Power Users is restricted to the tables that they will need.

Power Users are responsible for the following:

- running information maps that have been assigned to the Power User
- working with the Information Architect to update or request new information maps

CHAPTER 5

Creating and Registering Tables in the SAS Detail Data Store for Retail

Overview 17
Prerequisites 17
Installing the Metadata Import-Export Tool 17
Verification of Installation 18
Creating the Retail DDS 18
Registering the Metadata 19
Editing the Library Path 24
Creating a Subset Version of the Retail DDS Physical Model 26
  Modifying the %DDLDDS Macro 26
  Selective Registration 27
Verifying the Metadata Registration 27

Overview

This chapter details how to instantiate the retail DDS. This process includes the steps to create the physical tables and register the metadata in the SAS Metadata Repository.

Prerequisites

Complete the following before creating and registering the retail DDS tables:

1. All SAS products and solutions must be correctly installed by following the installation instructions that are shipped with the products and solutions.
2. SAS Data Integration Studio must be set up and functional. “Setup Tasks for Administrators” in the *SAS Data Integration Studio: User’s Guide* outlines all of the required tasks prior to using SAS Data Integration Studio. These same tasks are required prior to loading some of the components of the retail DDS.
3. The SAS Metadata Server and a metadata repository must be operational.

Installing the Metadata Import-Export Tool

The Metadata Import-Export Tool (MIET) - a plug-in for SAS Data Integration Studio - must be installed to register metadata. If the MIET plug-in is not available as part of the solution’s installation package, it can be downloaded from the SAS Customer Support Center at: http://support.sas.com. From this page, select **Documentation →**

Products & Solutions and then select **SAS Detail Data Store for Retail** from the **Select a Product** menu. The MIET plug-in can be found under the **Metadata Registration and DDS Upgrade Tools** section of the **SAS Detail Data Store for Retail Resource Center**.

To install the MIET:

1 Download the MIET and save it to your local directory.

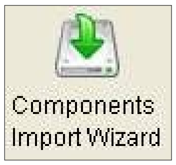
2 Extract the .jar file to the SAS Data Integration Studio plug-ins directory. On a Windows-based computer, the plug-ins directory is usually located at **C:\Program Files\SAS\SASETLStudio\9.1\plugins**.

Note: **Metadata_Import_913_V1.2.jar** works with SAS 9.1.2 ETL Studio (renamed SAS Data Integration Studio) and **Metadata_Import_914_V1.2.jar** works with SAS 9.1.3 Data Integration Studio 3.2.

Verification of Installation

After you install the MIET, SAS Data Integration Studio must be restarted (closed and opened again). The MIET installation can be verified by the following methods.

The following shortcut is in the SAS Data Integration Studio shortcut bar.



The following menu option is in the SAS Data Integration Studio **Tools** menu.

Creating the Retail DDS

DDL CREATE TABLE statements instantiate the retail DDS physical model. The %DDLDDS macro creates the table definitions. %INCLUDE statements in this macro reference <*tablename*>.sas files that contain the CREATE TABLE statements to instantiate the physical model.

A default DATE9 informat/format is applied to date fields in the supplied DDL. For customer sites with different date requirements, the macro variable DTFMT can be changed to represent the locale (such as EURDFDE*w*.). The macro variable is referenced in the invocation string for the %DDLDDS macro, which is detailed below.

To create the DDS tables, do the following:

1 Modify and execute the following SAS %DDLDDS macro code (available as part of the retail DDS installation):

```
 /* Macro variables to be assigned:
Fileloc: Assign fileloc macro parameter to point to the directory which
contains the DDL create table statements contained in <tablename>.sas
files.
```

```
DTFMT: (Optional: Defaults to DATE9.) If international date values are
needed, assign a new DTFMT= value, such as dtfmt=EURDFDEw. */

%include "<location_of_provided_macro>/ddldds.sas";
LIBNAME DDS "<location_to_store_tables>";
%ddldds(fileloc=<location_of_SAS_DDL>, dtfmt=<International_Date Value>);
```

2 Verify that the tables were created properly by issuing the following SAS code:

```
proc datasets lib=DDS; quit;
```

The DATASETS procedure output can be compared with the retail DDS physical model.

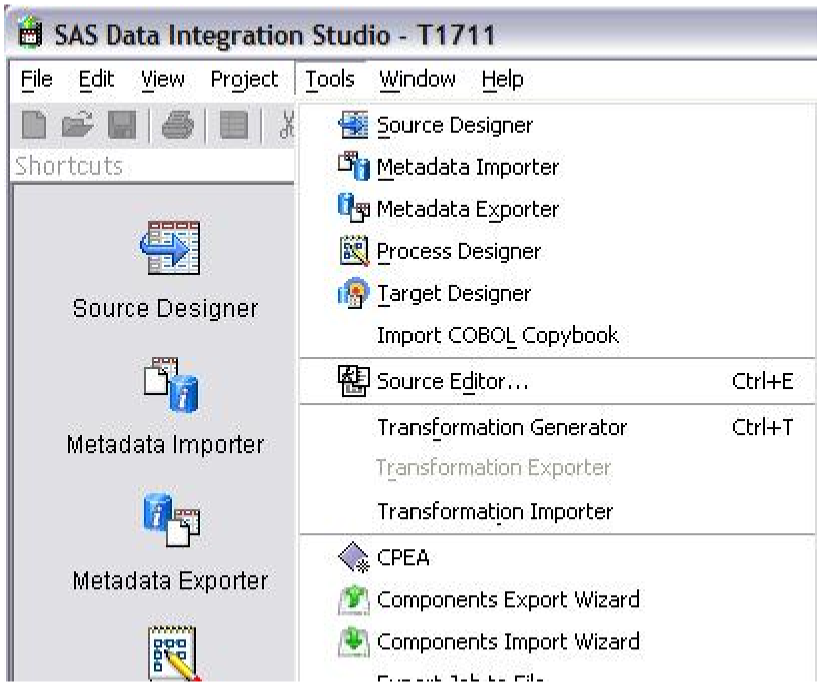
The full set of DDS tables that make up the retail DDS data model is now instantiated.

Registering the Metadata

One file per table is registered in the metadata. The tables, library, and pre-created custom folders are shipped in the installation package and must be imported to the customer's production repository using the MIET, which is available under the **Tools** menu in the main menu of SAS Data Integration Studio.

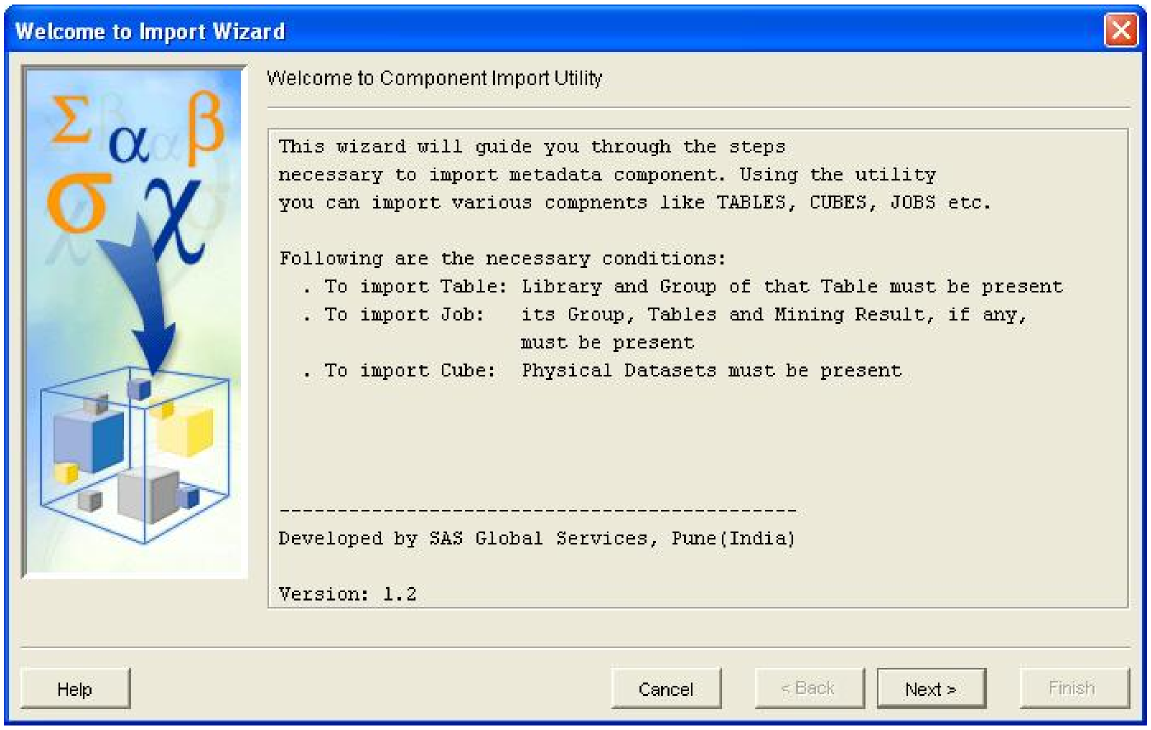
1 In SAS Data Integration Studio, select **Tools → Components Import Wizard**.

Figure 5.1 Accessing the Components Import Wizard



2 When you select the Components Import Wizard, a Welcome page opens.

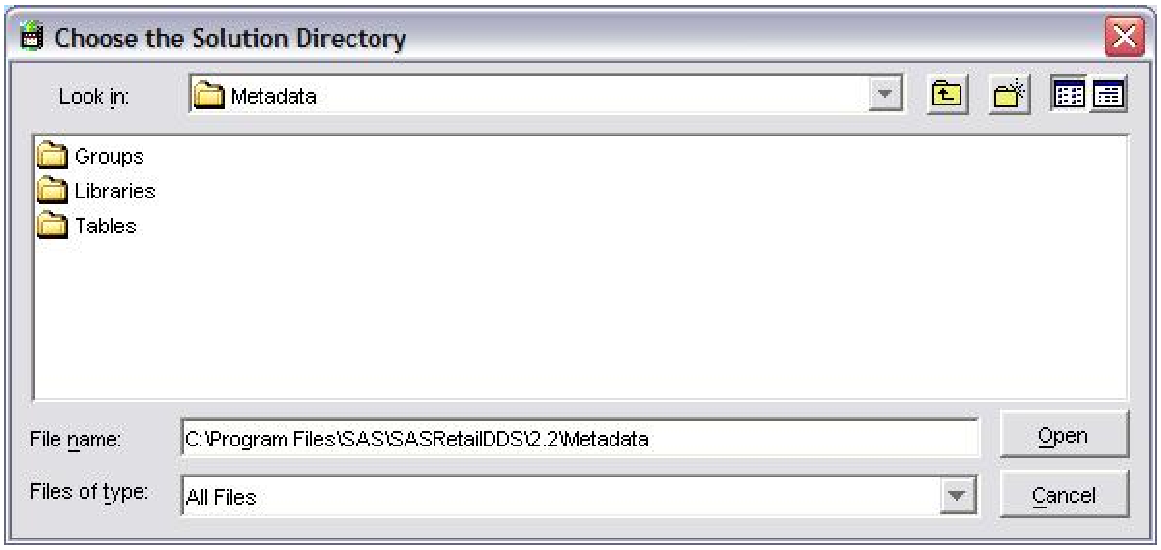
Figure 5.2 Welcome Page



```
This wizard will guide you through the steps
necessary to import metadata component. Using the utility
you can import various compnents like TABLES, CUBES, JOBS etc.

Following are the necessary conditions:
  . To import Table: Library and Group of that Table must be present
  . To import Job:   its Group, Tables and Mining Result, if any,
                     must be present
  . To import Cube:  Physical Datasets must be present




---------------------------------------------
Developed by SAS Global Services, Pune(India)

Version: 1.2
```

3 Click **Next** to move to the Choose the Solution Directory page.

Figure 5.3 Choose the Solution Directory Page



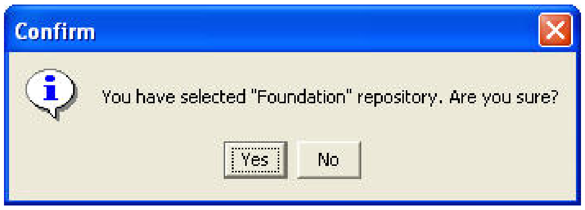
Look in: Metadata

File name: C:\Program Files\SAS\SASRetailDDS\2.2\Metadata

Files of type: All Files

4 Click **Open** and select the objects from the folder where the retail DDS is installed. The default installation directory is **C:\Program Files\SAS\SASRetailDDS\2.2\Metadata**. Choose the repository from the menu and click **Next**. The following message appears:

Figure 5.4 Confirmation Message



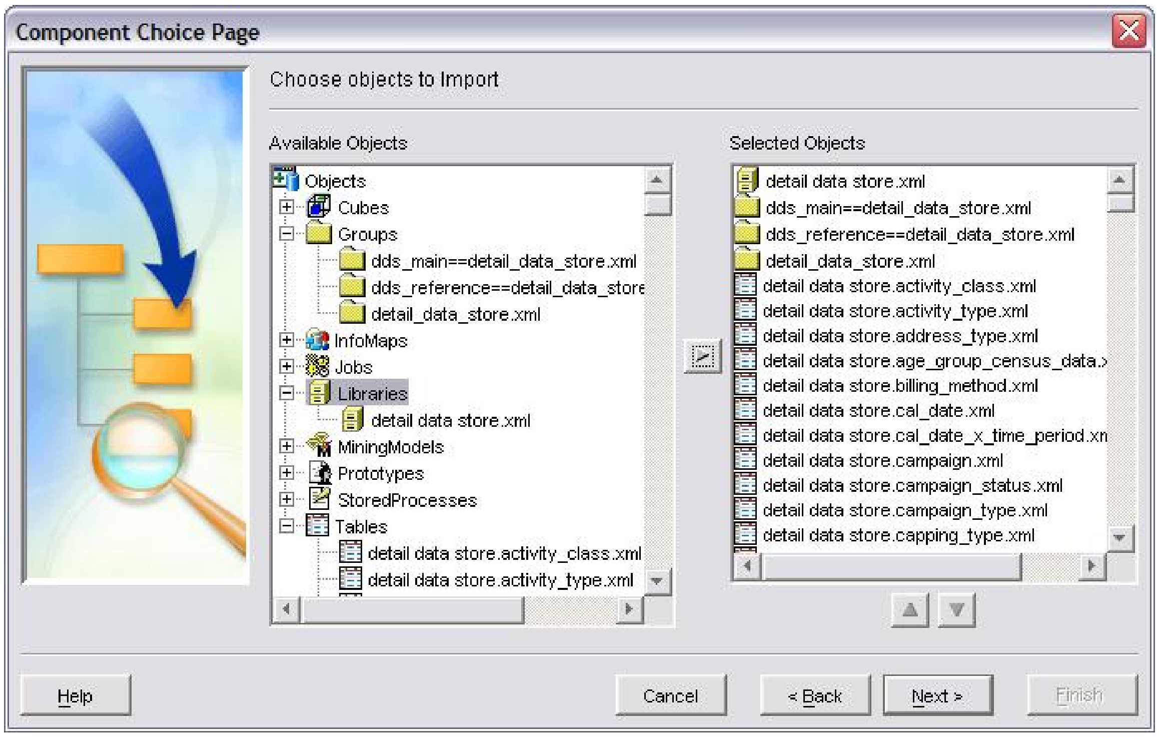
You have selected "Foundation" repository. Are you sure?

5 Click **Yes** to move to the Component Choice Page. If you have not selected the right repository, click **No** to go back and change the repository selection.

6 On the Component Choice Page, a tree view of the objects is shown. Select the objects for import and click the **Add** button ▻ .

Select only those objects that are required for the metadata registration process – **Groups** (known as custom folders in SAS Data Integration Studio), **Libraries**, and **DDS Tables** – and import them to the Foundation repository. The following figure shows that some groups and tables have been added.

Figure 5.5 Component Choice Page



Available Objects

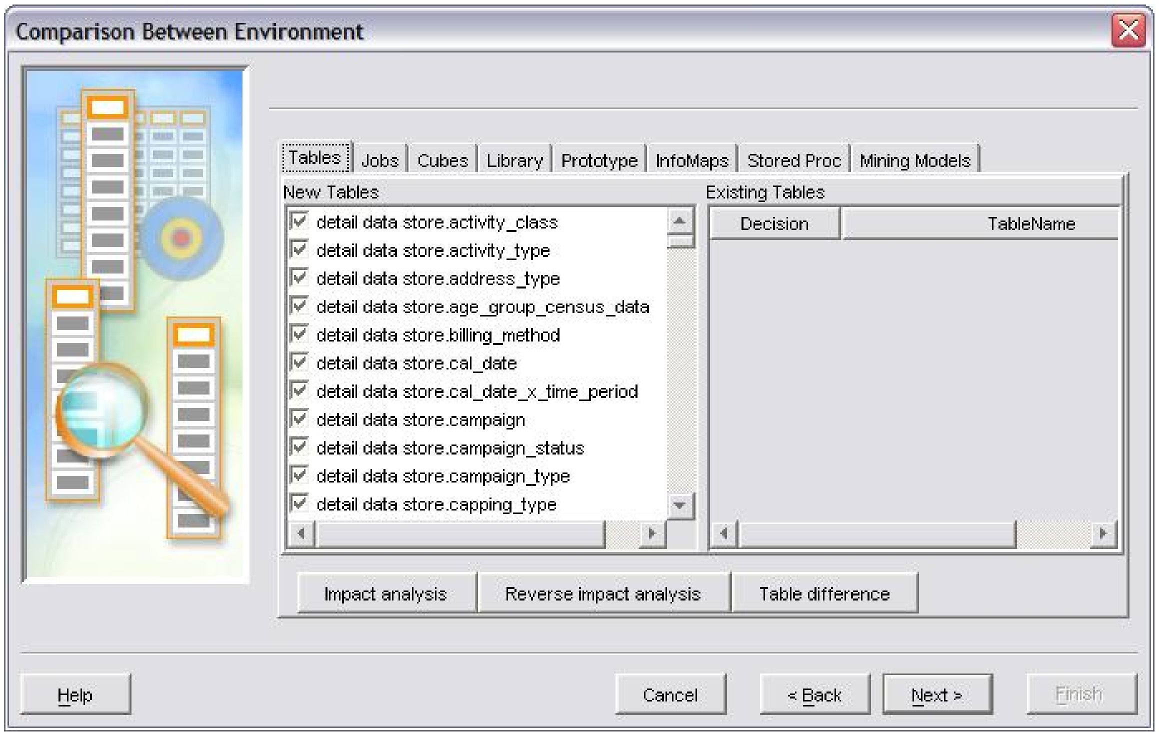
- Objects
  - Cubes
  - Groups
    - dds_main==detail_data_store.xml
    - dds_reference==detail_data_store
    - detail_data_store.xml
  - InfoMaps
  - Jobs
  - Libraries
    - detail data store.xml
  - MiningModels
  - Prototypes
  - StoredProcesses
  - Tables
    - detail data store.activity_class.xml
    - detail data store.activity_type.xml

Selected Objects

- detail data store.xml
- dds_main==detail_data_store.xml
- dds_reference==detail_data_store.xml
- detail_data_store.xml
- detail data store.activity_class.xml
- detail data store.activity_type.xml
- detail data store.address_type.xml
- detail data store.age_group_census_data.›
- detail data store.billing_method.xml
- detail data store.cal_date.xml
- detail data store.cal_date_x_time_period.xn
- detail data store.campaign.xml
- detail data store.campaign_status.xml
- detail data store.campaign_type.xml
- detail data store.capping_type.xml

7 Click **Next** to move to the Comparison Between Environment page.

Figure 5.6 Comparison Between Environment Page



New Tables

- ☑ detail data store.activity_class
- ☑ detail data store.activity_type
- ☑ detail data store.address_type
- ☑ detail data store.age_group_census_data
- ☑ detail data store.billing_method
- ☑ detail data store.cal_date
- ☑ detail data store.cal_date_x_time_period
- ☑ detail data store.campaign
- ☑ detail data store.campaign_status
- ☑ detail data store.campaign_type
- ☑ detail data store.capping_type

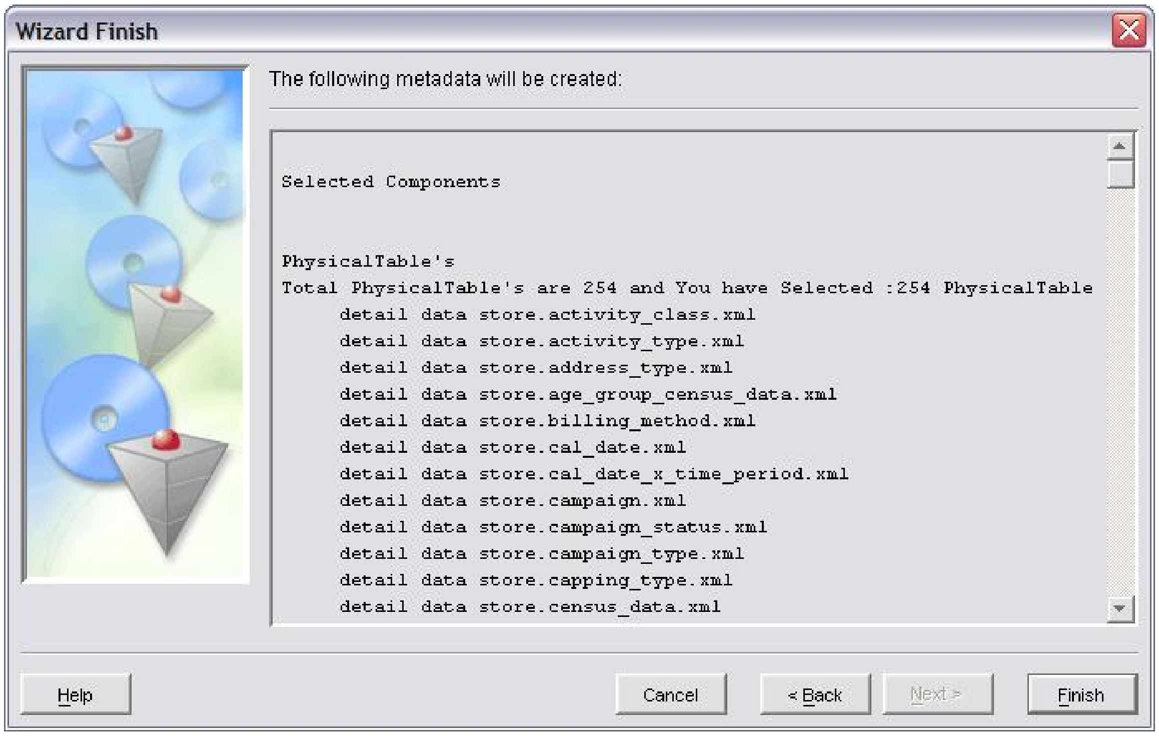
Existing Tables

Decision	TableName

Each component, such as a table or a library, appears under its respective tab. Groups does not have a tab. You can deselect the tables and libraries that you do not need.

8 Click **Next** to move to the Wizard Finish page.

Figure 5.7 Wizard Finish Page



The following metadata will be created:

```
Selected Components


PhysicalTable's
Total PhysicalTable's are 254 and You have Selected :254 PhysicalTable
     detail data store.activity_class.xml
     detail data store.activity_type.xml
     detail data store.address_type.xml
     detail data store.age_group_census_data.xml
     detail data store.billing_method.xml
     detail data store.cal_date.xml
     detail data store.cal_date_x_time_period.xml
     detail data store.campaign.xml
     detail data store.campaign_status.xml
     detail data store.campaign_type.xml
     detail data store.capping_type.xml
     detail data store.census_data.xml
```

The Wizard Finish page gives information on the metadata that is being created. If you would like to make changes, click **Back** to change the information. If the information is correct, click **Finish**. The selected objects will be added to the Foundation repository.

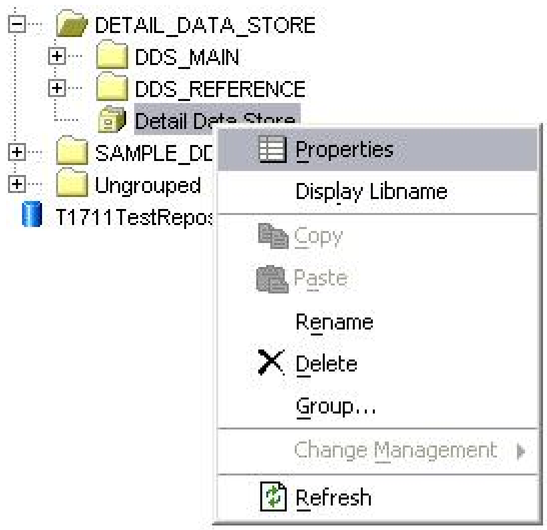
9 Select **View → Refresh** from the SAS Data Integration Studio main menu to see the updated Foundation repository that contains the created objects.

Editing the Library Path

After you register the metadata, you need to edit the path of the imported library to point to the physical location of the retail DDS.

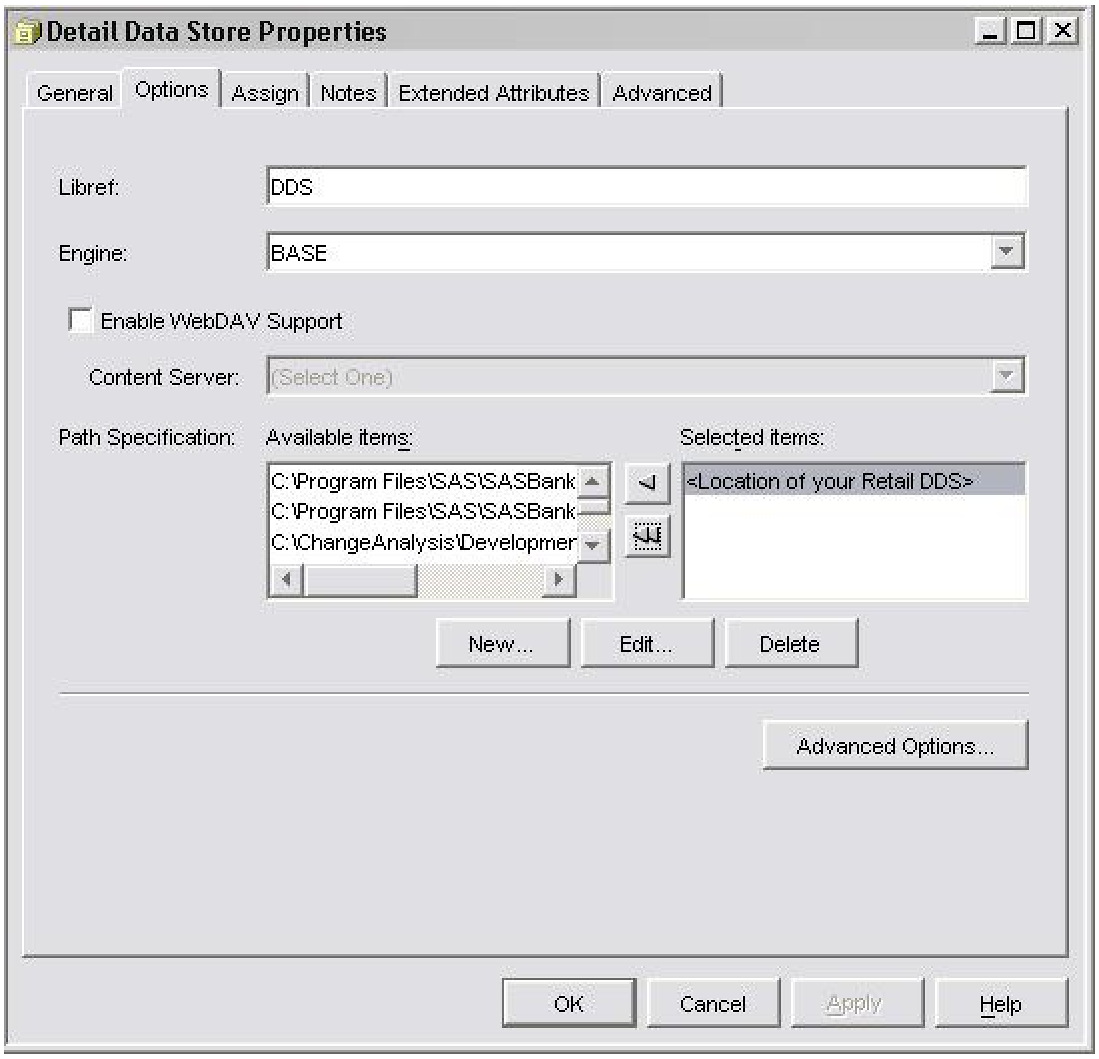
1 In SAS Data Integration Studio, expand the **DETAIL_DATA_STORE** library. Right-click **Detail Data Store** and select **Properties**, as shown in the following figure.

Figure 5.8 Edit the Path in SAS Data Integration Studio



2 Click the **Options** tab to display the Detail Data Store Properties dialog box.

Figure 5.9 Detail Data Store Properties Dialog Box



Libref: DDS

Engine: BASE

Enable WebDAV Support

Content Server: (Select One)

Path Specification: Available items:

C:\Program Files\SAS\SASBank
C:\Program Files\SAS\SASBank
C:\ChangeAnalysis\Developmer

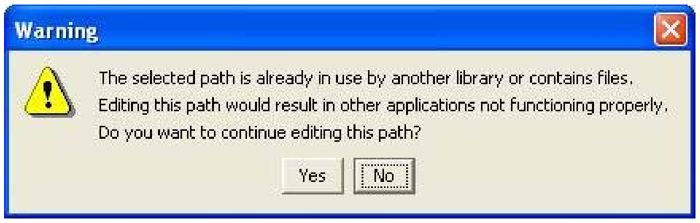
Selected items:

<Location of your Retail DDS>

Select the path under **Selected items** (which is **<Location of your Retail DDS>**) and click **Edit**.

3 Click **Yes** when the Warning dialog box appears.

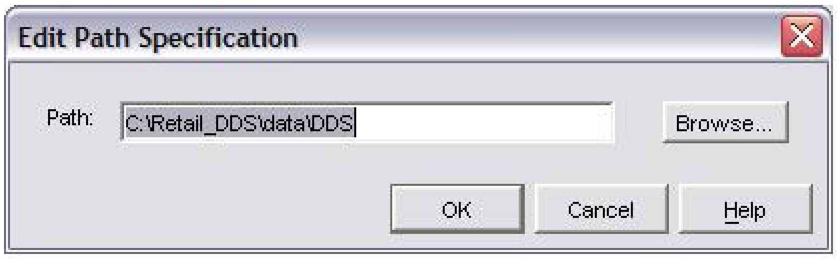
Figure 5.10 Warning Dialog Box



The selected path is already in use by another library or contains files. Editing this path would result in other applications not functioning properly. Do you want to continue editing this path?

This opens the Edit Path Specification dialog box. Replace the existing path with the path where the retail DDS is located, for example **C:\Retail_DDS\data\DDS** as shown in the following figure.

Figure 5.11 Edit Path Specification Dialog Box



Path: C:\Retail_DDS\data\DDS

4 Click **OK** in the Edit Path Specification dialog box and click **OK** in the Detail Data Store Properties dialog box.

Creating a Subset Version of the Retail DDS Physical Model

If the entire retail DDS model is not instantiated, you can modify the %DDLDDS macro and select from the MIET only the tables that you need to create and register for the solution that is being installed.

Modifying the %DDLDDS Macro

For example, if the SUPPLIER and SUPPLIER_DEAL tables are not required for the current retail DDS implementation, you can remove the %INCLUDE statements for these tables from the %DDLDDS macro (as shown in the italicized bold lines in the following code):

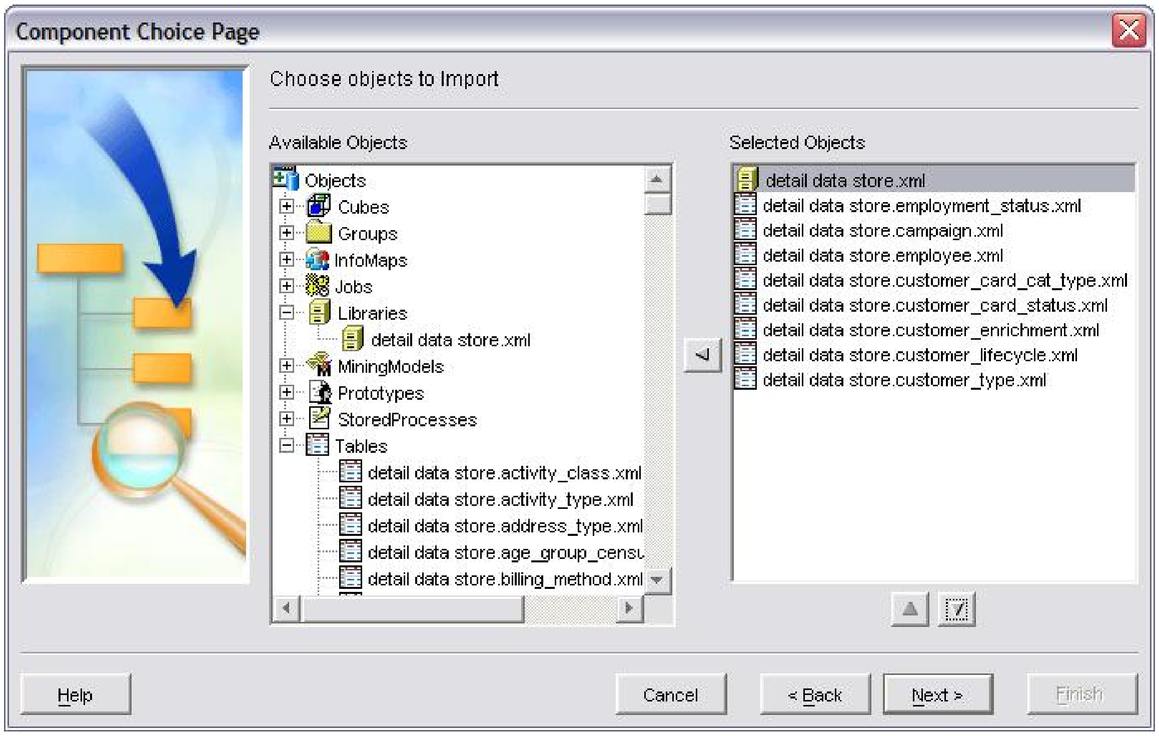
```
%macro ddldds(LIBREF=DDS,DTTMFMT=NLDATM21., DTFMT=DATE9., FMTRK=12.,
fileloc=);
PROC SQL;
  %include "&fileloc./section.sas";
  %include "&fileloc./supplier.sas";          ← Remove
  %include "&fileloc./sub_fixture.sas";
  %include "&fileloc./substitute_item.sas";
  %include "&fileloc./time_period.sas";
  %include "&fileloc./ time_period_assoc.sas";
  %include "&fileloc./ supplier_deal.sas";        ← Remove
.
.
  %include "&fileloc./vehicle.sas";
  %include "&fileloc./weather.sas";
QUIT;
%MEND;
```

If you want to import only the tables that are needed for a specific solution, follow the steps in the next section.

Selective Registration

On the Component Choice Page (also shown in Figure 5.5), select only those tables that are needed for a specific solution and click the **Add** button. Follow the remaining steps after Figure 5.5 in the "Registering the Metadata" section.

Figure 5.12 Component Choice Page



Available Objects:
- detail data store.xml
- detail data store.activity_class.xml
- detail data store.activity_type.xml
- detail data store.address_type.xml
- detail data store.age_group_censu
- detail data store.billing_method.xml

Selected Objects:
- detail data store.xml
- detail data store.employment_status.xml
- detail data store.campaign.xml
- detail data store.employee.xml
- detail data store.customer_card_cat_type.xml
- detail data store.customer_card_status.xml
- detail data store.customer_enrichment.xml
- detail data store.customer_lifecycle.xml
- detail data store.customer_type.xml

Verifying the Metadata Registration

You can use one of the following methods to verify that the metadata was registered properly.

- Review the log file that was produced by the MIET. This log file captures whether the import was successful and reports any errors that were found during the process.
- Connect to the SAS Metadata Server or the metadata repository as the DDS Administrator (using either SAS Management Console or SAS Data Integration Studio) and visually verify that the proper table metadata was created in the library.

CHAPTER 6

Customizing the SAS Detail Data Store for Retail

Overview 29
Modifying the Retail DDS 30
Using Caution 30
Avoiding Trouble 30
Things to Keep in Mind 31
Modifying SAS Data Integration Studio Jobs for Retail DDS Changes 31
Adding Tables 31
Adding Columns 32
Changing Column Length 32

Overview

Initially, the retail DDS scope can be larger than the scope of the data that is to be subsequently loaded. New data might be required or a new SAS solution might be needed that requires data that was not previously loaded into the retail DDS. You might need to populate tables and fields in the retail DDS that exist, but were not previously required.

An issue to consider is time consistency. If new data will not be loaded to the same historical point in time as the current retail DDS, then that fact must be "known" to the application that is accessing the data.

New data table- or field-level requirements imply the following:

- a change to data source system extract files
- a change to SAS Data Integration Studio jobs
- a possible change to dependencies, DDS tables, and indices
- the need for new SAS Data Integration Studio jobs for downstream applications (for example, the solution's applications that created the new data requirements)

Understanding the data flow from the data source to the retail DDS means understanding the impact of adding new data requirements. Identify each affected component, starting with data source system extracts, and modify accordingly in a test environment.

Note: Contact your SAS Support Consultant to get a copy of the retail DDS Computer Associates AllFusion ERwin Data Modeler source if it is needed for implementation.

Modifying the Retail DDS

Typically, you can expect to make the following modifications during an implementation:

- adding local language or value format requirements
- adding fields in the tables because of business requirements
- adding an entity or table because of a new area of business
- customizing the language code (LANGUAGE_CD) to be specific to a country
- customizing most of the amount values that use currencies (CURRENCY_CD) to be specific to a country (however, you should not use mixed currencies for different tables within the retail DDS because this can result in confusion during reporting)
- reproducing warehouse results for a particular date
  The retail DDS supports two date pairs for its tables: the business effective date pair (EFFECTIVE_FROM_DTTM and EFFECTIVE_TO_DTTM), and the warehouse effective date pair (VALID_FROM_DTTM and VALID_TO_DTTM). Both date pairs are maintained because the data might not arrive in the warehouse on the business effective date. If you do not need this functionality (for example, reference tables), you could set VALID_FROM_DTTM to have the same value as EFFECTIVE_FROM_DTTM and VALID_TO_DTTM to have the same value as EFFECTIVE_TO_DTTM.
- changing column labels without any impact

Using Caution

- You can add fields to tables; however, if you use these fields in downstream processes that have already been implemented, you will need to customize the downstream processes.
- You cannot reduce the length of fields because it can impact downstream processing.
- You can carefully expand the length of fields.
- You can add tables; however, if you use these tables in downstream processes that have already been implemented, you will need to customize the downstream processes. If you add a new business area, you should add new processes instead of modifying existing processes.

Avoiding Trouble

- Do not change column names because this action can have significant impact on downstream processes that have already been implemented.
- Do not change data types for columns because this action can have significant impact on downstream processes that have already been implemented.

- Do not delete columns. Columns that are not required by the client can hold missing values.

Things to Keep in Mind

- Develop and test modifications to the retail DDS in a controlled development or test environment before you migrate the modifications to a production environment.
- You can add tables or columns or change the length of a column within the SAS Data Integration Studio environment. See the next section “Modifying SAS Data Integration Studio Jobs for Retail DDS Changes” for more information.
- If you change supplied DDL files, back up or copy the original DDL files in case you need to get back to a known starting point.
- Identify the job dependencies and job order for new jobs and schedule jobs accordingly.

Modifying SAS Data Integration Studio Jobs for Retail DDS Changes

Changes to the retail DDS must be properly handled in the SAS Data Integration Studio jobs that depend on them. Changes are categorized as follows:

- adding tables
- adding columns
- changing column length

Adding Tables

To add a table to the retail DDS, perform the following steps:

1. Open SAS Data Integration Studio and use the developer user ID to connect to the metadata repository.
2. Open the wizard. Select **Tools→Source Designer**.
3. Select **SAS** from the list of **Sources** and click **Next**.
4. Select the SAS library that includes the new table and click **Next**.
5. Select the new table that you want to add and click **Next**.
6. Select the group in which you want the table to be listed and click **Next**.
7. Select **Finish**. The wizard imports the table into the metadata repository.
8. Create a SAS Data Integration Studio job for the new table and add relevant transformations to map the source table columns to the target table columns.

Adding Columns

To add a column to the retail DDS, perform the following steps:

1. Open SAS Data Integration Studio and use the developer user ID to connect to the metadata repository.
2. Select the table in which you want to add a column.
3. Right-click the table and select **Update Table Metadata**. This menu option runs a synchronization routine that updates the stored table metadata to match the table definitions that are in the physical location of the table.
4. A warning appears that selecting this menu option will perform the requested update and might require additional changes to jobs that contain this table. Click **Yes** to continue.
5. Modify all of the SAS Data Integration Studio jobs where the table is used. To see the list of the SAS Data Integration Studio jobs where the table is used, select the **Impact Analysis** or **Reverse Impact Analysis** menu option (depending on whether the table is a source table or a target table) in SAS Data Integration Studio. Identify which SAS Data Integration Studio jobs need to be modified to incorporate this change.
6. Map the column so that it can be used in the transformations.

Changing Column Length

To change the length of a column in the retail DDS, perform the following steps:

1. Open SAS Data Integration Studio and use the developer user ID to connect to the metadata repository.
2. Select the table that includes the column that you want to change.
3. Right-click the table and select **Update Table Metadata**.
4. A warning appears that selecting this menu option will perform the requested update and might require additional changes to jobs that contain this table. Click **Yes** to continue.
5. Modify all of the SAS Data Integration Studio jobs where the table is used. To see the list of the SAS Data Integration Studio jobs where the table is used, select the **Impact Analysis** or **Reverse Impact Analysis** menu option (depending on whether the table is a source table or a target table) in SAS Data Integration Studio. Identify which column lengths need to be changed and manually perform all the transformations for this column. These changes are reflected in the intermediary tables (which are temporary tables that are created by the transformations).

CHAPTER 7

Logical Model Report

Overview *33*
Viewing the Logical Model Report *33*

Overview

This chapter explains how to set up and view the retail DDS logical model report. This report provides a subject area view of the retail DDS, which includes the entities, attributes, key groups, and relationships that make up the retail DDS.

Viewing the Logical Model Report

HTML files and other supporting files are provided in a specific directory of your installation. In a default Windows installation, the directory is **C:\Program Files\SAS\SASRetailDDS\2.2\Doc\LogicalModel** with a subdirectory of **\Images**. Both of these directories and all of the files contained within these directories should be loaded to your Web server, maintaining the source directory's structure.

After the files are loaded to your Web server, the logical model report can be invoked by opening the file **1a_retaildds_22_logical_model.htm**, which is available in the **LogicalModel** directory. The logical model report will open in a separate Web browser.

Retail business views are provided by the following logical model subject areas:

- Logical Census
- Logical CRM
- Logical Customer
- Logical Facility
- Logical Item
- Logical Location
- Logical Order
- Logical Organization
- Logical Supply Chain
- Logical Transaction

In the **'Picture' section** of the logical model report, double-clicking on a table displays a dialog box that shows the name and definition of the table. Double-clicking a relationship line in the diagram displays a dialog box that shows the Parent to Child Phrase and the Parent Entity definition.

The **'Entity' section** of the logical model report provides a listing of tables, their definitions, and links to the child relationships for each table, including assertion rules.

CHAPTER 8

Support Resources

For solution-specific technical support, see the SAS Technical Support Web site at **http://support.sas.com/**.

In addition to documentation and technical support, SAS Professional Services supports the following areas:

- consulting services
- solution assessment, including feasibility and methodology for implementing SAS Retail solutions
- training
- DDS customization and implementation
- project management

The data model for the retail DDS in Computer Associates AllFusion ERwin Data Modeler format is available by request.

APPENDIX 1

Table Loading Sequence

The retail DDS table loading sequence is listed in the following table. Tables should be loaded in the numerical order of their wave numbers, as shown in the following table.

Tables Loaded in Wave 1
ACTIVITY_CLASS
ADDRESS_TYPE
BILLING_METHOD
CAL_DATE
CAMPAIGN_STATUS
CAMPAIGN_TYPE
CAPPING_TYPE
CODE_LANGUAGE
COLOR
COMMUNICATION_CHANNEL
COMMUNICATION_STATUS
COMMUNICATION_TYPE
CONTACT_ROLE
COUNTRY
CREDIT_RATING
CURRENCY
CURRENCY_EXCH_RATE_SRC
CURRENCY_EXCH_RATE_TYPE
CUSTOMER_ACTIVE
CUSTOMER_CARD_CAT_TYPE
CUSTOMER_CARD_STATUS
CUSTOMER_CARD_TYPE
CUSTOMER_LIFECYCLE
CUSTOMER_TYPE
DEAL
DELAY_REASON
DELIVERY_LINE_ITEM_CATEGORY
DELIVERY_TYPE
DESIGNATION
DISCOUNT_REASON
DISCOUNT_TYPE
DIVISION_TYPE
EAN
EDUCATION_LEVEL

Tables Loaded in Wave 1
EMPLOYEE_UNION
EMPLOYMENT_STATUS
ENRICHMENT_DATA_PUB_TYPE
ENRICHMENT_DATA_TYPE
EPC
ETHNICITY
EVENT_CATEGORY
EVENT_STATUS
EVENT_TYPE
EXTERNAL_ORG_ASSOC_TYPE
FACILITY_AREA_TYPE
FACILITY_FORMAT
FACILITY_TYPE
FIXTURE_TYPE
FLAVOR
GENDER
GEO_DEMOGRAPHIC
INCOME_CATEGORY
INDUSTRY
INTERNAL_ORG_ASSOC_TYPE
INVOICE_TYPE
ITEM_BUNDLE_TYPE
ITEM_CATEGORY_ASSOC_TYPE
ITEM_CATEGORY_CODE
ITEM_CATEGORY_TYPE
ITEM_GROUP
ITEM_INVENTORY_STATUS
ITEM_PACK_CONFIG_ASSOC_TYPE
ITEM_PACK_TYPE
ITEM_REGIME_TYPE
ITEM_SIZE
ITEM_STATUS
ITEM_TYPE
LEGAL_ENTITY_TYPE
LOCATION_ASSOC_TYPE
LOCATION_TYPE
MAKE_OR_BUY
MARITAL_STATUS
MINOR_ACTIVITY_TYPE
ORDER_METHOD
ORDER_STATUS
ORDER_TYPE
ORGANIZATION_TYPE
OWN_OR_RENT
PACK_CONFIG
PAYMENT_METHOD
PAYMENT_TERMS

Tables Loaded in Wave 1
PERIOD_TYPE
PERMANENCE
PLANOGRAM
POLICY_TYPE
POS_TERMINAL_TYPE
PO_STATUS
PRICE_INDEX_TYPE
PRICE_TYPE
PRIZM
PROCESS_TYPE
PROMO_TYPE
PROMO_VEHICLE_TYPE
QUALITY
QUALITY_ASSESSMENT
RADIO_REGION
RECEIPT_STATUS
REGION
RESPONSE_CHANNEL
RESPONSE_RULE
RESPONSE_TYPE
RETAIL_OUTLET_CATALOG_TYPE
RETAIL_OUTLET_FORMAT
RETAIL_OUTLET_GRADE
RETAIL_OUTLET_GROUP
RETAIL_OUTLET_LOCATION
RETAIL_OUTLET_TYPE
RETAIL_TRANSACTION_TYPE
RETURN_REASON
REVISION_LEVEL
SEASON
SERVICE_PRIORITY
SERVICE_STATUS
SERVICE_TYPE
SHAPE_TYPE
SHRINKAGE_TYPE
SOURCE_SYSTEM
SPECIAL_NEEDS
SPEND_TYPE
STANDARD_OCCUPATION
STATE_REGION
STOCK_PROFILE
STYLE
SUB_FIXTURE_REGIME_TYPE
SUPPLIER_TYPE
TAX_BRACKET
TIME_PERIOD_ASSOC_TYPE
TIME_PERIOD_TYPE

Tables Loaded in Wave 1
TIME_UOM
TRANSACTION_TYPE
TRANSPORTATION_MODE
TV_REGION
UNIT_OF_MEASURE
UNSPSC
VEHICLE_CATEGORY

Tables Loaded in Wave 2
ACTIVITY_TYPE
CAMPAIGN
CURRENCY_EXCH_RATE
EMPLOYEE
EVENT
EXTERNAL_ORG
FIXTURE
HOUSEHOLD
ITEM_CATEGORY
PRICE_INDEX
QUESTION_MASTER
SECTION
SUB_FIXTURE
SURVEY
TIME_PERIOD
TRANSPORTATION_NETWORK_MODEL
VEHICLE_TYPE

Tables Loaded in Wave 3
CAL_DATE_X_TIME_PERIOD
COMMUNICATION
EXTERNAL_ORG_ADDRESS
EXTERNAL_ORG_ASSOC
HOUSEHOLD_ENRICHMENT
INTERNAL_ORG
ITEM
ITEM_CATEGORY_ASSOC
QUESTION_OPTION
SURVEY_X_QUESTION TIME_PERIOD_ASSOC

Tables Loaded in Wave 4
INTERNAL_ORG_ASSOC
ITEM_BUNDLE
ITEM_PACK_CONFIG
LOCATION
SUBSTITUTE_ITEM
SUPPLIER

Tables Loaded in Wave 5
CENSUS_DATA
COMPETITOR_BASKET_SALES
COMPETITOR_GEO_SALES
COMPETITOR_ITEM_PRICE
CUSTOMER
FACILITY
INTERNET
ITEM_BUNDLE_X_ITEM
ITEM_CATEGORY_SALES_TARGET
ITEM_INVENTORY
ITEM_PACK_CONFIG_ASSOC
ITEM_PRICE
ITEM_SEASON
ITEM_SHRINKAGE
LOCATION_ASSOC
LOCATION_X_ITEM
ON_ORDER
PROMO
RETAIL_CATALOG
RETAIL_OUTLET
SHIPPER
SUPPLIER_DEAL
SUPPLIER_X_ITEM
WEATHER_RECORD

Tables Loaded in Wave 6
AGE_GROUP_CENSUS_DATA
CORP_CUSTOMER
CUSTOMER_CARD
CUSTOMER_CONTACTS
CUSTOMER_ENRICHMENT
CUSTOMER_EVENTS
CUSTOMER_LAST_ORDER
CUSTOMER_LAST_TRANSACTION
CUSTOMER_SERVICE
CUSTOMER_SURVEY_ANSWER
CUSTOMER_X_COMMUNICATION
CUSTOMER_X_SURVEY
EDUCATION_CENSUS_DATA
FACILITY_AREA
HOUSEHOLD_CENSUS_DATA
INDIVIDUAL_CUSTOMER
INDUSTRY_CENSUS_DATA
LOCATION_ITEM_X_TRANS_NETWORK
OCCUPATION_CENSUS_DATA
PROMO_AMOUNTS
PROMO_VEHICLE
PROMO_X_ITEM
RESPONSE

Tables Loaded in Wave 6
RETAIL_OUTLET_FOOTFALL
RETAIL_OUTLET_SECTION
RETAIL_OUTLET_TIME_DETAIL
RETAIL_OUTLET_X_CALENDAR_DATE
RETAIL_OUTLET_X_EVENT
RETAIL_OUTLET_X_FACILITY
ROUTE_SEGMENT
SUPPLIER_ITEM_DEAL
SUPPLIER_X_FACILITY
VEHICLE

Tables Loaded in Wave 7
CUSTOMER_CARD_BALANCES
CUSTOMER_ORDER
DELIVERY_HEADER
FACILITY_AREA_PROCESS
INDIVIDUAL_CUSTOMER_ADDRESS
IND_CUSTOMER_X_HOUSEHOLD
PROMO_VEHICLE_X_ITEM
PURCHASE_ORDER
RETAIL_OUTLET_FIXTURE
TRANSPORTATION_NETWORK_X_ROUTE
TRANSPORT_RATE

Tables Loaded in Wave 8
CUSTOMER_ORDER_LINE_ITEM
DELIVERY_LINE_ITEM
INVOICE
MAJOR_ACTIVITY_COSTS
PO_LINE_ITEM
PO_X_DELIVERY
RETAIL_OUTLET_SUB_FIXTURE
RETAIL_TRANSACTION

Tables Loaded in Wave 9
DELIVERY_X_INVOICE
INVOICE_LINE_ITEM
MINOR_ACTIVITY_COSTS
PO_LINE_ITEM_X_DEL_LINE_ITEM
RETAIL_TRANS_LINE_ITEM
RETAIL_TRANS_PAYMENT
RO_SUB_FIXTURE_X_ITEM

Tables Loaded in Wave 10
DEL_LINE_ITEM_X_INV_LINE_ITEM
RTL_TR_LN_ITM_X_CST_ORD_LN_ITM

APPENDIX 2

Table Types in the SAS Detail Data Store for Retail

Transactional Tables ... *43*
Master Tables ... *43*
Reference Tables ... *44*
Intersection Tables ... *44*
Association Tables ... *44*

Transactional Tables

Transactional tables capture events that occur at a particular point in time, or have a beginning and ending time. Examples of transactional tables include:

- RETAIL_TRANSACTION
- ITEM_INVENTORY
- RETAIL_OUTLET_FOOTFALL

Master Tables

Master tables are reference tables that contain a significant number of rows (for example, over 100 rows) and are frequently updated. Master tables differ from reference tables, which contain a limited number of rows and are relatively static. Master tables typically include the following columns:

- an _RK column for a retained key that remains the same for different versions of the same instance of an entity
- a VALID_FROM_DTTM column and a VALID_TO_DTTM column that distinguish different versions of the same instance of an entity
- an _ID column that identifies the source ID
- a SOURCE_SYSTEM_CD column for the source system code that identifies where the _ID column comes from

Examples of master tables include the following:

- ITEM
- CUSTOMER
- SUPPLIER

Reference Tables

Reference tables contain code value definitions. Reference tables are nonvolatile and have a restricted set of values. Reference tables include a _CD column for the code and a _DESC column for the code description. Reference tables include a VALID_FROM_DTTM column and a VALID_TO_DTTM column to track changes. Examples of reference tables include the following:

- SEASON_CD
- VALID_FROM_DTTM
- LANGUAGE_CD
- VALID_TO_DTTM
- SEASON_DESC
- PROCESSED_DTTM

Intersection Tables

Intersection tables resolve many-to-many relationships between tables. For example, an employee might be associated with more than one internal organization, and an internal organization contains multiple employees. Examples of intersection tables include the following:

- LOCATION_X_ITEM
- CUSTOMER_X_ADDRESS

Association Tables

Association tables establish hierarchy relationships or other relationships between rows in a table. Association tables have a _ASSOC suffix.

Association tables include a column that refers to the base rows and a column that refers to the parent row. For example, the INTERNAL_ORG table includes the INTERNAL_ORG_RK column and the PARENT_INTERNAL_ORG_RK column. The INTERNAL_ORG_ASSOC_TYPE_CD column indicates the kind of hierarchy that this row represents. For example, the TYPE code could indicate that the hierarchy is an employee-reporting relationship, such as departments rolling up into divisions. Or, the TYPE code could indicate departments rolling up into different divisions for financial reporting purposes.

All association tables have an ASSOC_TYPE table to define the relationship between parent and child rows in the table. Examples of association tables include the following:

- INTERNAL_ORG_ASSOC
- ITEM_CATEGORY_ASSOC

APPENDIX 3

Standard Data Types and Naming Conventions

The retail DDS models use a consistent naming convention that includes an abbreviation formula and the use of class codes on the columns. The class codes indicate the type of data that is contained in that field (such as code, indicator flag, and so on). The retail DDS models use a set of domains to define consistency in data types and lengths.

Table A3.1 Standard Data Types

Domain	Data Type	Length	Applicable Class Codes	Comment/Example
Identifier	varchar	32	ID	identifier from the source system
Small Code	varchar	3	CD	short length codes, such as ADDRESS_TYPE_CD
Medium Code	varchar	10	CD	medium length codes, such as EXCHANGE_SYMBOL_CD
Large Code	varchar	20	CD	long length codes, such as POSTAL_CD
Standard Count Code	numeric	6	CNT	standard counts, such as AUTHORIZED_USERS_CNT
Name	varchar	40	NM	proper name, for example, LAST_NM, FIRST_NM
Short Length Text	varchar	20	TXT	short free-form text
Medium Length Text	varchar	100	TXT, DESC	longer free-form text and descriptions that are associated with code tables
Indicator Field	character	1	FLG	binary indicatory flag (Y or N)
Surrogate Key	numeric	10	RK, SK	generated surrogate keys
Currency Amount	numeric	18,5	AMT	standard currency amount
Rates and Percentages	numeric	9,4	PCT, RT	exchange rates
DateTime	date		DT, DTTM	accommodate dates and date/time

APPENDIX 4

Data Model Notation

Reading the Relationship Notation 47
Things to Keep in Mind 47
Understanding a One-to-One Relationship 47
Understanding a One-to-Many Relationship 48

Reading the Relationship Notation

Two tables can have different types of relationships between them. The notation between the tables in the displayed or printed data model describes how they are related. This appendix describes how to read this notation.

Note: The entire retail DDS model is available in *SAS Detail Data Store for Retail: Data Dictionary Reference*.

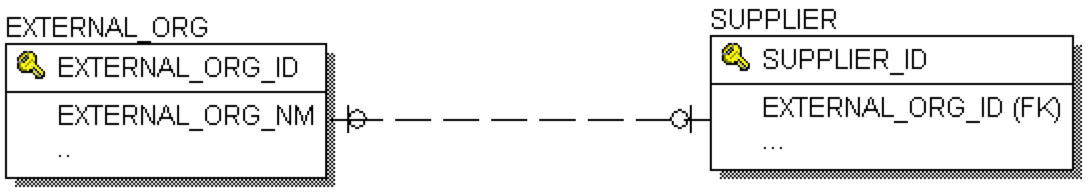
Things to Keep in Mind

- The retail DDS data model uses Information Engineering notation as it is represented in Computer Associates AllFusion ERwin Data Modeler.
- The ellipses in the columns indicate that additional columns exist, but they are not necessary to explain the notation.
- The key sign indicates that the column(s) represent the primary key of the table. These column(s) are needed to uniquely identify rows in that table. For example, in Figure A4.2, EMPLOYEE_ID is the primary key of the column.
- The letters (FK) indicate that the column serves as the foreign key. A foreign key is a column, or combination of columns, that refers to a primary key in another table. However, in the case of a recursive relationship, a foreign key could refer to a primary key in the same table. In Figure A4.3, DEPARTMENT_ID in the EMPLOYEE table serves as a foreign key to the DEPARTMENT table.

Understanding a One-to-One Relationship

In a one-to-one relationship between two tables, the relationship is a non-identifying relationship, which is optional in both directions. The same notation applies when you specify an optional relationship and when you identify relationships in a one-to-one relationship.

Figure A4.1 One-to-One Relationship

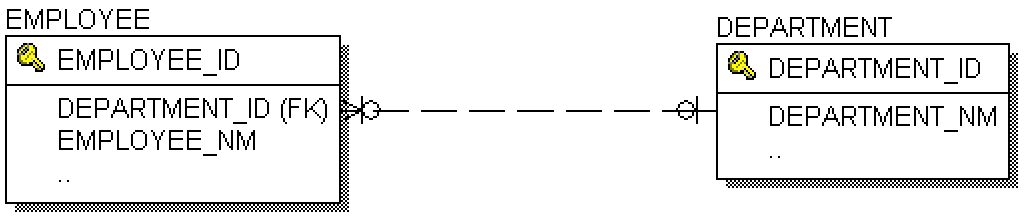


Understanding a One-to-Many Relationship

Figures A4.2, A4.3, A4.4, and A4.5 illustrate a one-to-many relationship between a department and an employee. Each department can potentially contain multiple employees; however, subtle differences exist in these relationships.

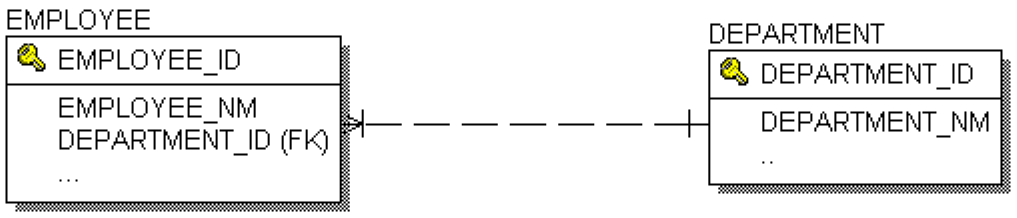
In Figure A4.2, an optional relationship exists between a department and an employee in both directions. An EMPLOYEE table does not have to have a DEPARTMENT_ID and a DEPARTMENT table does not have to have an EMPLOYEE_ID. This optional relationship is represented by the circles on both sides.

Figure A4.2 Optional Relationship between EMPLOYEE and DEPARTMENT Tables



In Figure A4.3, a DEPARTMENT table must contain one or more EMPLOYEE tables. This requirement is indicated by the hash mark on the EMPLOYEE side of the relationship.

Figure A4.3 DEPARTMENT Table Must Contain an EMPLOYEE Table



In Figure A4.4, the relationship is the same as in Figure A4.2, except that an EMPLOYEE table must contain a DEPARTMENT_ID. This requirement is indicated by the hash mark on the DEPARTMENT side of the relationship.

Figure A4.4 EMPLOYEE Table Must Have a DEAPRTMENT_ID

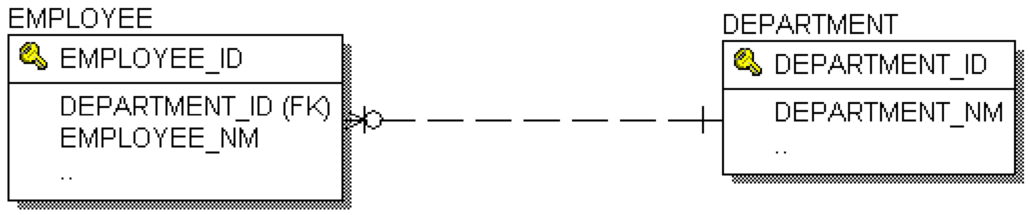
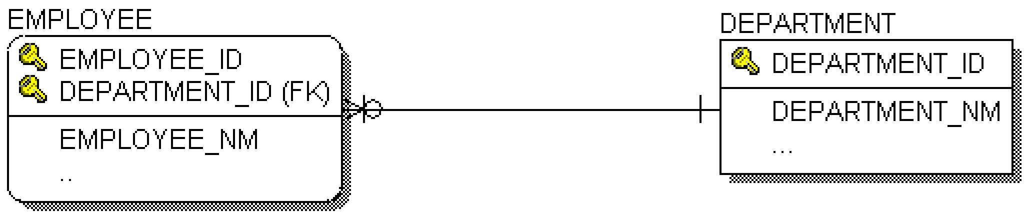


Figure A4.5 illustrates an identifying relationship between tables, which is indicated by the solid line between the DEPARTMENT and EMPLOYEE tables. This identifying relationship means that an EMPLOYEE table cannot exist outside the context of a DEPARTMENT table. In identifying relationships, the primary key of the parent table becomes part of the primary key of the child table.

Figure A4.5 Identifying Relationship between Tables



Glossary

AllFusion ERwin Data Modeler
a visual data modeling tool that is used to create and maintain databases, data warehouses, and enterprise data models. The AllFusion ERwin Data Modeler helps to visually determine the proper data structure, key elements, and optimal design for the database.

association table
the table that establishes hierarchical relationships or other relationships between rows in a table. An association table often includes the _ASSOC suffix.

attribute
in entity-relationship modeling, an attribute is a property or a characteristic of an entity that can be stored as a data fact. In subject modeling, an attribute represents any characteristic that the modeler might choose to capture about a subject.

business key
the primary source system identifier that uniquely identifies the data to be loaded into a detail data store.

data mart
a subset of the data in a data warehouse. A data mart is designed to meet the needs of a set of users. A data mart can store the results of ad hoc queries or cross-subject analyses. A data mart is more limited in scope than a data warehouse, which contains information that is used by more than one set of users.

data source
information that a user will extract, transform, and summarize in a detail data store. A data source can be in any format that SAS can read and on any supported hardware platform.

DDL (document description language)
a language that allows the structure and instances of a database to be defined in a human- and machine-readable form.

DDLDDS
a macro that includes the SAS files that are necessary to create tables that are used to instantiate the model.

denormalized
a data object structure that has intentional data redundancy, most often done to improve query performance.

entity
a topic about which the business stores or plans to store data. An entity is a person, place, thing, concept, or event that represents a subject of business information.

ETL (extract, transform, and load)
the process of (1) obtaining data from operational sources (the extract step), (2) cleansing and preparing data for import into the data warehouse (the transform step), and (3) importing the transformed data into the data warehouse (the load step).

ETL job
a set of instructions that is used to specify ETL processes that are needed to create output.

foreign key
the primary key of one data structure that is placed into a related data structure to represent a relationship between those data structures. A foreign key resolves relationships and supports navigation among data structures.

information engineering (IE)
a methodology that is used in entity-relationship modeling that uses its own notation for model diagrams.

job
a metadata record that specifies the processes that create one or more data stores (output tables). A job connects a series of process steps into a single unit. A job can include scheduling metadata that enables the process flow or user-supplied program to be executed in batch mode at a specified date and time.

master table
a reference table that usually contains a significant number of rows and is frequently updated.

metadata repository
a collection of related metadata objects, such as the metadata for a set of tables and columns that are maintained by an application. A SAS Metadata Repository is an example.

natural key
the primary source system identifier, which is often used to go to the transactional system to investigate the source data.

physical model
represents the detailed specification of what is physically implemented.

primary key
a set of one or more attributes that uniquely identifies a single occurrence in a data structure, such as a row in a table. In a logical model, every entity (or meter) has a primary key.

reference table
contains the code value definitions. A reference table is non-volatile (rarely updated) and has a restricted set of values.

relationship
an important, real-world association among entities that has meaning as business information. In a subject model, relationships illustrate the most visible business associations among subjects.

Retail solution
a SAS solution that addresses retail-specific issues, such as Customer Insight, Transaction Insight, Campaign Management, Supply Chain Costing, and so on. A Retail solution can offer an enterprise-wide business scorecard with retail-specific key performance indicators (KPIs) that gives a single, strategic view of an enterprise to help drive the retail strategies forward.

retained key
a numeric column in a dimension table that is combined with a begin-date column to make up the primary key.

source
an input to an operation. Source is often referred to as source data.

subject
a high-level view of a topic of business interest that is equivalent to both a global data class and a class of data entities.

surrogate key
a single-part, artificially established identifier for an entity. A surrogate key is a special case of derived data - one where the primary key is derived. A common way of deriving surrogate key values is to assign integer values sequentially.

target
an output location of an operation.

transactional table
a table that captures events or occurrences that occur at a particular date and time or have a beginning time and an ending time.

transformation
a metadata object that specifies how to extract data, transform data, or load data into data stores. Each transformation that is specified in a process flow diagram generates or retrieves SAS code. User-written code can be specified in the metadata for any transformation in the process flow diagram.

Your Turn

If you have comments or suggestions about *SAS Detail Data Store for Retail 2.2: Implementation and Administration Guide*, please send them to us on a photocopy of this page or send us electronic mail.

For comments about this book, please return the photocopy to

SAS Publishing
SAS Campus Drive
Cary, NC 27513
E-mail: **yourturn@sas.com**

For suggestions about the software, please return the photocopy to

SAS Institute Inc.
Technical Support Division
SAS Campus Drive
Cary, NC 27513
E-mail: **suggest@sas.com**