SAS® Financial Management 5.5

Data Administrator's Guide

SAS® Documentation

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2015. *SAS® Financial Management 5.5: Data Administrator's Guide*. Cary, NC: SAS Institute Inc.

SAS® Financial Management 5.5: Data Administrator's Guide

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513-2414.

January 2015

SAS provides a complete selection of books and electronic products to help customers use SAS® software to its fullest potential. For more information about our offerings, visit **support.sas.com/bookstore** or call 1-800-727-3228.

Contents

Using This Book

Audience

This book is intended for SAS Financial Management data administrators. To administer SAS Financial Management, you must be familiar with the operating system on which it is installed. For example, you must know how to create folders, run scripts, and update environment variables. If you are using Microsoft Windows, you must also be an administrator of the machine.

Documentation Conventions

Directory Paths

Directory Paths Used by Previous Installations

This book uses the following documentation conventions to identify directory paths used by SAS Financial Management 5.3 and earlier installations:

Term	Refers to	Example Path

!sasroot	Path to the SAS root directory in a SAS 9.2 installation	Windows:
		`C:\Program Files\SAS\SASFoundation\9.2`
		UNIX:
		`/usr/local/SAS/SASFoundation/9.2`

!sasroot	Path to the SAS root directory in a SAS 9.3 installation	Windows:
		`C:\Program Files\SASHome\SASFoundation\9.3`
		UNIX:
		`/usr/local/SASHome/SASFoundation/9.3`

SAS-config-dir	Path to the SAS configuration directory	Windows:
		`C:\SAS\Config\Lev1`
		UNIX:
		`/usr/local/SAS/config/Lev1`

Term	Refers to	Example Path

MySQL-install-dir	Path to the MySQL installation directory in a SAS installation prior to SAS 9.4	Windows:
		`C:\MySQL\bin`
		UNIX:
		`/usr/local/mysql`
		Note: As of SAS Financial Management 5.4, MySQL is no longer supported. For more information, see *SAS Financial Management: System Administration Guide.*

Directory Paths Used by a SAS Financial Management 5.4 and SAS Financial Management 5.5 Installation

This book uses the following documentation conventions to identify directory paths that are used by SAS Financial Management 5.5:

Term	Refers to	Example Path
!sasroot	Path to the SAS root directory (SAS Foundation)	Windows: `C:\Program Files\SASHome\SASFoundation\9.4` UNIX: ha`/usr/local/install/SASHome/SASFoundation/9.4`

!sasinst	Path to the SAS installation directory	Windows:
		`C:\Program Files\SASHome`
		UNIX:
		`/usr/local/install/SASHome`

SAS-config-dir	Path to the SAS configuration directory	Windows:
		`C:\SAS\Config\Lev1`
		UNIX:
		`/usr/local/SAS/config/Lev1`

Terms

This book uses the following terms:

Term	Description
Data Mart	The SAS Financial Management Data Mart
data tier	The machine on which you install the data-tier software for SAS Financial Management.

Term	Description
middle tier	The machine on which you installed the web application server and on which your web applications run.
metadata tier	The machine on which you installed the SAS Metadata Server. Usually, this is the same machine as the data tier.
multi-tier installation	An installation that is done on more than one machine (for example, with a data tier and a middle tier).
single-tier installation	An installation that is done on one machine. In that case, the single machine functions as both the data tier and the middle tier. Follow instructions for both the data tier and the middle tier.
staging area	The SAS Financial Management staging area.

Note:

- The name of the configuration directory and the SAS release might be different at your site.
- If your configuration is the result of a migration from the previous release of SAS Financial Management, the SASApp directory might be called SASMain instead. For example: `C:\SAS\Config\Lev1\SASMain` instead of `C:\SAS\Config\Lev1\SASApp`. Please make the appropriate substitutions as you read this book.
- File system pathnames are typically shown with Windows separators (\); for UNIX, substitute a forward slash (/).
- Some code examples contain line breaks so that the code fits on the line. If you copy the code, remove the line breaks.

What's New

What's New in SAS Financial Management 5.5

Overview

The following sections describe the new features and enhancements introduced in SAS Financial Management 5.5.

Performance Improvements

New features and enhancements that improve the performance of the SAS Financial Management solution include the following:

- High Availability—Leverages the standby and failover support provided by the SAS Web Server to implement a high availability configuration. Standby and failover is a traffic balancing configuration in which two instances of a SAS Web Server are redundantly configured in a primary server and standby server relationship. The standby server accepts traffic only when the connection to the primary server fails. When the connection to the primary server is restored, traffic is automatically rerouted back to the primary server.
- Reduced impact of the refreshing the fact cache during data load.
- Improved performance when opening SAS Financial Management reports and forms.
- Improved performance when expanding and collapsing SAS Financial Management read-only reports and data-entry forms.
- Improved auto-allocation performance.

Improved Usability

New features and enhancements that improve the usability of the SAS Financial Management solution include the following:

- Viewing Reports—The functionality of the Reports Workspace in the SAS Financial Management Web Application has been replaced by the SAS Visual Analytics Viewer.
- Extended Partial Promotion—Content promotion has been expanded to support the promotion of the following SAS Financial Management objects:
  - Exchange Rate Sets and Driver Rate Sets
  - Models
  - Manual Adjustments
  - Cycles
- Enhanced SAS Financial Management Forecasting—When generating a forecast, you can now also generate a report that provides details about how the forecast was derived.
- Expanded Auto-Allocation Functionality—Auto-Allocation capabilities have been added to the forms viewed in the SAS Financial Management Web Application. The reallocation of values for a selection of cells, or at a table level, is supported by both the SAS Financial Management Web Application and the SAS Financial Management Add-In for Microsoft Excel. In addition, batch auto-allocation is supported.

Accessibility

Accessibility Information

For information about the accessibility of SAS Financial Manager, see the *SAS Financial Management: User's Guide*.

For information about the accessibility of other products mentioned in this document, see the documentation for that product.

1

Introduction

Overview

This chapter describes the following topics:

- SAS Financial Management
- SAS Financial Management data administrations
- summary of data administrator tasks
- SAS Financial Management server configuration
- related documentation

About SAS Financial Management

SAS Financial Management Features

SAS Financial Management is an advanced SAS solution for planning and reporting. It is designed to support the following financial management activities:

- data collection and retrieval

- currency translation
- management of dynamic hierarchical structures
- intercompany eliminations
- allocations and balancing entries
- ownership eliminations
- reporting

SAS Financial Management User Interfaces

There are three ways to interface with SAS Financial Management:

Table 1.1 *SAS Financial Management Interfaces*

Interface	Description
SAS Financial Management Studio	Desktop application from which data administrators and other users build and manage the infrastructure that is required by the SAS Financial Management Add-In for Microsoft Excel application and the SAS Financial Management web application. For example, an administrator can use SAS Financial Management Studio to define the metadata that feeds into meaningful reports and forms.
SAS Financial Management Add-In for Microsoft Excel	SAS application that connects a desktop copy of Microsoft Excel to the SAS Financial Management database. With this client application, certain users build and save financial reports and forms. Other users (with permission) can use this application to view report data and enter data into forms for planning purposes. Data administrators also use the SAS Financial Management Add-In for Microsoft Excel client to create form templates for planning.
SAS Financial Management Web Application	Web-based application that alerts the information consumer when a report has been published or alerts the information provider or information reviewer when an action is required. The SAS Financial Management web application includes a workspace for managing and editing data-entry forms for financial planning. The flow of data is controlled by a workflow that an administrator defines in SAS Financial Management Studio. The SAS Financial Management web application also includes workspaces for accessing business processes and (for administrators) viewing security information. From the web application, users can access financial reports by using the SAS Visual Analytics Viewer.

Note: Every SAS Financial Management deployment also includes SAS Data Integration Studio. Data administrators use SAS Data Integration Studio to load data and metadata for SAS Financial Management.

Elements of SAS Financial Management Data

Cycles and Dimension Types

In SAS Financial Management, a *cycle* is a structured pool of stored data. There are seven required dimension types in a cycle:

- Account
- Analysis
- Currency
- Frequency
- Organization
- Time
- Source
- Trader (a mirror of Organization)

Note: If necessary, administrators can define additional dimension types to meet the requirements of their site.

Note: The Source dimension and Frequency dimension are defined at implementation.

Dimension Members and Crossings

Each numeric value belongs to a *crossing*. A crossing consists of the set of all dimension members that are associated with that value. There is one member from each relevant dimension type.

Each data record in the SAS Financial Management database consists of one crossing and one associated numeric value. The dimension members that are associated with a SAS Financial Management data record indicate what the numeric value in that record represents. For example, the dimension members of one record might indicate that the numeric value represents the actual revenue for an Italian subsidiary in October 2011, expressed in euros. The dimension members of another record might indicate that the numeric value represents the planned salary expense for a Japanese subsidiary for fiscal year 2012, expressed in yen.

Hierarchies

The dimension members that are used by SAS Financial Management belong to *hierarchies*. The hierarchical relationships between members help define the structure of the dimension.

For certain dimension types (Currency, Frequency, and Analysis), there is no hierarchical relationship between the dimension members. These are known as *flat hierarchies*.

Models

A *model* is a structure for viewing and interacting with the data in a cycle. The model is the basis of both forms and reports. The structure of a model includes a

set of hierarchies (from required and optional dimension types) as well as rates and formulas. A model can also have its own set of adjustments and rules.

SAS Financial Management Server Configuration

When using SAS Financial Management, you might work with the following types of servers.

- Metadata server—Server on which the SAS Metadata Server software is running. SAS must be available on this same machine.
- Data Tier server—Server on which SAS runs data-handling programs (including the logical servers for SAS Workspace and SAS Stored Process servers). Transformations, error tables, and jobs are installed on the data tier server.

  Note: The same machine is often used as both the data tier server and the metadata server.
- Middle-Tier server—Server on which the SAS managed servers run.

  Note: SAS 9.4 does not use SAS Remote Services. However, SAS Remote Services is included in SAS 9.4 installations for backward capability.

SAS Financial Management Data Administration

As SAS Financial Management data administrator, your primary task is to supply data to SAS Financial Management. The data that you supply spans a variety of *content categories*. The roles a content category plays, and the time at which a content category is needed varies.

For example:

- To enable SAS Financial Management to work, you must initially supply data that belongs to certain content categories.
- To enable SAS Financial Management to produce timely output, you must periodically supply data that belongs to other content categories.
- The data that belongs to some content categories, you might not need at all.

Supplying data to each content category involves considerations unique to your deployment. However, the following are some standard concepts about the process of supplying data:

- The final destination of the data that you supply is a Data Mart (table or tables) to which SAS Financial Management has access.
- The primary method of moving data from one table to another is by running a SAS Data Integration Studio job.
- For many content categories, the data travels from its source to the SAS Financial Management Data Mart through a set of intermediate SAS tables

called *staging tables*. For these content categories, you move the data by completing the following two tasks:

1. Running a custom, site-specific job that extracts the data from its source, and loads it into a staging table that is designed to hold the data.
2. Running a SAS Data Integration Studio job or an equivalent SAS Financial Management Studio wizard to move the data from the staging table to its destination in the Data Mart.

Summary of Data Administrator Tasks

As a SAS Financial Management data administrator, the tasks that you must complete depend on your site-specific circumstances. However, at any site, you must complete the following two tasks in the specified order before you can begin to load data:

1. Install SAS Financial Management.

   For information about installing SAS Financial Management, see *SAS Financial Management: Installation and Configuration Guide* (see"Related Documentation" on page 6).

2. Prepare the SAS Data Integration Studio environment.

   For information about preparing the SAS Data Integration Studio environment, see Chapter 2, "Setting Up the SAS Data Integration Studio Environment," on page 9.

In addition to installing SAS Financial Management and preparing the SAS Data Integration Studio environment, review and complete the following tasks as required by your implementation:

- Load user and user group data into the Data Mart. Ensure that the user and group data in the Data Mart matches the user and group data in the metadata repository.

  For more information about loading users and user groups, see Chapter 5, "Loading Users and User Groups," on page 23.

- SAS Financial Management software includes a set of predefined dimension types. If the set of predefined dimension types does not meet your needs, define additional dimension types to describe your financial accounting data.

  Note: When creating dimension types, you must use at minimum, the following dimension types to describe financial accounting data:

  - ACCOUNT
  - ANALYSIS
  - CURRENCY
  - INTORG
  - TIME

  For more information about creating dimension types, see Chapter 10, "Adding a Dimension Type," on page 65.

- For each dimension type that you use to describe data, ensure that it is properly stocked with dimensions, members, and hierarchies.

  For more information about loading members and hierarchies into a dimension, see Chapter 7, “Loading Members and Hierarchies into a Dimension,” on page 33.

- To control Write access to planning forms, load **Users** tab user-member associations.

  For more information about loading **User** tab user-member associations, see “Users Tab Data” on page 46.

- To control Read access to reports in the SAS Financial Management Add-In for Microsoft Excel, load **Security** tab user-member and group-member associations.

  For more information about loading **Security** tab user-member associations, see “Security Tab Data” on page 46.

- On a periodic basis, load fresh financial accounting data.

  For information about loading accounting data into a SAS Financial Management cycle, see Chapter 17, “Loading Base Data into a Financial Cycle,” on page 127.

- On a periodic basis, load fresh currency exchange rates.

  For information about loading currency exchange rates, see Chapter 12, “Loading Exchange Rates into a SAS Financial Management Exchange Rate Set,” on page 79.

- Load cell visibility rules that users of the SAS Financial Management Add-In for Microsoft Excel can apply to read-only tables and data-entry tables.

  For more information about loading cell visibility rules, see Chapter 15, “Loading Cell Visibility Rules for a Model,” on page 111.

- If you are managing two related SAS Financial Management systems (for example, a development system and a production system), you can promote dimension members and hierarchies from one system to the other.

  For information about promoting dimension members and hierarchies, see Chapter 9, “Exporting and Promoting Members and Hierarchies,” on page 57.

- At your discretion, you can widen the availability of any SAS Data Integration Studio job by converting it into a stored process.

  For information about converting a SAS Data Integration Studio job to a stored process, see Chapter 11, “Creating a Stored Process,” on page 77.

Related Documentation

SAS Financial Management

For information about installing, administrating, or migrating SAS Financial Management, see the documentation located at

http://support.sas.com/documentation/onlinedoc/fm/

Note: This site is password-restricted. You can find the user name and password in the pre-installation checklist, the Instructions.html, or by contacting SAS Technical Support at http://support.sas.com/techsup/contact/

SAS Intelligence Platform

For information about administering the SAS Intelligence Platform, see the documentation located at

http://support.sas.com/documentation/onlinedoc/intellplatform/index.html

SAS Information Delivery Portal

For information about the SAS Information Delivery Portal, see the documentation located at

http://support.sas.com/documentation/onlinedoc/portal/index.html

SAS Notes

SAS Technical Support develops SAS Notes to inform customers of issues that they need to be aware of when using SAS software. SAS Notes contain additional information about a SAS product and support fixes.

To view SAS Notes for SAS Financial Management, see the product page at

http://support.sas.com/software/products/fm/index.html

On the left side of the product page, select **Samples & SAS Notes** from the menu.

2

Setting Up the SAS Data Integration Studio Environment

Overview

This chapter describes the following topics:

- about the SAS Data Integration Studio environment
- securing data directories
- configuring server access for SAS Data Integration Studio users
- assigning groups and roles for data administrators

.

About the SAS Data Integration Studio Environment

As a SAS Financial Management data administrator, you use SAS Data Integration Studio to load data and metadata for SAS Financial Management.

After you install SAS Financial Management, you must set up the SAS Data Integration Studio environment before you can load data. Setting up the environment includes configuring access settings for the data tier server, securing data directories, and assigning groups and roles for the data administrators.

Access Settings for the Data Tier Server

Securing Data Directories

For information about protecting data files and folders, see “Post-Configuration Steps” in the *SAS Financial Management: System Administration Guide*.

Configuring Server Access for SAS Data Integration Studio Users

Each SAS Data Integration Studio user must have a user ID and password for the Data Tier server.

When configuring server access to a SAS Data Integration Studio user, note the following:

- A data administrator user cannot be the unrestricted user. If you log on as the unrestricted user, you cannot attach the libraries that are necessary to run SAS Data Integration Studio jobs that supply data to SAS Financial Management.
- The data administrator must also have the following rights and permissions:
  - in a Windows environment, the **`Log on as a batch job`** right.

    The recommended way to grant this right to a user is to place the user in the SAS Server Users group and grant the right to this group. For more information, see “Windows Privileges” in the *SAS Intelligence Platform: Security Administration Guide*.
  - read/write/update access to the directories that hold data

    This includes Read, Write, and Update access to the ***`SAS-config-dir`*****`\Lev1\Data`** directory and all of its subdirectories.

Assigning Groups and Roles for Data Administrators

For information about group and role requirements for data administrators, see “Assigning Groups and Roles” in the *SAS Financial Management: System Administration Guide*.

3

Supplying Data to SAS Financial Management

Overview

This chapter describes the following topics:

- about loading data
- loading data into staging tables
- loading data into the Data Mart
- extending the staging area

About Loading Data

As a SAS Financial Management data administrator, you load most data from its source, through the SAS Financial Management staging area in SAS Data Integration Studio, to the SAS Financial Management Data Mart. The Data Mart consists of PostgreSQL tables specific to SAS Financial Management that support dimensional data, fact data, and special data.

The staging area has a SAS library named StageFM. The StageFM library consists of approximately 85 SAS Financial Management tables that are categorized as follows: dimension tables, fact tables, reference tables, and special tables. Typically, you load data from the StageFM library into the SAS Financial Management Data Mart.

In general, sources of data are transactional systems or databases that are located outside the SAS environment. However, there are some SAS tables of predefined data that install with the SAS Financial Management software.

As a SAS Financial Management data administrator, you load data into the Data Mart by completing the following tasks:

1 Loading data into staging tables.

- If the data that you are loading is from a source outside of the SAS environment, you must write a job that extracts the data from its source and loads the data into the appropriate SAS Financial Management staging table in SAS Data Integration Studio.
- If the data that you are loading is from a SAS table of predefined data, you can use a SAS Data Integration Studio job to load the corresponding staging area table.

2 Loading the data from the staging tables into the SAS Financial Management Data Mart.

You can perform this step from SAS Data Integration by running the appropriate job, or for some data categories, you can also perform this step from SAS Financial Management Studio.

For example, to load base accounting data into the Data Mart, you can do either of the following:

- In SAS Data Integration Studio, run the fm_1100_load_base_data job or the fm_1100_load_base_data_unlock_periods job.
- In the Periods workspace of SAS Financial Management Studio, run the Load New Data wizard.

Loading Data into Staging Tables

The method that you use to load data from its source to a staging table in SAS Data Integration Studio depends on whether the source of the data is outside the SAS environment or the source is a SAS table of predefined data.

Loading Data from an Outside Source to a Staging Table

If a data source is outside the SAS environment, you can load the appropriate staging table from the data source by writing a job. When loading data from an outside source into a staging table in SAS Data Integration Studio, note the following:

- you can write a separate job to load each staging table
- you can write jobs that load groups of related staging tables
- you can run the jobs in any order
- you can store the jobs in any folder

When loading data from an outside source, ensure that the job that you write places the correct data in the correct columns of the correct staging tables. Before you write a job to extract data from an external source and load it into a

staging table, ensure that you understand the data columns in the relevant staging table. For each column, you must also determine the data source or verify that it is appropriate to leave the column empty.

For example, to write a job to load the GL_TRANSACTION_SUM table, you must understand all of the data columns in the GL_TRANSACTION_SUM table, as explained in Chapter 17, “Loading Base Data into a Financial Cycle,” on page 127.

Note: This administration guide discusses the structure of some of the staging tables. For detailed information about the structure of every staging area table, see *SAS Financial Management: Data Model Reference*.

When writing a job to extract data from an external source and load it into a staging table, you might be able to use the following:

- User-Written Code transformation.
- Register tables.

  In SAS Data Integration Studio, right-click a metadata folder in the **Folders** tree and select **Register Tables** to register your data sources in the metadata repository. When you register tables, they are displayed as icons in SAS Data Integration Studio. You can then use the icons in the Process Designer.

In addition, when loading data to a staging table from an outside source, note the following:

- If one of your data sources is SAP, then you can use the SAS Financial Management Adapter for SAP to load data from its source to the staging tables. For more information about loading data from an SAP data source, see *SAS Financial Management Adapter for SAP: User's Guide*.
- If you are running SAS under 64-bit Windows, and the source files are on a machine running 32-bit Windows, you must use SAS PC Files Server to configure the data source. For instructions on using the SAS PC Files Server to configure data sources, see the *SAS Financial Management: System Administration Guide*.

Loading Data from a SAS Table of Predefined Data to a Staging Table

If the source of the data that you are loading is a SAS table of predefined data, you can use a job that is supplied with SAS Data Integration Studio. The job loads the data into the correct data columns in the relevant staging table. Therefore, you do not have to understand all the data columns of the staging table like you do when you write a job to load data from an external source.

To access the SAS tables that contain predefined data, complete the following steps:

1. In SAS Data Integration Studio, select the **Folders** tab.
2. In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **SAS Supplied FM**.

   Note: The tables with predefined data that install with the SAS Financial Management software and the staging area tables are all SAS tables

To load the data in one of the SAS tables into the corresponding staging table, you can use a job that is supplied with SAS Data Integration Studio. If necessary, you can supplement the predefined data with additional data from another source without writing an additional job by using the SAS Data Integration Studio Append transformation. For information about using the Append transformation, see the *SAS Data Integration Studio User's Guide*.

To display a list of the jobs supplied with SAS Data Integration Studio, complete the following steps:

1. In SAS Data Integration Studio, select the **Folders** tab.
2. In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.
3. To run a job, double-click the job name and in the Job Editor window, click **Run**.

For example, to load the SAS_DIMENSION_TYPE table, you use the SAS Data Integration Studio solvnc_0200_load_stagefm_dimension_type_table job.

Loading Data into the Data Mart

For some categories of data, you can bypass the staging area and load the data directly into the Data Mart. For other categories of data, you must load the data into the Data Mart through the staging area.

You can bypass the staging area and load the following data categories directly into the Data Mart:

- User and user group data.

  User and user group data travels through the metadata repository. It is loaded first into the metadata repository and then into the Data Mart from the metadata repository

  For information about loading user and user group data, see Chapter 5, "Loading Users and User Groups," on page 23.

- Dimensions

  Dimensions can be created in the staging area or directly in the Data Mart

  For information about creating dimensions, see Chapter 6, "Creating a Dimension," on page 25.

You must load the following data categories into the Data Mart through the staging area. However, you can load them from the staging area into the Data Mart in two or more ways:

- Driver rates.

  For information about loading driver rates, see Chapter 13, "Loading Driver Rates into a SAS Financial Management Driver Rate Set," on page 91.

- Members and hierarchies for an existing dimension.

  For information about loading members and hierarchies, see Chapter 7, "Loading Members and Hierarchies into a Dimension," on page 33.

- Exchange rates.

For information about loading exchange rates, see Chapter 12, “Loading Exchange Rates into a SAS Financial Management Exchange Rate Set,” on page 79.

- Base accounting data.

  For information about loading exchange rates, see Chapter 17, “Loading Base Data into a Financial Cycle,” on page 127.

Extending the Staging Area

You can extend the staging area in two ways:

- Add more tables to the SAS Financial Management staging area.
- Add columns to installed tables.

Typically, if you extend the staging area, you cannot load the additional data into a predefined Data Mart. To use the additional data, you must load it into tables in a separate location that is accessible by an appropriate application.

However, there are two exceptions to this rule:

- You can add custom dimension types whose members can be used to qualify financial accounting data for SAS Financial Management. Each custom dimension type is supported by a set of four additional staging area tables. The data in these additional tables can be loaded into the Data Mart in the same way as data for the basic dimension types.

  For detailed information about adding a dimension type, see Chapter 10, “Adding a Dimension Type,” on page 65.

- You can add a column that represents a custom property to the primary member table of any dimension type. You can load the values of a custom member property into the Data Mart.

  For detailed information about loading member properties, see Chapter 8, “Registering Member Properties,” on page 51.

If you add staging area tables to use to load non-Data Mart tables, you must create a data pathway to the non-Data Mart target tables. This data pathway is analogous to the main data pathway to the Data Mart.

To create a data pathway to the non-Data Mart target tables, complete the following steps:

1. Create the staging area tables.
2. Create the non-Data Mart target tables, if they do not already exist.

   Note: Do not write an application that accesses staging area tables.

3. In SAS Data Integration Studio, select the **Folders** tab.
4. In the **Folder** tree, right-click a folder and select **Register Tables** to register the metadata of all the new tables, the staging area tables, and the target tables.
5. Write jobs that load the staging tables.

6 Write jobs that load the non-Data Mart target tables from the staging area tables.

If you add columns to an existing staging area table to be used to load a non-data-mart target table, complete the following steps:

1 Add the columns to the staging area tables.

2 In SAS Data Integration Studio, select **Actions** ▸ **Update Metadata** to register the metadata of all the modified tables, including staging area tables.

3 Modify the jobs that load the staging tables.

4 Create the non-data-mart target tables, if they do not already exist.

Note: Do not write an application that accesses staging area tables.

5 In SAS Data Integration Studio, select the **Folders** tab.

6 In the **Folders** tree, right-click a folder and select **Register Tables** to register the metadata of the new non-Data Mart target tables..

7 Write jobs that load the non-Data Mart target tables from the staging area tables.

Note: If you add a column to a member table for the purpose of loading an additional member property into the Data Mart, complete the first six steps of this procedure. Completing these steps prepares for the trip into the staging area, and then for the trip from the staging area to the Data Mart as described in Chapter 8, “Registering Member Properties,” on page 51.

4

Loading Language Codes and Data Locale Codes

Overview

This chapter describes the following topics:

- language and data locales
- loading language and locale data into staging tables
- loading data local codes into the Data Mart

About Languages and Data Locales

Language codes and data locale codes identify the language in which associated textual data is expressed. The names and descriptions of these objects are visible to users of SAS Financial Management Studio and SAS Financial Management Add-In for Excel.

The SAS Financial Management software includes the names and descriptions of the predefined dimension types and dimensions in the following data locales:

- da (Danish)
- de (German)
- en (English)
- es (Spanish)
- fr (French)
- it (Italian)
- ja (Japanese)
- ko (Korean)

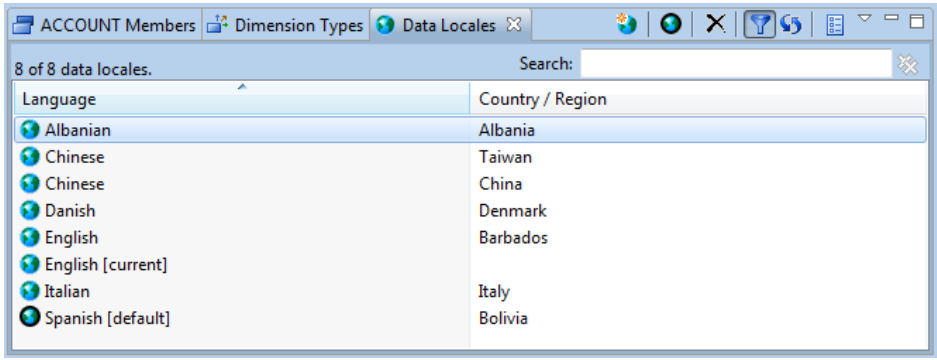
- pl (Polish)
- ru (Russian)
- zh_CN (simplified Chinese)
- zh_TW (traditional Chinese)

To view the language codes and data locale codes for a dimension, complete the following steps:

1 In SAS Financial Management Studio, select the **Dimensions** tab.

2 Select **Tools** ▸ **Data Locales**. The Data Locales information is displayed.

Display 4.1 *SAS Financial Management Studio — Data Locales Tab in the Dimensions Workspace*



Loading Language and Locale Data into the Staging Table

The staging table defined in the StageFM library for language and locale data is the CODE_LANGUAGE table.

To load the CODE_LANGUAGE table with data, you must write and run a job that loads all the language codes and data locale codes that your site requires into the table.

Before writing the job that loads language codes and data locale codes that your site requires into the CODE_LANGUAGE table, note the following:

- Each record in the table defines a language code and a three-part data locale code.
- Each record also associates the language code with the data locale code.
- Load only those languages and data locales your data uses. For example, if all of your data is in a single data locale, then you need to load only one record into this table.

Before writing a job to load data into the CODE_LANGUAGE table, review the column structure of the table to ensure that the job that you write places the data in the correct columns.

To view the column structure of the CODE_LANGUAGE table, complete the following steps:

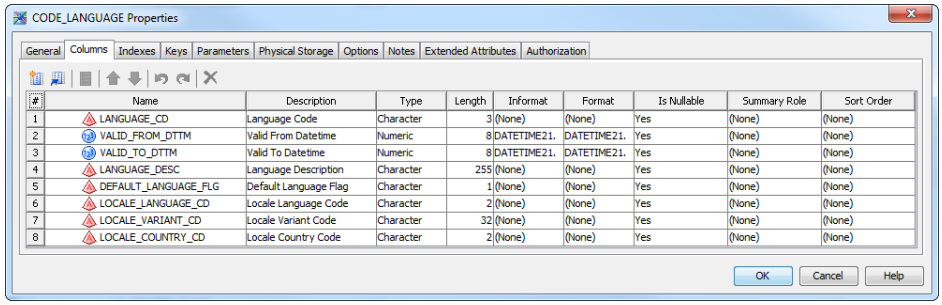
1 In the SAS Data Integration Studio, select the **Folders** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.

3 Double-click **CODE_LANGUAGE** in the list of tables. The CODE_LANGUAGE Properties window is displayed.

4 Click the **Columns** tab to view the column structure of the staging table.

Display 4.2 *CODE_LANGUAGE Properties Window — Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summary Role	Sort Order
1	LANGUAGE_CD	Language Code	Character	3	(None)	(None)	Yes	(None)	(None)
2	VALID_FROM_DTTM	Valid From Datetime	Numeric	8	DATETIME21.	DATETIME21.	Yes	(None)	(None)
3	VALID_TO_DTTM	Valid To Datetime	Numeric	8	DATETIME21.	DATETIME21.	Yes	(None)	(None)
4	LANGUAGE_DESC	Language Description	Character	255	(None)	(None)	Yes	(None)	(None)
5	DEFAULT_LANGUAGE_FLG	Default Language Flag	Character	1	(None)	(None)	Yes	(None)	(None)
6	LOCALE_LANGUAGE_CD	Locale Language Code	Character	2	(None)	(None)	Yes	(None)	(None)
7	LOCALE_VARIANT_CD	Locale Variant Code	Character	32	(None)	(None)	Yes	(None)	(None)
8	LOCALE_COUNTRY_CD	Locale Country Code	Character	2	(None)	(None)	Yes	(None)	(None)

The CODE_LANGUAGE table contains the following columns:

Column	Description
LANGUAGE_CD	Language code that is used in staging tables. Typically, the language code is one of the two-character codes in the ISO0639_LANGUAGE_CD column of the SAS_LANGUAGE_ISO0639 table. One exception is if you need two or more records that represent variants of the same language. For example, if you have a record for French as used in France and another record for French as used in Canada, then you might use language codes **`frf`** and **`frc`**, respectively. **Note:** Do not use the same language code in two records.
VALID_FROM_DTTM	Moment that begins the time period during which a row of data is valid.
VALID_TO_DTTM	Moment that ends the time period during which a row of data is valid.
LANGUAGE_DESC	Description of the language or language variant that the Language Code designates. For example, you might specify **`French`** or **`Canadian French`**.
DEFAULT_ LANGUAGE_FLG	Language code that is used in all of the primary tables. You can mark the default language flag **Y** for only one record and must specify **N** for all other records. Therefore, ensure that you coordinate the language that you mark here as the default language with the language that you use in the primary member tables. For information about primary and secondary member tables, see "Loading Member and Hierarchy Data into Staging Tables" on page 34.

Column	Description
LOCALE_ LANGUAGE_CD	Identifies the Data Mart data locale that is associated with the staging area language code. The Locale Language Code must be one of the two-character codes in the ISO0639_LANGUAGE_CD column of the SAS_LANGUAGE_ISO0639 table.
LOCALE_VARIANT_ CD	Not used.
LOCALE_COUNTRY_ CD	Identifies the Data Mart data locale that is associated with the staging area country code. The Locale Country Code must be one of the two-character codes in the ISO3166_COUNTRY_CD column of the SAS_COUNTRY_ISO3166 table. In many cases, Locale Language Code can be the same two-character code as Language Code, and the other locale columns can remain empty. Typically, the Data Mart data locale in the record that has a Default Language Flag of **Y** should be the data locale that is set in the Data Mart by the SAS Financial Management installation. **Note:** Do not use the same combination of locale language code and locale country code in two records.

5 Click **OK** to close the CODE_LANGUAGE Properties window.

To load language and locale data into the CODE_LANGUAGE table, complete the following steps:

1 Create a SAS Data Integration Studio job that loads data from its source into the CODE_LANGUAGE table in the StageFM library.

2 Click **OK** to save the job.

3 Run the job and validate that the table loaded successfully.

Note: For detailed information about creating and working with SAS Data Integration Studio jobs to extract data from a source table and load it into a target table, see the *SAS Data Integration User's Guide*.

Loading Data Locale Codes into the Data Mart

The solnsvc_1200_import_locales job loads data locale codes from the staging area into the Data Mart.

To load data locale codes from the CODE_LANGUAGE table into the Data Mart, complete the following steps:

1 In SAS Data Integration Studio, click the **Folders** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.

3 Make a copy of the solnsvc_1200_import_locales job.

Note: It is a good practice to create and maintain a separate, appropriately named job for each set of option values. Changing the value of a job's options is possible, but likely to generate confusion.

4 Double-click the **solnsvc_1200_import_locales** job in the list of jobs.

5 In the Job Editor window, click **Run**.

6 When the job displays as completed in the **Status** column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: SAS managed servers must be running on the middle-tier server to run the job. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

5

Loading Users and User Groups

Overview

This chapter describes the following topics:

- SAS Financial Management users and user groups
- loading user and user group data

About SAS Financial Management Users and User Groups

When working with SAS Financial Management user and user groups data, note the following:

- You load user and user groups data directly into the Data Mart, bypassing the staging area entirely.
- A number of default users, groups, and roles are automatically defined during the SAS Financial Management installation. For more information about these default users, groups, and roles, see the *SAS Financial Management: System Administration Guide*.
- In addition to the default users and user groups, you must add all of the users of your site. You must define the group membership and role membership for each user. You can define these users through a bulk-loading process or interactively though SAS Management Console.
- The definitions of users and groups are maintained in the metadata repository.
- When a user logs in to the SAS Financial Management software, the authentication process verifies the user data in the metadata repository. Additional uses of SAS Financial Management require that the user data be present in the Data Mart. Therefore, you must load this information from the metadata repository to the Data Mart.

Note: Whenever you make changes to the user data in the metadata repository, you must also update the user data in the Data Mart to reflect your changes.

Loading User and User Group Data into the Data Mart

You can use the following three SAS Data Integration Studio jobs to load user and user group definitions from the metadata repository into the Data Mart:

- solnsvc_1300_load_users—Loads the user definitions.
- solnsvc_1400_load_groups—Loads the user group definitions.
- solnsvc_1500_load_user_x_group—Loads the information about which users belong to which groups.

When loading user and user group data using the jobs supplied with SAS Data Integration Studio, note the following:

- A best practice is to run these three jobs on a regular schedule. For example, you can schedule a batch job to run each night.

  Note: SAS managed servers must be running on the middle-tier server to run the job. For more information about the managed servers and about scheduling batch jobs, see the *SAS Financial Management: System Administration Guide*.

- The user account from which these jobs run must have Read and Write permissions to the `SAS-config-dir\Lev1` directory on the metadata server.
- There is a stored process that includes the three jobs that load user and user group data.

  For information about running the Import Users and Groups stored process, see “Assigning Groups and Roles” in the *SAS Financial Management: System Administration Guide* (see“Related Documentation” on page 6).

To load user and user group data using the jobs listed above, complete the following steps:

1. In the SAS Data Integration Studio, select the **Folders** tab.
2. In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.
3. Double-click to select the job from the list of jobs.
4. In the Job Editor window, click **Run**.
5. When the job displays as completed in the **Status** column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

6

Creating a Dimension

Overview

This chapter describes the following topics:

- dimension types, dimensions, hierarchies, and members
- creating a dimension
- creating a dimension by using the New Dimension wizard
- creating a dimension by using a job
- creating a dimension by using a staging table
- loading new dimensions into the Data Mart

About Dimension Types, Dimensions, Hierarchies, and Members

Before you perform any task that involves dimension types, dimensions, hierarchies, or members, ensure that you understand how the four concepts are related.

- A *dimension type*:
  - represents a category of information. Examples of predefined dimension types include ACCOUNT, CURRENCY, and TIME.
  - can contain many dimensions
  - A *dimension*:

    - contains members and at least one hierarchy that is built from some or all of the members of a dimension.
    - a dimension type is like a folder that enable you to separate the hierarchies and members into different groups.
- Two dimension types—CURRENCY and ANALYSIS—have only flat, single-level hierarchies.

  All other dimension types typically have multi-level hierarchies, for example:

    - Members of the ACCOUNT dimension type are the accounts from a general ledger chart of accounts. In a typical account hierarchy, Liabilities, Current Liabilities, and Accounts Payable are on different levels, as are Assets, Current Assets, and Inventory.
    - Members of a TIME dimension type are time periods of different lengths. In a typical time hierarchy, years, quarters, and months are on different levels.

This chapter describes how to create a new, empty dimension. You create a dimension within an existing dimension type. Therefore, the dimension type must already exist before you create the dimension.

For information about how to create a dimension type, see Chapter 10, “Adding a Dimension Type,” on page 65.

After you create a dimension, you must place members and hierarchies in it.

For information about how to place members and hierarchies in a dimension, see Chapter 7, “Loading Members and Hierarchies into a Dimension,” on page 33.

Creating a Dimension

A dimension is defined by a single dimension code. However, a dimension can also have names and descriptions in many data locales.

There are three methods that you can use to create a dimension. The method that you choose to use might vary depending on the number of dimensions that you are creating and the number of data locales for each dimension.

You can create a dimension by using one of the following methods:

1. In SAS Financial Management Studio by using the New Dimension wizard.
2. In SAS Data Integration Studio by using the solnsvc_2200_create_dimension job or by writing a job that uses the create_dimenstion transformation.
3. In SAS Data Integration Studio by using the APP_DIMENSION staging table and then loading the new dimension definition(s) into the Data Mart by running the solnsvc_2100_create_application_dimension job.

   Note: By using this method, you can define any number of dimensions in any number of data locales all at once by placing all of the necessary specifications in the APP_DIMENSION table.

Creating a Dimension by Using the New Dimension Wizard

To create a new dimension by using the New Dimension Wizard in SAS Financial Management Studio, complete the following steps:

1. In the Dimensions workspace, select **Create a new dimension** to launch the New Dimension wizard.
2. Proceed through the New Dimension wizard, referring to the online Help as necessary.
3. If you are using several data locales, after creating the new dimension, use the **Identification** tab of the dimension properties window in the Dimensions workspace to add names and descriptions in additional data locales.

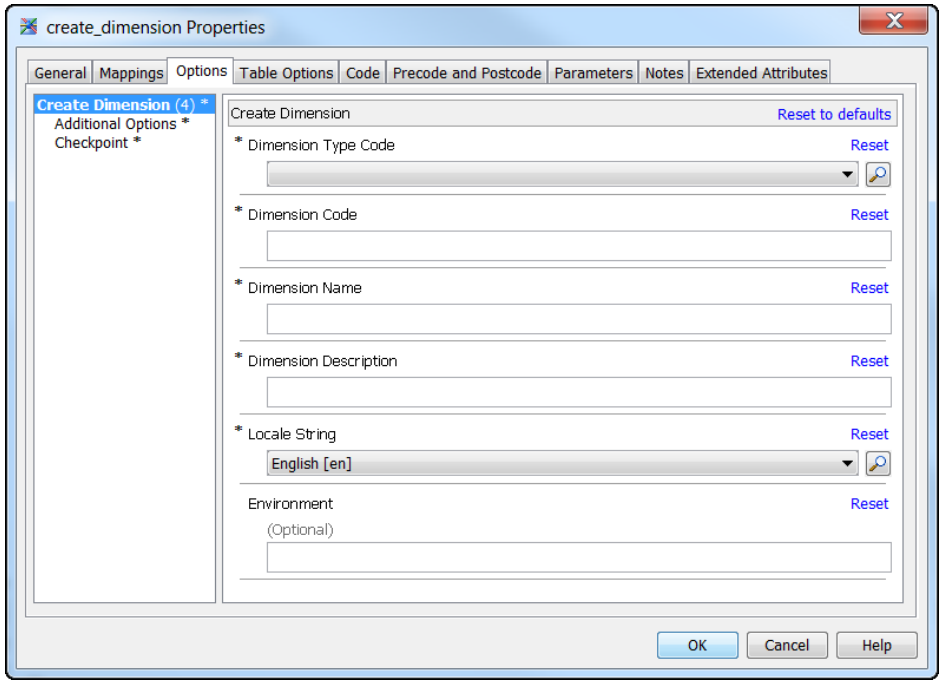
Creating a Dimension by Using a Job

To create a new dimension by writing a job that uses Create Dimension transformation, complete the following steps:

1. In SAS Data Integration Studio, select the **Folders** tab.
2. In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.
3. Copy the solnsvc_2200_create_dimension job.

   Note: It is a good practice to create and maintain a separate, appropriately named job for each set of option values. Changing the value of a job's options is possible, but likely to generate confusion.
4. Double-click the **solnsvc_2200_create_dimension** job in the list of jobs.
5. In the Job Editor window, right-click the create_dimension transformation, and select **Properties**. The create_dimension Properties window is displayed
6. Select the **Options** tab.

Display 6.1 *Create Dimension Properties Window — Options Tab View*



7 Enter values for the following options:

Option	Description
Dimension Type Code	Code of the dimension type within which the new dimension is created. To check the spelling of existing dimension type codes, use the Dimensions workspace of SAS Financial Management Studio.
Dimension Code	Unique code that identifies the new dimension. A valid value is 1 to 32 characters. You must use this code whenever you load members and hierarchies into the dimension. For information about loading members and hierarchies into a dimension, see “Loading Member and Hierarchy Data from the Staging Area into the Data Mart” on page 46).
Dimension Name	Name of the new dimension. The dimension name can be up to 50 characters and should identify the dimension in a way that is helpful to users. The name that you enter for the Dimension Name is associated with the data locale that you specify in the **Locale String** field. After you create the dimension, you can enter names and descriptions for other data locales by using the Dimensions workspace of SAS Financial Management Studio.

Option	Description
Dimension Description	Description of the new dimension. The dimension description can be up to 255 characters and should describe the dimension in a way that is helpful to users. The name and description that you supply here are associated with the data locale that you specify with the Locale String option. After you create the dimension, you can enter names and descriptions for other data locales by using the Dimensions workspace of SAS Financial Management Studio.
Locale String	Values defined in the LANGUAGE_CD column in the CODE_LANGUAGE table from which you select a value for the new dimension. For information about loading data locale codes, see Chapter 4, “Loading Language Codes and Data Locale Codes,” on page 17.
Environment	(Optional) Any environment that is defined in your sas-environment.xml file. If you leave this field empty, then the environment “default” is used.

8 Click **OK** to save your changes and close the window.

9 Select **File** ▸ **Save**.

10 In the Job Editor window, click **Run**.

11 When the job displays as completed in the Status column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

Creating a Dimension by Using a Staging Table

The staging table defined in the StageFM library to create one or more dimensions is the APP_DIMENSION table.

To load the APP_DIMENSION table with data, you must write and run a job that loads new dimension data into the table.

Note: By using this method, you can define any number of dimensions in any number of data locales all at once by placing all of the necessary specifications in the APP_DIMENSION table.

Before writing the job to load new dimension data into the APP_DIMENSION table, review the column structure of the table to ensure that the job that you write places the correct data in the correct columns.

To view the column structure of the APP_DIMENSION table, complete the following steps:

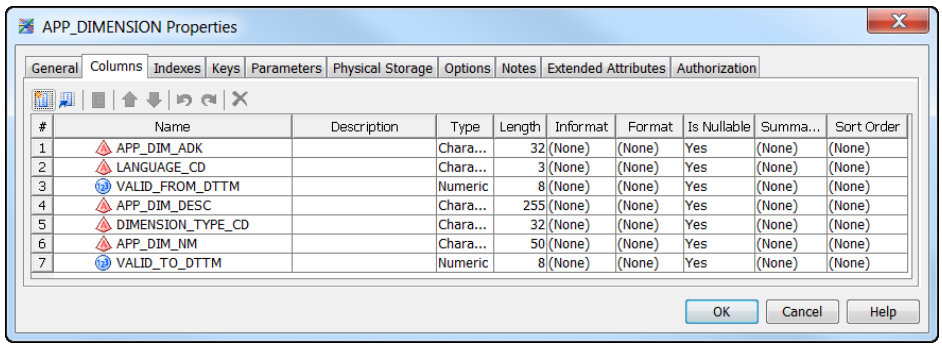
1 In SAS Data Integration Studio, select the **Folders** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.

3 Double-click **APP_DIMENSION** in the list of tables. The APP_DIMENSION Properties window is displayed.

4 Click the **Columns** tab to view the column structure of the table.

Display 6.2 *APP_DIMENSION Properties Window—Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	APP_DIM_ADK		Chara...	32	(None)	(None)	Yes	(None)	(None)
2	LANGUAGE_CD		Chara...	3	(None)	(None)	Yes	(None)	(None)
3	VALID_FROM_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
4	APP_DIM_DESC		Chara...	255	(None)	(None)	Yes	(None)	(None)
5	DIMENSION_TYPE_CD		Chara...	32	(None)	(None)	Yes	(None)	(None)
6	APP_DIM_NM		Chara...	50	(None)	(None)	Yes	(None)	(None)
7	VALID_TO_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)

The APP_DIMENSION table contains the following columns:

Column	Description
APP_DIM_ADK	Unique code that identifies the new dimension. You must use this code whenever you load members and hierarchies into the dimension. For information about loading members and hierarchies into a dimension, see "Loading Member and Hierarchy Data from the Staging Area into the Data Mart" on page 46.
LANGUAGE_CD	Identifies the Language Code that is in the CODE_LANGUAGE table. The data locale that the language code is associated with in the CODE_LANGUAGE table is the data locale that is associated with the name and description that you specify in the APP_DIM_ADK and APP_DIM_DESC columns.
VALID_FROM_DTTM	Defines the beginning of the lifespan of the record with a valid from datetime value.
APP_DIM_DESC	Identifies the new dimension description in a way that is helpful to users. The name and description that you supply in the APP_DIMENSION table are associated with the data locale that you specify indirectly with the LANGUAGE_CD. **Note:** To specify names and descriptions in several data locales for the same dimension, create several records that have the same APP_DIM_ADK value but different LANGUAGE_CD values.

Column	Description
DIMENSION_TYPE_CD	Identifies the code of the dimension type within which the new dimension is created. To verify the spelling of dimension type codes, use the Dimensions workspace of SAS Financial Management Studio.
APP_DIM_NM	Identifies the new dimension name in a way that is helpful to users.
VALID_TO_DTTM	Defines end of the lifespan of the record with a valid to datetime value. The name and description that you supply in the APP_DIMENSION table are associated with the data locale that you specify indirectly with the LANGUAGE_CD. **Note:** To specify names and descriptions in several data locales for the same dimension, create several records that have the same APP_DIM_ADK value but different LANGUAGE_CD values.

5 Click **OK** to close the APP_DIMENSION Properties window.

To load dimension data into the APP_DIMENSION table, complete the following steps:

1 Create a SAS Data Integration Studio job that loads data from its source into the APP_DIMENSION table in the StageFM library.

2 Click **OK** to save the job.

3 Run the job and validate that the table loaded successfully.

Note: For detailed information about creating and working with SAS Data Integration Studio jobs to extract data from a source table and load the data into a target table, see the *SAS Data Integration User's Guide*.

Loading New Dimensions into the Data Mart

The solnsv_2100_create_application_dimension job loads new dimensions into the Data Mart.

Note: Before loading the new dimension(s) into the Data Mart, ensure that the data locales for which you are loading dimension names and descriptions are already defined in the Data Mart. For information about loading data locale codes, see Chapter 4, "Loading Language Codes and Data Locale Codes," on page 17.

To load new dimension data from the APP_DIMENSION table into the Data Mart, complete the following steps:

1 In SAS Data Integration Studio, click the **Folders** tab.

2 In the Folders tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.

3 Copy the solnsv_2100_create_application_dimension job.

Note: It is a good practice to create and maintain a separate, appropriately named job for each set of option values. Changing the value of a job's options is possible, but likely to generate confusion.

4 Double-click the **solnsvc_2100_create_application_dimension** job in the list of jobs.

5 In the Job Editor window, click **Run**.

6 When the job displays as completed in the **Status** column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

7

Loading Members and Hierarchies into a Dimension

Overview

This chapter describes following topics:

- modifying the content of a dimension
- loading member and hierarchy data from the data source to staging tables
- loading member and hierarchy data from the staging area to the Data Mart

Modifying the Content of a Dimension

The content of most dimensions is modified by using an ETL job and the staging area of SAS Data Integration Studio. However, you can modify the members and

hierarchies of a dimension interactively by editing the members and hierarchies directly.

The following table describes each method that you can use to modify a dimension:

Table 7.1 Methods of Modifying a Dimension

Method	Description
Interactive (editing members and hierarchies)	In the Dimensions workspace of SAS Financial Management Studio, select **Show Members** and use the Members window to edit members and hierarchies.
ETL job (in SAS Data Integration Studio, by using the staging area)	■ From a third-party software product or another external source, load the members and hierarchies into the staging area. ■ From a SAS Financial Management system, load the staging area with members and hierarchies that you exported. This method is part of the content promotion facility for SAS Financial Management. You can copy the members and hierarchies of a dimension from one system to another system by exporting them from the source system and loading them into the target system. For information about exporting members and hierarchies, see Chapter 9, “Exporting and Promoting Members and Hierarchies,” on page 57.

Before modifying a dimension, note the following:

- You can load members and hierarchies through the staging area for only one dimension per dimension type.
- If you are using the GL_TRANSACTION_SUM table to supply financial accounting data to SAS Financial Management, every member that is used in the table must belong to a dimension that you load through the staging area.

  For information about loading data into a cycle, see Chapter 17, “Loading Base Data into a Financial Cycle,” on page 127.

Loading Member and Hierarchy Data into Staging Tables

Dimension Type Tables

To load members and hierarchies into a given dimension, you can use a set of SAS Financial Management tables for the dimension type to which that dimension belongs. These tables load the members and hierarchies into the staging area. From the staging area, you can then load the data into the Data Mart.

Note: You can use the staging area to load members and hierarchies into only one dimension per dimension type. You must populate the other dimensions in a dimension type interactively.

To load members and hierarchies into a dimension by using a table, complete the following steps:

1 In SAS Data Integration Studio, click the **Folders** tab.

2 In the Folders tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM** .

The StageFM folder contains a set of tables for each dimension type. For most dimension types, there are four tables that you can use to load data. For the Currency and Item category dimension types, there are only three tables.

The set of staging tables for each dimension type includes the following:

Primary Member Table

- The primary member table is specified in the TABLE_NM column in DIMENSION_TYPE table. For example, the primary member table for the Organization dimension type is INTERNAL_ORG.
- For most dimension types, the primary member table must contain a row for each member that you are loading, with text in the staging area default language.
- For the Currency and Item Category dimension types, the primary member table must contain all of the member records that you are loading, regardless of language. The Currency and Item Category primary member tables contain a Language Code column. This column identifies the language used in each record. The primary member tables for other dimension types do not have a Language Code column. The records for these dimension types use the staging area default language.
- The columns of the primary member tables differ from one dimension type to another because the members of different dimension types are characterized by different properties. The sections of this chapter on the Account dimension type, the Organization dimension type, and the Time dimension type include illustrations of the primary member tables for those dimension types.

  You can add other columns that represent custom properties to any primary member table. For information about adding additional columns to the primary member table see “About Member Properties” on page 51.

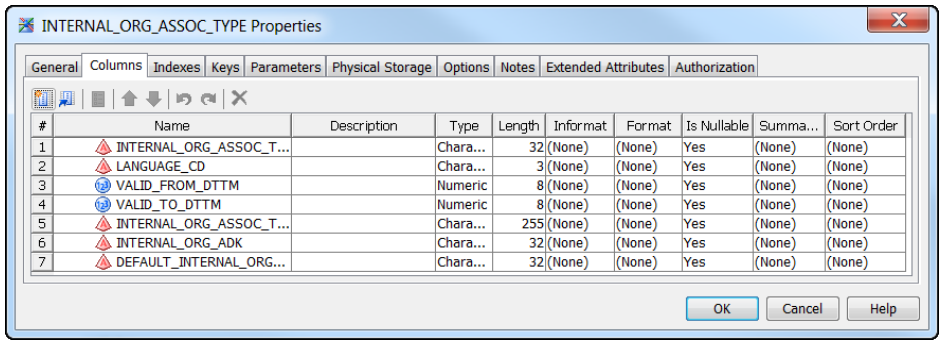
Secondary Member Table

- The secondary member table is specified in the NLS_TABLE_NM column in the DIMENSION_TYPE table. For example, the secondary member table for the Organization dimension type is INTERNAL_ORG_NLS
- For most dimension types, the secondary member table is the table in which to place any member records that use languages other than the default staging area language. For the Currency and Item Category dimension types, there is no secondary member table because their primary member tables can accommodate records in all languages.
- You can ignore the secondary member table if you are loading member records in only one language. If you use a secondary member table, then any member that you place in it must also be in the associated primary member table.

Hierarchy Identification Table

- The hierarchy identification table is the table is specified in the ASSOC_TYPE_TABLE_NM column in the DIMENSION_TYPE table. For example, the hierarchy identification table for the Organization dimension is INTERNAL_ORG_ASSOC_TYPE.
- The hierarchy identification table must contain a row for each hierarchy that you are loading into the target dimension. If you are loading hierarchy descriptions in more than one language, then this table must contain additional rows that describe the hierarchies in the other languages.
- The hierarchy identification tables have the same column structure for all dimension types, because identifying a hierarchy involves the same considerations regardless of dimension type. Some of the column names differ from table to table reflect the different dimension types, but the number, order, and characteristics of the columns are the same. Here are the columns of the INTERNAL_ORG_ASSOC_TYPE table:

Display 7.1 *INTERNAL_ORG_ASSOC_TYPE Properties Window — Columns View*

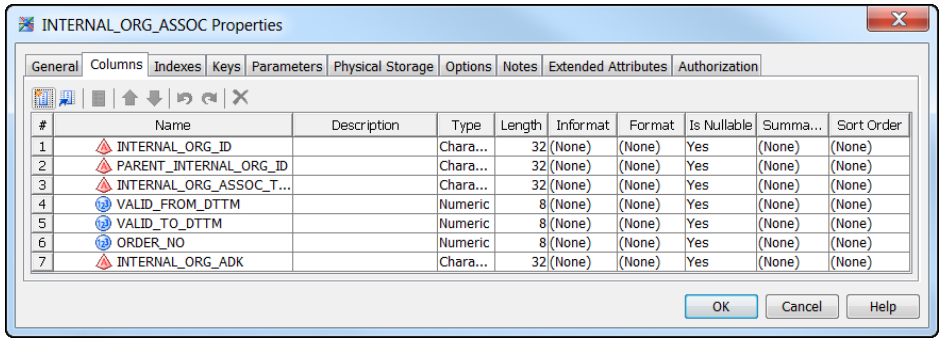


#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	INTERNAL_ORG_ASSOC_T...		Chara...	32	(None)	(None)	Yes	(None)	(None)
2	LANGUAGE_CD		Chara...	3	(None)	(None)	Yes	(None)	(None)
3	VALID_FROM_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
4	VALID_TO_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
5	INTERNAL_ORG_ASSOC_T...		Chara...	255	(None)	(None)	Yes	(None)	(None)
6	INTERNAL_ORG_ADK		Chara...	32	(None)	(None)	Yes	(None)	(None)
7	DEFAULT_INTERNAL_ORG...		Chara...	32	(None)	(None)	Yes	(None)	(None)

Hierarchy Structure Table

- The hierarchy structure table is the table that is specified in the ASSOC_TABLE_NM column in the table DIMENSION_TYPE. For example, the hierarchy structure table for the Organization dimension is INTERNAL_ORG_ASSOC.
- The hierarchy structure table must contain a row for each parent-child relationship within each hierarchy that you are loading into the target dimension. Each row of this table identifies a member, its parent member, and the hierarchy that the relationship is a part of. It also specifies the display position of the member in a fully expanded display of the hierarchy in the Dimensions workspace of SAS Financial Management Studio.
- The hierarchy structure tables have the same column structure for all dimension types because detailing a hierarchical structure involves the same considerations regardless of dimension type. Some of the column names differ from table to reflect the different dimension types, but the number, order, and characteristics of the columns are the same.

Display 7.2 *INTERNAL_ORG_ASSOC_Properties Window — Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	INTERNAL_ORG_ID		Chara...	32	(None)	(None)	Yes	(None)	(None)
2	PARENT_INTERNAL_ORG_ID		Chara...	32	(None)	(None)	Yes	(None)	(None)
3	INTERNAL_ORG_ASSOC_T...		Chara...	32	(None)	(None)	Yes	(None)	(None)
4	VALID_FROM_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
5	VALID_TO_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
6	ORDER_NO		Numeric	8	(None)	(None)	Yes	(None)	(None)
7	INTERNAL_ORG_ADK		Chara...	32	(None)	(None)	Yes	(None)	(None)

Note: When loading these tables, there are many points to keep in mind. The points that apply across all or most dimension types are discussed in "Requirements for All or Most Dimension Types" on page 37. The points that are specific to a particular dimension type are discussed in the subsequent sections of this chapter.

Requirements for All or Most Dimension Types

For any dimension type, the data that goes into the member tables, the hierarchy identification table, and the hierarchy structure table must meet the following conditions:

- If the primary member table has a Roll Up to Parent Flag column, then this column must have a value of either **Y** or **N**.

  In SAS Financial Management Studio, **Y** corresponds to selecting the **This member rolls up into its parent** check box on the **General** tab of the Properties window for a selected member. **N** corresponds to not selecting this check box. If you do not specify a value, the software provides the default value of **Y** when the data is loaded into the Data Mart.

- The hierarchy identification table must contain at least one record.

- In the hierarchy identification table, you can either specify a default member in each record or leave this column blank. If you leave the column blank, then a default member is designated automatically for each hierarchy when the hierarchies are loaded into the Data Mart. The automatically designated default member is the member in the first record of the hierarchy structure table that describes the relevant hierarchy and that makes a member its own parent.

  All the default members that you specify must also be in the primary member table. If you have several records for the same hierarchy in different languages, then either specify the same default member in all of them or leave them all blank.

- A member can be used in a hierarchy only if it is in the dimension to which the hierarchy belongs. In other words, any member that is in a parent-child record in the hierarchy structure table must also be in the primary member table.

- In the subset of the hierarchy structure table that describes a given hierarchy as of a given moment, each member that occurs as either a parent or a child must occur as a child in exactly one record:

  - If the member has a parent in that hierarchy at that moment, then that one record indicates which member is its parent.
  - If the member has no parent in that hierarchy at that moment, then that one record names the member as its own parent. This is how top-level members are identified.
- The Order Number column of the hierarchy structure table holds integers that determine the top-to-bottom display order of each parent's children in SAS Financial Management. Among each parent's children, the child with the lowest order number is displayed first, the child with the next lowest order number is displayed second, and so on.

  One approach to assigning order numbers is to assign a unique order number to every record in the table. Another approach is to start a fresh count for the children of each parent. The first approach gives the software more information than it needs. This is because the hierarchical structure already determines that each member displays as subordinate to its parent. However, you might find that a table with unique order numbers is easier to maintain than a table that reuses the same low numbers many times.

  With either approach, it is not necessary to use consecutive integers. For example, by numbering initially with multiples of ten you can provide room to insert new members without having to renumber old members.

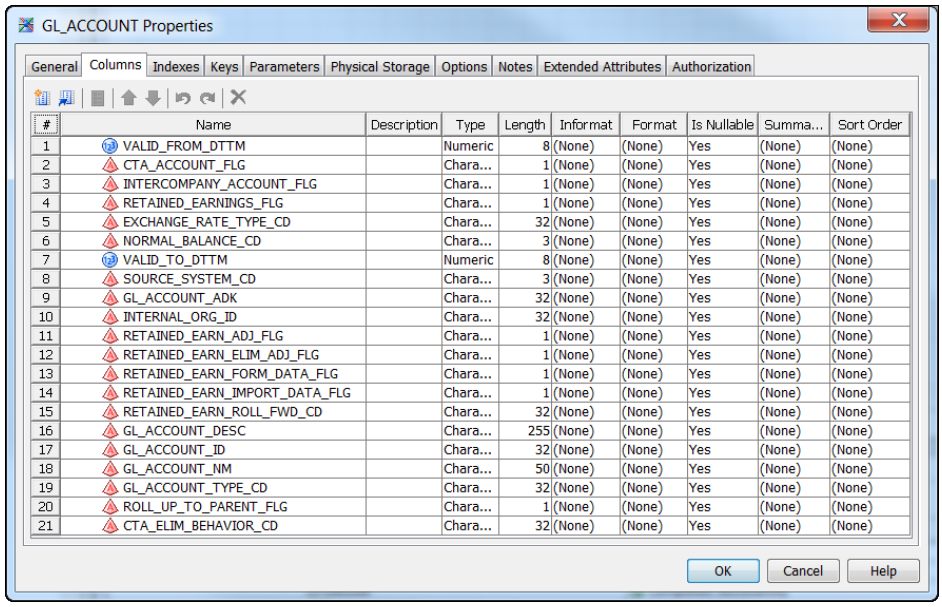
  If you leave this column blank in all the records of a hierarchy structure table, the software assigns default order numbers that reflect the order of the records in the table.
- The records of the hierarchy structure table can occur in any order, but it is a good idea to load this table so that the records are grouped by hierarchy.
- Each table includes a Valid From Datetime column and a Valid To Datetime column, which define the lifespans of its records.

Additional Requirements for the Account Dimension Type

The Account Primary Member Table

Each member of an Account dimension has properties that correspond to the columns of the GL_ACCOUNT table.

Display 7.3 *GL_ACCOUNT Properties Window — Columns View*

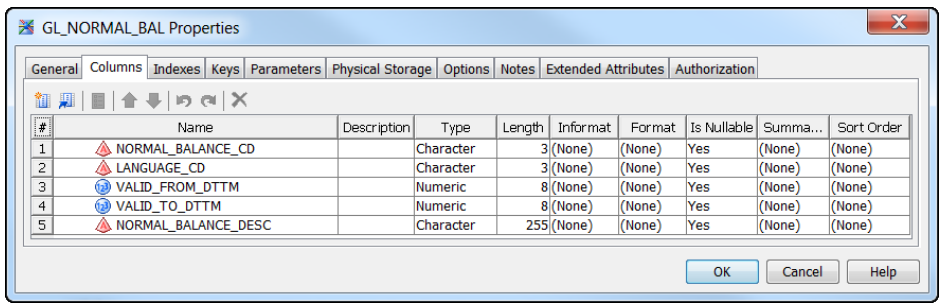


#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	VALID_FROM_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
2	CTA_ACCOUNT_FLG		Chara...	1	(None)	(None)	Yes	(None)	(None)
3	INTERCOMPANY_ACCOUNT_FLG		Chara...	1	(None)	(None)	Yes	(None)	(None)
4	RETAINED_EARNINGS_FLG		Chara...	1	(None)	(None)	Yes	(None)	(None)
5	EXCHANGE_RATE_TYPE_CD		Chara...	32	(None)	(None)	Yes	(None)	(None)
6	NORMAL_BALANCE_CD		Chara...	3	(None)	(None)	Yes	(None)	(None)
7	VALID_TO_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
8	SOURCE_SYSTEM_CD		Chara...	3	(None)	(None)	Yes	(None)	(None)
9	GL_ACCOUNT_ADK		Chara...	32	(None)	(None)	Yes	(None)	(None)
10	INTERNAL_ORG_ID		Chara...	32	(None)	(None)	Yes	(None)	(None)
11	RETAINED_EARN_ADJ_FLG		Chara...	1	(None)	(None)	Yes	(None)	(None)
12	RETAINED_EARN_ELIM_ADJ_FLG		Chara...	1	(None)	(None)	Yes	(None)	(None)
13	RETAINED_EARN_FORM_DATA_FLG		Chara...	1	(None)	(None)	Yes	(None)	(None)
14	RETAINED_EARN_IMPORT_DATA_FLG		Chara...	1	(None)	(None)	Yes	(None)	(None)
15	RETAINED_EARN_ROLL_FWD_CD		Chara...	32	(None)	(None)	Yes	(None)	(None)
16	GL_ACCOUNT_DESC		Chara...	255	(None)	(None)	Yes	(None)	(None)
17	GL_ACCOUNT_ID		Chara...	32	(None)	(None)	Yes	(None)	(None)
18	GL_ACCOUNT_NM		Chara...	50	(None)	(None)	Yes	(None)	(None)
19	GL_ACCOUNT_TYPE_CD		Chara...	32	(None)	(None)	Yes	(None)	(None)
20	ROLL_UP_TO_PARENT_FLG		Chara...	1	(None)	(None)	Yes	(None)	(None)
21	CTA_ELIM_BEHAVIOR_CD		Chara...	32	(None)	(None)	Yes	(None)	(None)

For each record in this table, note the following:

- VALID_FROM_DTTM and VALID_TO_DTTM define the lifespan of the record.
- Ignore the following columns; they are not used:
  - CTA_ACCOUNT_FLG
  - CTA_ELIM_BEHAVIOR_CD
  - GL_ACCOUNT_ADK
  - INTERNAL_ORG_ID
  - RETAINED_EARNINGS_FLG
- INTERCOMPANY_ACCOUNT FLG must be Y or N. In SAS Financial Management Studio, these values correspond to selecting or not selecting the **Intercompany** check box on the **Account Details** tab of the Properties window for a selected account.
- Normal Balance Code must be C (credit) or D (debit). These values are predefined in GL_NORMAL_BAL table.

Display 7.4 *View Data: GL_NORMAL_BAL Window*

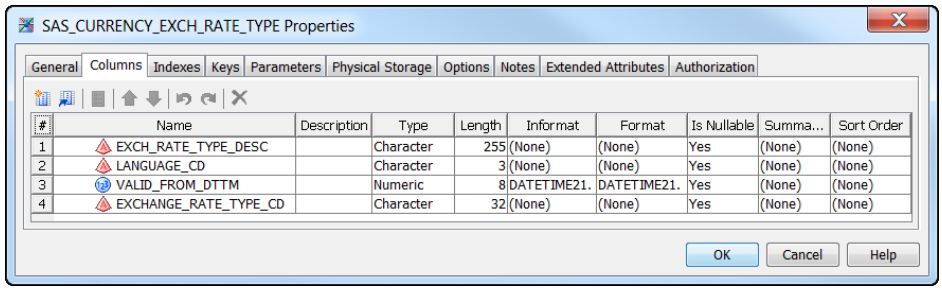


#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	NORMAL_BALANCE_CD		Character	3	(None)	(None)	Yes	(None)	(None)
2	LANGUAGE_CD		Character	3	(None)	(None)	Yes	(None)	(None)
3	VALID_FROM_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
4	VALID_TO_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
5	NORMAL_BALANCE_DESC		Character	255	(None)	(None)	Yes	(None)	(None)

In SAS Financial Management Studio, these values correspond to the **Credit** and **Debit** radio buttons on the **Account Details** tab of the Properties window for a selected account.

- Exchange Rate Type Code must be one of the predefined values in the EXCHANGE_RATE_TYPE_CD column of table SAS_CURRENCY_EXCH_RATE_TYPE.

Display 7.5 *SAS_CURRENCY_EXCH_TYPE_CD Properties Window — Columns View*

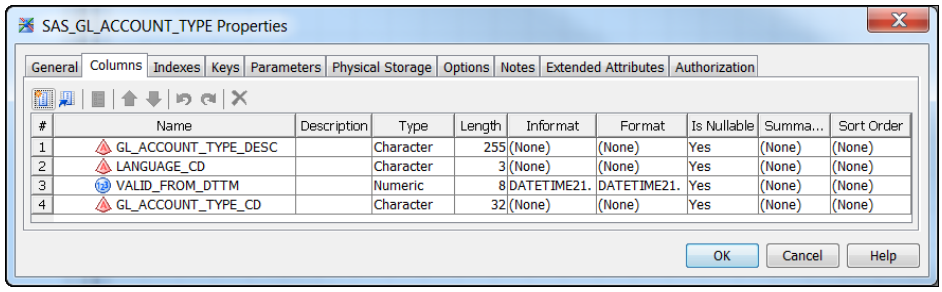


#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	EXCH_RATE_TYPE_DESC		Character	255	(None)	(None)	Yes	(None)	(None)
2	LANGUAGE_CD		Character	3	(None)	(None)	Yes	(None)	(None)
3	VALID_FROM_DTTM		Numeric	8	DATETIME21.	DATETIME21.	Yes	(None)	(None)
4	EXCHANGE_RATE_TYPE_CD		Character	32	(None)	(None)	Yes	(None)	(None)

In SAS Financial Management Studio, these values correspond to the available values for the **Exchange rate type** field on the **Account Details** tab of the Properties window for a selected account.

- Account Type Code must be one of the predefined values in the GL_ACCOUNT_TYPE_CD column of the SAS_GL_ACCOUNT_TYPE table.

Display 7.6 *SAS_GL_ACCOUNT_TYPE Properties Window — Columns View*

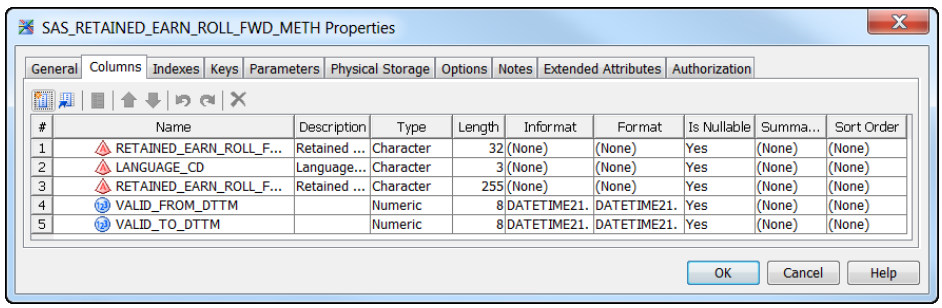


#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	GL_ACCOUNT_TYPE_DESC		Character	255	(None)	(None)	Yes	(None)	(None)
2	LANGUAGE_CD		Character	3	(None)	(None)	Yes	(None)	(None)
3	VALID_FROM_DTTM		Numeric	8	DATETIME21.	DATETIME21.	Yes	(None)	(None)
4	GL_ACCOUNT_TYPE_CD		Character	32	(None)	(None)	Yes	(None)	(None)

In SAS Financial Management Studio, these values correspond to the available values for the **Account type** field on the **Account Details** tab of the Properties window for a selected account.

- If Account Type Code is CTA, then Exchange Rate Type Code must not be Historic or Derived or None.
- If Account Type Code is RetainedEarnings, then Exchange Rate Type Code must be None and Retained Earnings Roll Forward Code requires a value. The value must be one of the predefined values in the SAS_RETAINED_EARN_ROLL_FWD_METH table.

Display 7.7 *SAS_RETAINED_EARN_ROLL_FWD_METH Properties Window — Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	RETAINED_EARN_ROLL_F...	Retained ...	Character	32	(None)	(None)	Yes	(None)	(None)
2	LANGUAGE_CD	Language...	Character	3	(None)	(None)	Yes	(None)	(None)
3	RETAINED_EARN_ROLL_F...	Retained ...	Character	255	(None)	(None)	Yes	(None)	(None)
4	VALID_FROM_DTTM		Numeric	8	DATETIME21.	DATETIME21.	Yes	(None)	(None)
5	VALID_TO_DTTM		Numeric	8	DATETIME21.	DATETIME21.	Yes	(None)	(None)

In SAS Financial Management Studio, these values correspond to the available values for the **Roll-forward method** field on the **Account Details** tab of the Properties window for a selected account. This field is on the tab only for accounts of the Retained Earnings account type.

- If Account Type Code is RetainedEarnings, then the four retained earnings flag columns require values. For each of these columns, the flag value must be either Y or N. In addition, the flag value must be Y for at least one column.

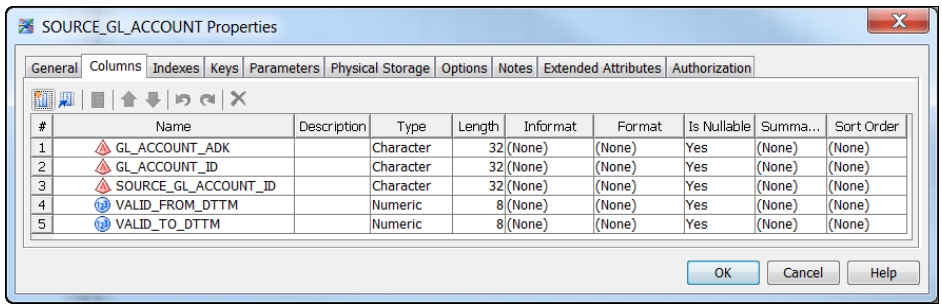
  In SAS Financial Management Studio, these flag columns correspond to the four **Basis data** check boxes on the **Account Details** tab of the Properties window for a selected account. These check boxes are on the tab only for accounts of the Retained Earnings account type. The flag values determine which members of the Source hierarchy to include in the crossings that contribute to the value of the account.

RE and CTA Source Accounts Table

For the Account dimension type, there is a fifth staging table in addition to the standard four. The fifth staging table is named SOURCE_GL_ACCOUNT.

If you use the GL_ACCOUNT table to load any accounts of the Retained Earnings or CTA account types, you must specify the source accounts for each Retained Earnings or CTA account by using the SOURCE_GL_ACCOUNT table.

Display 7.8 *SOURCE_GL_ACCOUNT Properties Window — Columns View*



For each record in the SOURCE_GL_ACCOUNT table, note the following:

- GL_ACCOUNT_ID must have the same value as the GL Account ID of the record in GL_ACCOUNT for which you are specifying a source account.
- SOURCE_GL_ACCOUNT_ID must be the code of the source account that you are specifying.
- If the account type of GL Account ID is CTA, then the account type of Source GL Account ID must be one of the following:
  - Asset
  - Equity
  - Liability
  - Retained Earnings
- If the account type of GL Account ID is Retained Earnings, then the account type of Source GL Account ID must be one of the following:
  - Asset

  - Equity
  - Expense
  - Liability
  - Revenue
- Valid From Datetime and Valid To Datetime define the lifespan of the record.
- GL_ACCOUNT_ADK is not used and can be ignored.

Account Type and Exchange Rate Type Constraints on Account Hierarchies

The following account types form a group known as *balance accounts*:

- Asset
- Equity
- Liability
- Statistical Balance

The following account types form a group known as *flow accounts*:

- Expense
- Revenue
- Statistical Flow

The following exchange rate types form a group known as *complex exchange rate types*:

- Derived (DER)
- Historic (HIS)

All the other exchange rate types are known as *simple exchange rate types*.

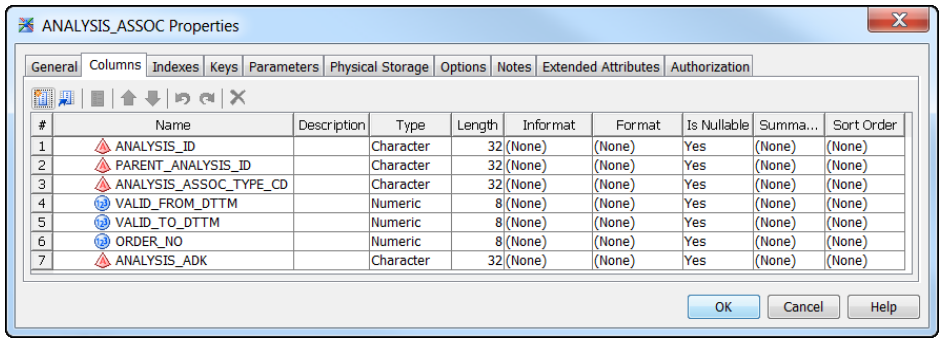
When using the GL_ACCOUNT_ASSOC table to define the parent and child relationships for an account hierarchy, you must observe the following constraints that involve account types and exchange rate types:

- The parent of a balance account, a Retained Earnings account, or a CTA account must be a balance account that uses a simple exchange rate type.
- The parent of a flow account must be a flow account that uses a simple exchange rate type.
- A child of a balance account that uses a simple exchange rate type must be either a balance account, a Retained Earnings account, or a CTA account.
- A child of a flow account that uses a simple exchange rate type must be a flow account.
- Retained Earnings accounts, CTA accounts, and all accounts that use complex exchange rate types must not have children.
- A Statistical (STA) account must have neither children nor a parent.

Additional Requirements for the Analysis Dimension Type

In the Analysis dimension type, every hierarchy must be flat. Every record that you load into the ANALYSIS_ASSOC hierarchy structure table must have the same analysis member code in the Analysis ID and Parent Analysis ID columns.

Display 7.9 *ANALYSIS_ASSOC Properties Window — Columns View*

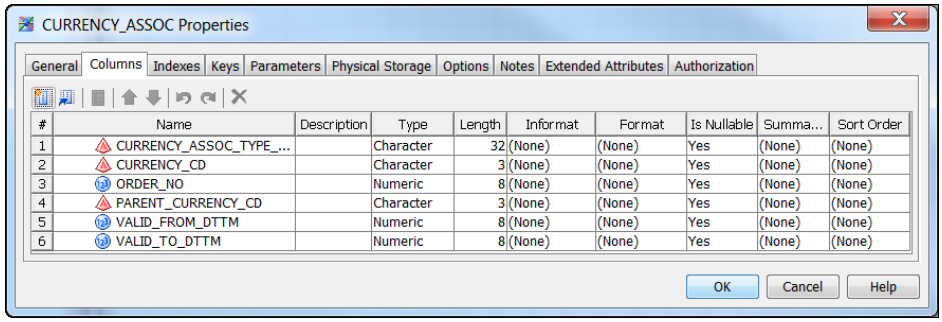


#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	ANALYSIS_ID		Character	32	(None)	(None)	Yes	(None)	(None)
2	PARENT_ANALYSIS_ID		Character	32	(None)	(None)	Yes	(None)	(None)
3	ANALYSIS_ASSOC_TYPE_CD		Character	32	(None)	(None)	Yes	(None)	(None)
4	VALID_FROM_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
5	VALID_TO_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
6	ORDER_NO		Numeric	8	(None)	(None)	Yes	(None)	(None)
7	ANALYSIS_ADK		Character	32	(None)	(None)	Yes	(None)	(None)

Additional Requirements for the Currency Dimension Type

In the Currency dimension type, every hierarchy must be flat. Every record that you load into the CURRENCY_ASSOC hierarchy structure table must have the same currency code in the Currency Code and Parent Currency Code columns:

Display 7.10 *CURRENCY_ASSOC Properties Window — Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	CURRENCY_ASSOC_TYPE_...		Character	32	(None)	(None)	Yes	(None)	(None)
2	CURRENCY_CD		Character	3	(None)	(None)	Yes	(None)	(None)
3	ORDER_NO		Numeric	8	(None)	(None)	Yes	(None)	(None)
4	PARENT_CURRENCY_CD		Character	3	(None)	(None)	Yes	(None)	(None)
5	VALID_FROM_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
6	VALID_TO_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)

You load the CURRENCY_ASSOC table from the SAS_CURRENCY table of predefined data by running the solnsvc_0210_load_stagefm_currency_table job.

Note: The CURRENCY_ASSOC table is the only dimension staging table for which you do not need to write your own job.

Additional Requirements for the Organization Dimension Type

The INTERNAL_ORG table must contain two special members, which are not visible in the software. One special member is defined by an Internal

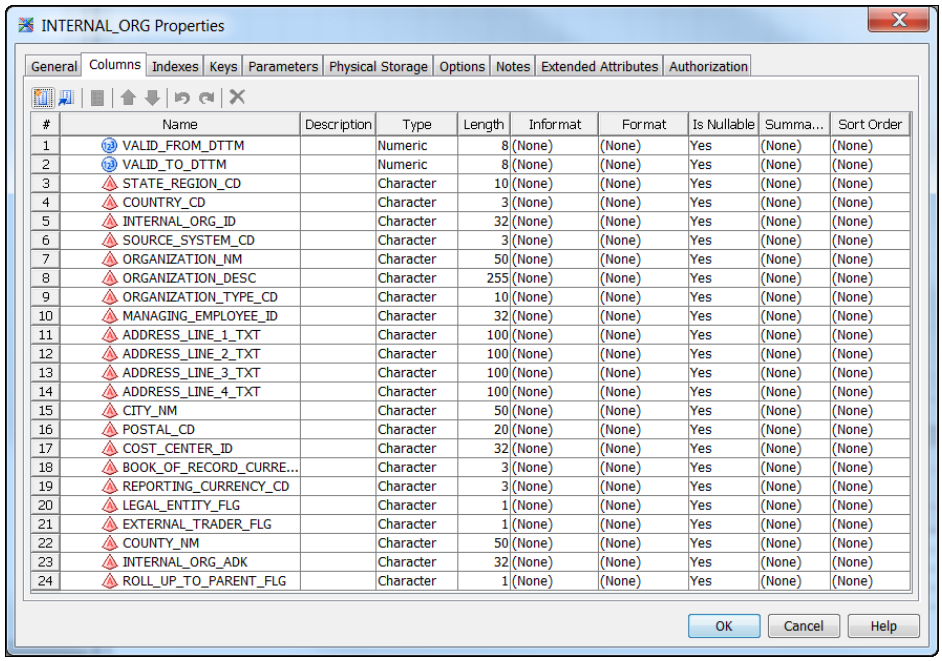
Organization ID of ALL. The other special member is defined by an Internal Organization ID of EXT.

The ALL and EXT members must be part of every hierarchy that is defined in the INTERNAL_ORG_ASSOC table. In every organization hierarchy, ALL must be the unique top member, and EXT must be a leaf that is directly under ALL. The formal constraints are as follows:

- ALL must not have a parent. This is indicated by a record in which ALL is its own parent.
- ALL must be the only member of the hierarchy that does not have a parent.
- ALL must be the parent of EXT.
- EXT must not be the parent of any member.

Each member of an Internal Organization dimension has properties that correspond to the columns of the INTERNAL_ORG table.

Display 7.11 *INTERNAL-ORG Properties Window — Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	VALID_FROM_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
2	VALID_TO_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
3	STATE_REGION_CD		Character	10	(None)	(None)	Yes	(None)	(None)
4	COUNTRY_CD		Character	3	(None)	(None)	Yes	(None)	(None)
5	INTERNAL_ORG_ID		Character	32	(None)	(None)	Yes	(None)	(None)
6	SOURCE_SYSTEM_CD		Character	3	(None)	(None)	Yes	(None)	(None)
7	ORGANIZATION_NM		Character	50	(None)	(None)	Yes	(None)	(None)
8	ORGANIZATION_DESC		Character	255	(None)	(None)	Yes	(None)	(None)
9	ORGANIZATION_TYPE_CD		Character	10	(None)	(None)	Yes	(None)	(None)
10	MANAGING_EMPLOYEE_ID		Character	32	(None)	(None)	Yes	(None)	(None)
11	ADDRESS_LINE_1_TXT		Character	100	(None)	(None)	Yes	(None)	(None)
12	ADDRESS_LINE_2_TXT		Character	100	(None)	(None)	Yes	(None)	(None)
13	ADDRESS_LINE_3_TXT		Character	100	(None)	(None)	Yes	(None)	(None)
14	ADDRESS_LINE_4_TXT		Character	100	(None)	(None)	Yes	(None)	(None)
15	CITY_NM		Character	50	(None)	(None)	Yes	(None)	(None)
16	POSTAL_CD		Character	20	(None)	(None)	Yes	(None)	(None)
17	COST_CENTER_ID		Character	32	(None)	(None)	Yes	(None)	(None)
18	BOOK_OF_RECORD_CURRE...		Character	3	(None)	(None)	Yes	(None)	(None)
19	REPORTING_CURRENCY_CD		Character	3	(None)	(None)	Yes	(None)	(None)
20	LEGAL_ENTITY_FLG		Character	1	(None)	(None)	Yes	(None)	(None)
21	EXTERNAL_TRADER_FLG		Character	1	(None)	(None)	Yes	(None)	(None)
22	COUNTY_NM		Character	50	(None)	(None)	Yes	(None)	(None)
23	INTERNAL_ORG_ADK		Character	32	(None)	(None)	Yes	(None)	(None)
24	ROLL_UP_TO_PARENT_FLG		Character	1	(None)	(None)	Yes	(None)	(None)

When building records for this table, note the following:

- VALID_FROM_DTTM and VALID_TO_DTTM define the lifespan of the record.
- Reporting Currency Code corresponds to the Functional Currency property in SAS Financial Management Studio. You must provide a valid currency code for each organization, including ALL and EXT.

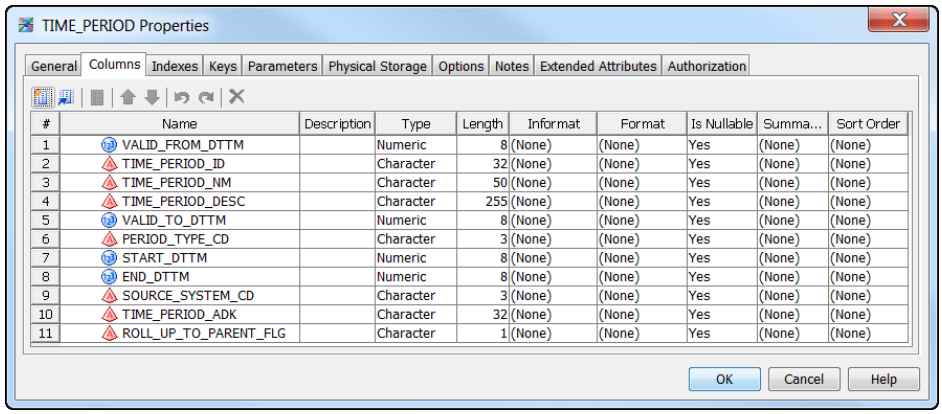
  If you are not using SAS Financial Management, then you can specify any currency code for each organization.
- Book of Record Currency Code is not used.
- Legal Entity Flag corresponds to the Reporting Entity property in SAS Financial Management Studio. Use Y for any organization that is a reporting entity and N for any organization that is not a reporting entity. For ALL and EXT, use N.

- The columns that contain geographical and address information are used by SAS Human Capital Management but not by SAS Financial Management or SAS Strategy Management. If you are not using SAS Human Capital Management, then leave these columns blank.
- Internal Organization ADK is not used.

Additional Requirements for the Time Dimension Type

Each member of a Time dimension has properties that correspond to the columns of the TIME_PERIOD table.

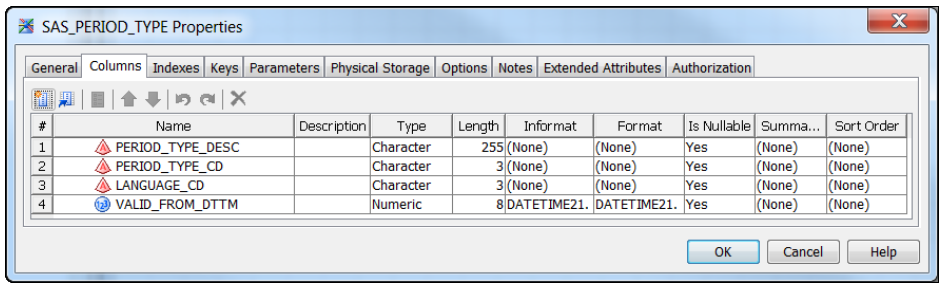
Display 7.12 *TIME_PERIOD Properties Window — Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	VALID_FROM_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
2	TIME_PERIOD_ID		Character	32	(None)	(None)	Yes	(None)	(None)
3	TIME_PERIOD_NM		Character	50	(None)	(None)	Yes	(None)	(None)
4	TIME_PERIOD_DESC		Character	255	(None)	(None)	Yes	(None)	(None)
5	VALID_TO_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
6	PERIOD_TYPE_CD		Character	3	(None)	(None)	Yes	(None)	(None)
7	START_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
8	END_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
9	SOURCE_SYSTEM_CD		Character	3	(None)	(None)	Yes	(None)	(None)
10	TIME_PERIOD_ADK		Character	32	(None)	(None)	Yes	(None)	(None)
11	ROLL_UP_TO_PARENT_FLG		Character	1	(None)	(None)	Yes	(None)	(None)

For each record in this table, note the following:

- VALID_FROM_DTTM and VALID_TO_DTTM define the lifespan of the record.
- Start Date and End Date define the time period that the member represents. You must place counts of seconds from January 1, 1960:00:00:00 in both of these columns even though the software shows only calendar dates. Do not put counts of days from January 1, 1960 in these columns.
- Period Type Code must be one of the codes in the SAS_PERIOD_TYPE table.



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	PERIOD_TYPE_DESC		Character	255	(None)	(None)	Yes	(None)	(None)
2	PERIOD_TYPE_CD		Character	3	(None)	(None)	Yes	(None)	(None)
3	LANGUAGE_CD		Character	3	(None)	(None)	Yes	(None)	(None)
4	VALID_FROM_DTTM		Numeric	8	DATETIME21.	DATETIME21.	Yes	(None)	(None)

- Time Period ADK. It is not used.

Users Tab Data

For the dimensions of every dimension type except for Analysis, Currency, and Time, the member Properties window in SAS Financial Management Studio includes a **Users** tab. You can load the user-member associations that you can view and edit with this tab.

These user-member associations can serve a useful purpose. In SAS Financial Management, a user who has a **Users** tab user-member association with a certain dimension member is authorized to enter data into any planning form that is assigned to that dimension member.

To load **Users** tab information, use the APP_USER_X_MEMBER staging table. For information about the columns of this table, see *SAS Financial Management: Data Model Reference*.

You also use the APP_USER_X_MEMBER table when you promote **Users** tab information from one SAS Financial Management system to another. For information about promoting dimension content, see Chapter 9, “Exporting and Promoting Members and Hierarchies,” on page 57.

Security Tab Data

For the dimensions of every dimension type, the Properties window in SAS Financial Management Studio includes a **Security** tab. You can load the security for the dimension type, hierarchies, dimension members, and custom properties for this dimension. In addition, you can load security for the dimension, which you can view and edit with this tab.

These user-member and group-member associations control Read access to the data that is associated with the relevant members in SAS Financial Management reports. The online Help for the **Security** tab contains examples of the security structures that you can build. For more information about the security structures that you can build, see the *SAS Financial Management: User's Guide*.

To load **Security** tab user-member associations, use the APP_USER_ACTIONS staging table. To load **Security** tab group-member associations, use the APP_GROUP_ACTIONS staging table. For information about the columns of these tables, see the *SAS Financial Management: Data Model Reference*.

The APP_USER_ACTIONS and APP_GROUP_ACTIONS tables are also used when you promote **Security** tab information from one SAS Financial Management system to another. For information about promoting dimension content, see Chapter 9, “Exporting and Promoting Members and Hierarchies,” on page 57.

Loading Member and Hierarchy Data from the Staging Area into the Data Mart

Overview

You can load members and hierarchies into a dimension in the Data Mart by using either one of the following methods:

- a SAS Data Integration Studio job that uses the import_dimension transformation
- the Load Dimension wizard in the Dimensions workspace of SAS Financial Management Studio

Typically, you can load your dimensions in any order. The only exception is that you must load currencies into a Currency dimension before you load organizations into an Organization dimension.

The data locales for which you are loading member and hierarchy names and descriptions must be defined in the Data Mart before you load the member and hierarchy data. For details about loading data locales, see Chapter 4, “Loading Language Codes and Data Locale Codes,” on page 17.

Loading Data by Using a Job

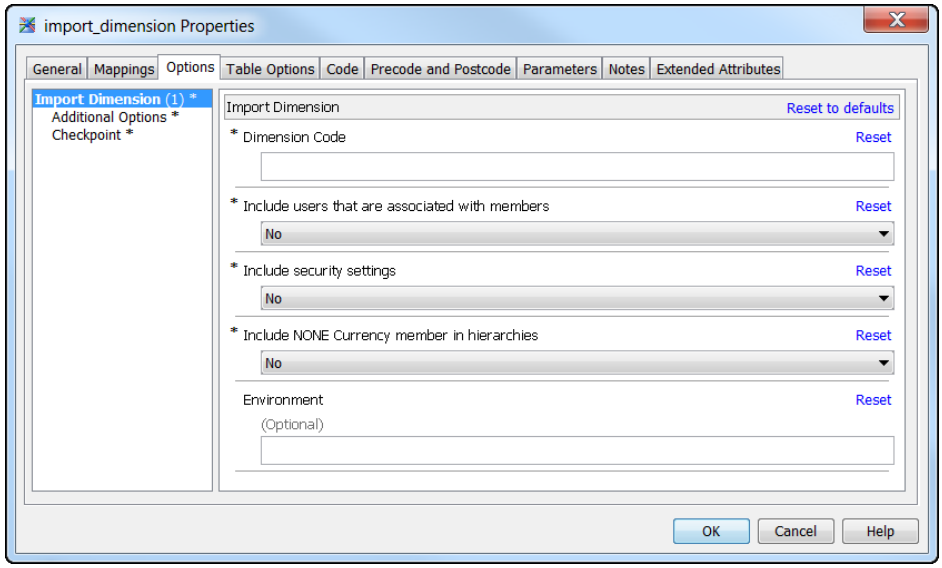
The solnsvc_3200_load_dimension job loads member and hierarchy data from the staging area into the Data Mart.

To load data by using the solnsvc_3200_load_dimension job, complete the following steps:

1. In SAS Data Integration Studio, select the **Folder** tab.
2. In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.
3. Make a copy of the solnsvc_3200_load_dimension job.

   Note: It is a good practice to create and maintain a separate, appropriately named job for each set of option values. Changing the value of a job’s options is possible, but likely to generate confusion.
4. Double-click the **solnsvc_3200_load_dimension** job in the list of jobs.
5. In the Job Editor window, right-click the **Import Dimension** transformation and then select **Properties**. The import_dimensions Properties window is displayed.
6. Select the **Options** tab.

Display 7.13 *import_dimension Properties Window — Options View*



7 Enter values for the following options:

Option	Description
Dimension Code	Code of the target dimension. You can look up the code in the Dimensions workspace of SAS Financial Management Studio.
Include users that are associated with members	Specifies whether you are importing the user-member associations that can be viewed in SAS Financial Management Studio on the **Users** tab of the Properties window. Select **Yes** if you are importing this information. Select **No** if you are not importing this information. If you select **Yes**, then all information of this type for the target dimension is deleted from the Data Mart before the new information is imported.
Include security settings	Specifies whether you are importing the security for the dimension type, hierarchies, dimension members, and custom properties for the dimension, as well as security for the dimension. You can view the security settings in SAS Financial Management Studio on the **Security** tab of the Properties window for each of these objects. Select **Yes** if you are importing this information. Select **No** if you are not importing this information. If you select **Yes**, then all information of this type for the target dimension is deleted from the Data Mart before the new information is imported.
Include NONE Currency member in hierarchies	In general, currency members are predefined and have three-character codes, such as EUR, JPY, and USD. The only exception is the NONE currency, which has a four-character code and is not predefined. To use NONE, you must add it to a currency hierarchy in the Dimensions workspace.

Option	Description
environment	(Optional) Any environment that is defined in your sas-environment.xml file. If you leave this field empty, then the environment “default” is used.

8 Click **OK** to save your changes and close the Properties window.

9 Select **File** ▸ **Save**.

10 In the Job Editor window, click **Run**.

11 When the job displays as completed in the **Status** column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

Loading Data by Using the Load Dimension Wizard in SAS Financial Management Studio

To load data into the Data Mart by using the Load Dimension wizard in the Dimensions workspace of SAS Financial Management Studio, complete the following steps.

1 Select the target dimension from the displayed list of dimensions.

2 Select **Load Dimension** to launch the Load Dimension wizard.

3 Proceed through the Load Dimension wizard, referring to the online Help as necessary.

When the load process is complete, a window appears from which you can view an HTML report of the results.

Viewing the Summary of Results

Whether you load members and hierarchies by using a SAS Data Integration Studio job or using the SAS Financial Management Studio Load Dimension wizard, the results are the same:

- All of the staging area data in the dimension-type-specific tables for the relevant dimension type is loaded. This includes the data in the primary and secondary member tables, the hierarchy identification table, and the hierarchy structure table. For an Account dimension, it also includes the data in the SOURCE_GL_ACCOUNT table.
- Each member that you load replaces the existing member that has the same code. Any existing member that is not replaced by a newly loaded member remains in the target dimension.
- For each member that you load, any associated formula data is also loaded. Associated **Security** tab data and **User** tab data is loaded only if you set the

relevant flags to **Y**. You must load any dimension that you use in a formula before the dimension with which the formula is associated.

- Each hierarchy that you load replaces the entire existing hierarchy that has the same code. Any existing hierarchy that is not replaced by a newly loaded hierarchy remains in the target dimension.

8

Registering Member Properties

Overview

This chapter describes the following topics:

- member properties
- defining new member properties
- registering member properties
- using member properties that you have registered

About Member Properties

Member properties are defined in the *primary member* table. When viewing the primary members table, note the following:

- Some columns contain information that is specific to a dimension type. The columns that contain dimension-type-specific information represent *member properties*. Examples of the columns that contain member properties include Account Type for the Account dimension type and Functional Currency for the Organization dimension type.
- Some of the columns in a primary member table contain information that is common across all or most dimension types. These generic columns represent member *attributes* that are not classified as member properties. Examples of these columns that contain information common across all dimension types include Code, Name, Description, Valid from Datetime, Valid to Datetime, and Roll Up to Parent Flag.

In addition, when working with member properties, note the following:

- When loading members into a dimension into the Data Mart, the information that loads includes generic columns and the values of member properties that are registered to be loaded.
- Many but not all member properties are preregistered in the software. For the Account and Time dimension types, all predefined member properties are preregistered. For the Organization dimension type, the predefined “Reporting Currency Code” and the “Legal Entity Flag” member properties are preregistered.

  You can register member properties, including member properties that you add to the staging area and member properties that are predefined in the staging area but not preregistered.

Defining New Member Properties

To define new member properties in the staging area for any dimension type, complete the following steps:

1. In SAS Data Integration Studio, select the **Folders** tab.
2. In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.
3. Double-click the relevant source primary member table in the list of tables. The Properties window for the table is displayed.
4. Click the **Columns** tab.
5. Click the **New Column** icon ()to add a column for the new property.
6. Click **OK** to save your changes and close the Properties window.
7. Right-click the table name in the list of tables and select **Update Metadata** from the pop-up menu.
8. Modify the job that you wrote to load the source primary member table so that it loads values into the new column.

For example, to define a new member property for the Account dimension type, you would add a column for the new member property to the GL_ACCOUNT table. Then, you would right-click the GL_ACCOUNT table, select **Update Metadata**, and modify the job that loads the GL_ACCOUNT table so that it loads values into the column that you added.

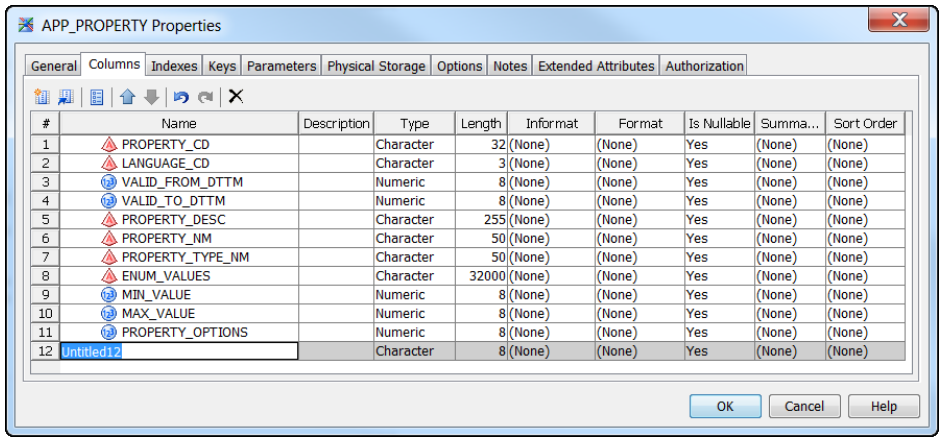
Registering Member Properties

To register a predefined member property, or a member property that you have added to the staging area, add a row that describes the property to the APP_PROPERTY table. Then, add a row to the APP_MEMBER_PROPERTY_MAP table that associates the property with the column in the dimension member table that contains its values.

To register a member property that you have added to the staging area, complete the following steps:

1. In SAS Data Integration Studio, select the **Folders** tab.
2. In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.
3. Double-click **APP_PROPERTY** in the list of tables. The APP_PROPERTY Properties window is displayed.
4. Click the **Columns** tab. In the Columns view, add a row that describes the property by clicking on the New Column ( ) icon.

Display 8.1 *APP_PROPERTY Properties Window — Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	PROPERTY_CD		Character	32	(None)	(None)	Yes	(None)	(None)
2	LANGUAGE_CD		Character	3	(None)	(None)	Yes	(None)	(None)
3	VALID_FROM_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
4	VALID_TO_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
5	PROPERTY_DESC		Character	255	(None)	(None)	Yes	(None)	(None)
6	PROPERTY_NM		Character	50	(None)	(None)	Yes	(None)	(None)
7	PROPERTY_TYPE_NM		Character	50	(None)	(None)	Yes	(None)	(None)
8	ENUM_VALUES		Character	32000	(None)	(None)	Yes	(None)	(None)
9	MIN_VALUE		Numeric	8	(None)	(None)	Yes	(None)	(None)
10	MAX_VALUE		Numeric	8	(None)	(None)	Yes	(None)	(None)
11	PROPERTY_OPTIONS		Numeric	8	(None)	(None)	Yes	(None)	(None)
12	Untitled12		Character	8	(None)	(None)	Yes	(None)	(None)

The APP_PROPERTY table contains the following columns:

Column	Description

PROPERTY_CD	Property code that is used in the staging tables.
	Must contain the same value as the Property Code column of the corresponding record in the APP_MEMBER_PROPERTY_MAP table.
	Do not use the following reserved property codes:
	`AccountBehavior`
	`AccountType`
	`BalanceType`
	`BasisData`
	`BookCurrency`
	`EndDate`
	`ExchangeRateType`
	`Formula`
	`FormulaId`
	`FormulaPrecedence`
	`FormulaScope`
	`FormulaType`
	`FunctionalCurrency`
	`Intercompany`
	`Level`
	`ReportingEntity`
	`RollForwardMethod`
	`SourceAccounts`
	`StartDate`
	`TotalAfterImport`

LANGUAGE_CD	Language code.
VALID_FROM_DTTM	Moment that begins the time period during which a row of data is valid.
VALID_TO_DTTM	Moment that ends the time period during which a row of data is valid.
PROPERTY_DESC	Description of the member property. The description can be up to 255 characters and should describe the member property in a way that is helpful to users.
PROPERTY_NM	Name of member property.

PROPERTY_TYPE_NM	Identifies the data type of the property's values. This column must contain one of the following strings:
	`boolean`
	`date`
	`double`
	`integer`
	`string`

ENUM_VALUES	String of validation values for string type properties that are separated by commas (for example, red,blue,green).
MIN_VALUE	Specifies a minimum numeric value for integer, double, or date type properties. For a date custom property, the value should be a numeric value in a yyyymmdd format.

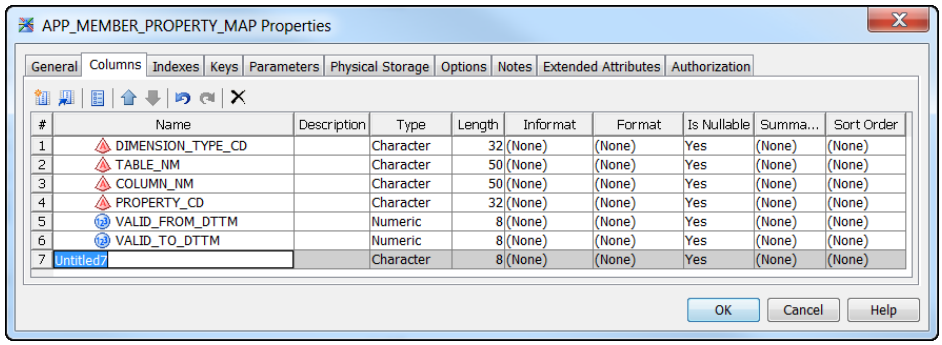
Column	Description
MAX_VALUE	Specifies a maximum numeric value for integer, double, or date type properties. For a date custom property, the value should be a numeric value and should have the following form: yyyymmdd.
PROPERTY_OPTIONS	Specifies whether property validation active. A value of 0 specifies that property validation is inactive and a value of 1 specifies that property validation active.

5 Click **OK** to save your changes and close the Properties window.

6 Double-click the **APP_MEMBER_PROPERTY_MAP** table from the list of tables. The APP_MEMBER_PROPERTY_MAP Properties window is displayed.

7 Click the **Columns** tab. In the Columns view, click the New Column icon () to add a row that associates the member property with the column in the APP_PROPERTY table that describes the property.

Display 8.2 *APP_MEMBER_PROPERTY_MAP Properties Window — Column View*



The APP_MEMBER_PROPERTY_MAP table contains the following columns:

Column	Description
DIMENSION_TYPE_CD	One of the values in the DIMENSION_TYPE_CD column of the DIMENSION_TYPE table. These values include the codes of the dimension types that you created as described in Chapter 10, “Adding a Dimension Type,” on page 65 and the codes of the predefined dimension types in the SAS_DIMENSION_TYPE table. **Note:** The SAS_DIMENSION_TYPE table is supplied by SAS.
TABLE_NM	Name of the primary member table for the specified dimension type. This name is in the TABLE_NM column of the DIMENSION_TYPE table.
COLUMN_NM	Name of the column that contains the values of the property.

Column	Description
PROPERTY_CD	Property code. Must contain the same value as the Property Code column of the corresponding record in the APP_PROPERTY table.
VALID_FROM_DTTM	Moment that begins the time period during which a row of data is valid.
VALID_TO_DTTM	Moment that ends the time period during which a row of data is valid.

8 Click **OK** to save your changes and close the APP_MEMBER_PROPERTY_MAP Properties window.

Using Member Properties That You Have Registered

After you load the additional member properties into the Data Mart, you can view their values in SAS Financial Management Studio by using either the Members view or the Hierarchies view.

For information about how to load members into the Data Mart, see Chapter 7, “Loading Members and Hierarchies into a Dimension,” on page 33.

To view information about the new member properties in SAS Financial Management Studio, complete the following steps:

1 In the Dimensions workspace, right-click on the appropriate dimension and select **Members** from the pop-up menu.

2 Right-click on the member and select **Properties** ▸ **Custom Properties**.

Note: The following SAS Financial Management functions retrieve values of the member property:

- the PROPERTY function in SAS Financial Management Studio
- the CDAProperty and fmProperty functions in the SAS Financial Management Add-in for Microsoft Excel

Note: If you need SAS Financial Management to do anything additional with the values of the custom properties, contact your SAS consultant about customizing the software.

9

Exporting and Promoting Members and Hierarchies

Overview

This chapter describes the following topics:

- exporting members and hierarchies
- using a job to export members and hierarchies
- using the Export Dimension Wizard to export members and hierarchies
- reviewing the details of the results
- possible obstacles to exporting a dimension

About Exporting Members and Hierarchies

When you export members and hierarchies, you can choose the export destination. You can also choose whether to export the **Users** tab and **Security** tab information in the Properties window in SAS Financial Management Studio for the members that you are exporting.

The following two scenarios might require that you to export members and hierarchies from SAS Financial Management. The scenario determines the method that you use:

1 You created members by using the **Dimensions** workspace of SAS Financial Management Studio. Creating members by using the **Dimensions**

workspace of SAS Financial Management Studio requires that the members be in the appropriate staging area dimension tables.

Now, you want to use these members in base accounting facts to be loaded through the GL_TRANSACTION_SUM and GL_JRNL_DETAILS tables of the staging area.

In this scenario, the appropriate export destination is the staging area that serves the Data Mart from which you are exporting.

Note: Because the information in the **Users** tab and **Security** tab is not used in the process of loading base accounting facts, there is no reason to export it.

2 You have created or modified members and hierarchies by using the **Dimensions** workspace of SAS Financial Management Studio. Now, you want to promote these members and hierarchies to a test system or to a production system.

In this scenario, the appropriate export destination is a Base SAS library other than the staging area that serves the Data Mart from where you are exporting. After you export the members and hierarchies to this library, you must move them to the staging area that serves the system that is your promotion target. From the staging area dimension tables for your promotion target, you load the members and hierarchies into the Data Mart for your promotion target.

For information about loading members and hierarchies, see "Loading Member and Hierarchy Data from the Staging Area into the Data Mart" on page 46.

Note: Before you can export the data, the tables must exist in the export library.

Depending on how you manage the **Users** tab and the **Security** tab information across the two systems, you might or might not want to export **Users** tab and **Security** tab information.

Export members and hierarchies only for those dimensions that you load through the staging area. Remember that you must choose a single dimension per dimension type to load with members and hierarchies through the staging area.

You can export the members and hierarchies of a dimension by using two methods:

- By running a SAS Data Integration Studio job that uses the Export Dimension transformation.
- By running the Export Dimension wizard in the **Dimensions** workspace of SAS Financial Management Studio.

Both methods yield the same result. Both methods are available regardless of the reason for the export operation.

Exporting Members and Hierarchies by Using a Job

The solnsvc_4100_export_dimension job exports members and hierarchies.

To export members and hierarchies by using the solnsvc_4100_export_dimension job, complete the following steps:

1 In SAS Data Integration Studio, select the **Folders** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Managment** ▸ **5.5 Jobs**.

3 Make a copy of the solnsvc_4100_export_dimension job.

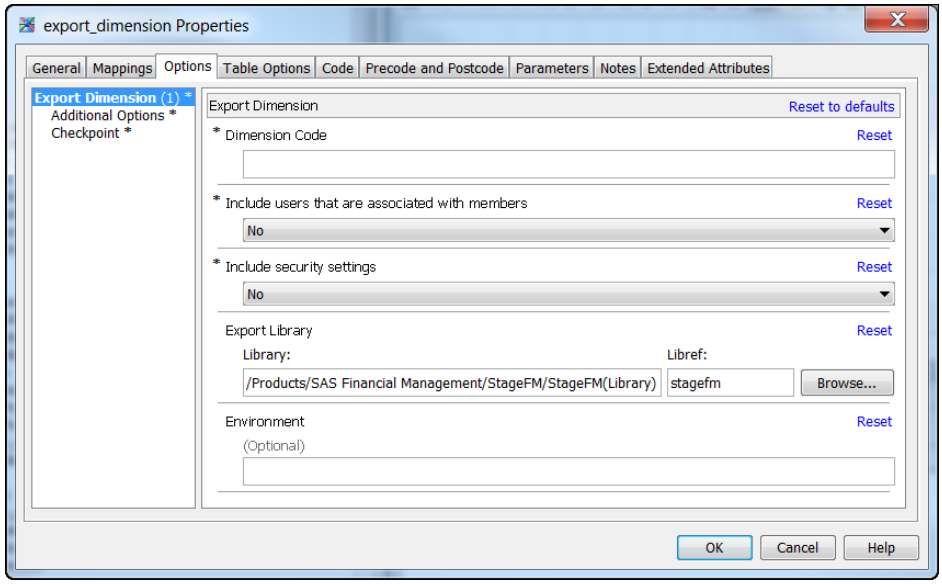
Note: It is a good idea to create and maintain a separate, appropriately named job for each set of option values. Changing the option values of a job occasionally is possible, but it is likely to generate confusion.

4 Double-click **solnsvc_4100_export_dimension** in the list of jobs. The job is displayed in the Job Editor window.

5 Right-click the **export_dimension transformation** and select **Properties** from the pop-up menu. The export_dimension Properties window is displayed.

6 Select the **Options** tab.

Display 9.1 *export-dimension Properties Window — Options View*



7 Enter values for the following options:

Option	Description
Dimension Code	Code of the source dimension. To look up existing codes, use the **Dimensions** workspace of SAS Financial Management Studio.
Include users that are associated with members	Specifies whether to export the user-member associations that can be viewed in SAS Financial Management Studio on the **Users** tab of the member properties window. The value is a Yes/No flag. Select **No** in order to withhold these user-member associations from the exported information. Select **Yes** to export members and hierarchies in order to promote them to another dimension or to another system.

Option	Description
Include security settings	Specifies whether to export security for the dimension type, hierarchies, dimension members, and custom properties for this dimension, as well as security for the dimension itself. You can view the security settings in SAS Financial Management Studio on the **Security** tab of the properties window for each of these objects. The value is a Yes/No flag.. Select **No** to withhold these security settings from the exported information. Select **Yes** to export the security settings in order to promote them to another dimension or to another system.
Export Library	Name of the Base SAS data library to which are exporting the data to. Click **Browse** to select a library. For example, select **`StageFM`** if you are exporting members and hierarchies to the staging area. If you specify a target library other than **`StageFM`**, then make sure that the target library meets the following requirements: ■ The library is on a machine that uses the same operating system as the machine that holds the source Data Mart. ■ The Solutions Host User has operating system Read and Write access to the library. ■ The library contains copies of all the staging tables that are needed to receive the exported data. These staging tables include the following: □ Dimension-type-specific tables for each dimension type with which you are working. For the Account dimension type, you need copies of the following five tables: GL_ACCOUNT, GL_ACCOUNT_ASSOC_TYPE, GL_ACCOUNT_ASSOC, GL_ACCOUNT_NLS, and SOURCE_GL_ACCOUNT. For most other dimension types, you need the counterparts of the first four of these tables. For the Currency and Item Category dimension types, you need the counterparts of the first three. □ Tables that contain formula information across all dimension types that support formulas: APP_FORMULA, APP_FORMULA_TARGET, APP_FORMULA_READ_MEMBER, and APP_FORMULA_WRITE_MEMBER. □ Tables that contain **Security** tab data across all dimension types: APP_GROUP_ACTIONS and APP_USER_ACTIONS. □ The table that contains **User** tab data across all dimension types except Analysis, Currency, and Time (which do not support **User** tab data): APP_USER_X_MEMBER. **Note:** To define additional Base SAS libraries, use SAS Management Console.
Environment	(Optional) Any environment that is defined in your sas-environment.xml file. If you leave this field empty, then the environment “default” is used.

8 Click **OK** to save your changes and close the export_dimension Properties window.

9 Select **File** ▸ **Save**.

10 In the Job Editor window, click **Run**.

11 When the job displays as completed in the **Status** tab of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

Exporting Members and Hierarchies by Using the Export Dimension Wizard

To export the members and hierarchies of a selected dimension by using the Export Dimension wizard, complete the following steps:

1 In the **Dimensions** workspace of SAS Financial Management Studio, select the source dimension.

2 Select **Export this dimension** to launch the Export Dimension wizard.

3 Proceed through the wizard, referring to the online Help as necessary.

Note: If you specify an export library other than `StageFM`, then the export library must satisfy all the conditions that are listed in "Exporting Members and Hierarchies by Using a Job" on page 58.

Reviewing the Details of the Results

The two methods of exporting members and hierarchies produce the same result, which includes the following:

- All the data in the target dimension-type-specific tables is deleted and replaced with the data that you are exporting. At the end of the process, these tables contain only the data that you just exported.
  - If the Data Mart contains member or hierarchy names and descriptions in more than one data locale, the export includes names and descriptions in each data locale that is defined in the CODE_LANGUAGE table. For information about the CODE_LAUAGE table, see Chapter 4, "Loading Language Codes and Data Locale Codes," on page 17.
  - The names and descriptions for the data locale that is associated with the staging area default language are exported to the primary member table. The names and descriptions for all other data locales are exported to the secondary member table. For information about the primary member and secondary member tables, see "Dimension Type Tables" on page 34.
- All the data in the target formula tables for the dimension type with which you are working is deleted and replaced with the formula data that you are exporting. At the end of the process, these tables contain the newly exported

formula data for the dimension type with which you are working. However, these tables contain the previously existing formula data for all other dimension types.

- If you export **Security** tab data, data in the target **Security** tab tables for the dimension type with which you are working is deleted and replaced with the **Security** tab data that you export.

  In this scenario, at the end of the process, these tables contain only the newly exported **Security** tab data for the dimension type with which you are working. However, these tables contain the previously existing **Security** tab data for all other dimension types.

  If you choose to not export **Security** tab data, then the export operation does not change the target **Security** tab tables in any way.

- If you export **User** tab data, data in the target **User** tab table for the dimension type with which you are working is deleted and replaced with the **User** tab data that you export.

  At the end of the process, this table contains only the newly exported **User** tab data for the dimension type with which you are working. However, this table contains the previously existing **User** tab data for all other dimension types.

  If you choose to not export **User** tab data, then the export operation does not change the target **User** tab table in any way.

Possible Obstacles to Exporting a Dimension

The solnsvc_4100_export_dimension job and the Export Dimension wizard can encounter various obstacles that prevent them from successfully exporting the members and hierarchies of the selected dimension.

Note: If the job or the wizard encounters any of these obstacles, an error message is displayed.

The possible obstacles to exporting a dimension include the following:

- The Solutions Host User does not have operating system Read and Write access to the target data library.
- A target table does not exist.
  - If the target data library is the staging area, this condition can occur if a table was accidentally deleted. This condition can also occur if the staging tables for the dimension type were never created. For information about creating a dimension type, see Chapter 10, “Adding a Dimension Type,” on page 65.
  - For a target data library other than the staging area, this condition can occur if you neglected to copy one of the necessary tables into the target library.
- A column is either misnamed or missing from a target table. This condition can occur if the target tables were not created correctly.

- The record for the relevant dimension type in the DIMENSION_TYPE table contains an error. This condition can occur if an incorrect value was placed in the record when it was created.
- One of the target tables is open and locked. This condition can occur if someone is working with the table.
- The CODE_LANGUAGE table has more than one record that is marked with a Default Language Flag value of Y. For more information about the CODE_LANGUAGE table, see Chapter 4, “Loading Language Codes and Data Locale Codes,” on page 17.
- The CODE_LANGUAGE table does not have a record for one of the languages that are used in the member and hierarchy data that you want to export.

10

Adding a Dimension Type

Overview

This chapter describes the following topics:

- dimension types
- adding a dimension type
- running the job that creates a new dimension type in the staging tables
- running the job that loads a dimension type into the staging table
- loading new dimension types into the Data Mart
- creating dimensions in a new dimension type
- loading members and hierarchies into dimensions that belongs to a new dimension type

About Dimension Types

Dimension types are a part of a *cycle*. A cycle is a structure pool of data. In SAS Financial Management, the following seven dimension types are required in a cycle:

- Account
- Analysis

- Currency
- Frequency
- Organization
- Time
- Source
- Trader (a mirror of Organization)

Note: The Source dimension and Frequency dimension are defined at implementation.

The SAS Financial Management software includes a set of predefined dimension types. These dimension types are defined in the SAS_DIMENSION_TYPE table in SAS Data Integration Studio.

If the predefined set of dimension types does not meet the needs of your site, you can add additional dimension types. This chapter describes how to add a dimension type.

Adding a Dimension Type

Note: To add a dimension type, you must complete the tasks in the entire chapter in order, without skipping any steps. To add two or more dimension types, repeat the steps that are described in this chapter for each dimension type that you are adding.

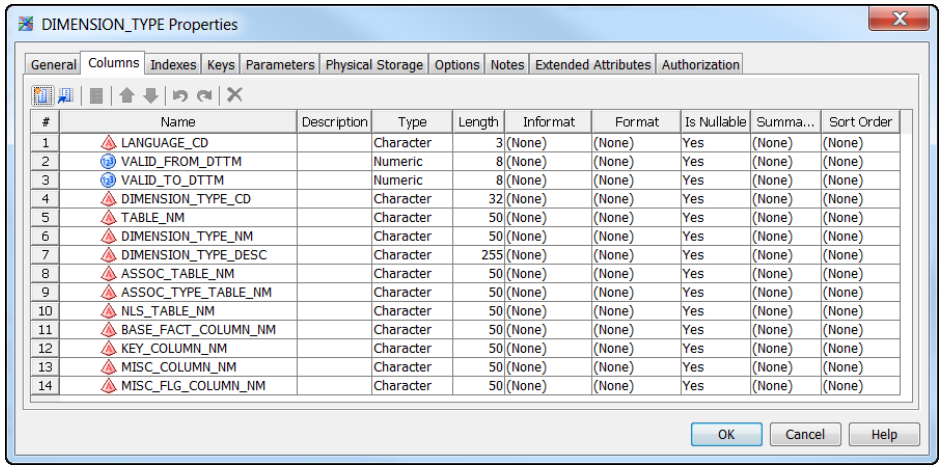
The staging table defined for dimension types is the DIMENSION TYPE table.

To load the table with data, you must write and run a job that loads the data into the table. Before you write a job to load data into the table, review the column structure of the tables to ensure that the job that you write places the correct data in the correct columns.

To view the column structure of the DIMENSION_TYPE table, complete the following steps:

1. In SAS Data Integration Studio, select the **Folders** tab.
2. In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.
3. Double-click **DIMENSION TYPE** in the list of tables. The DIMENSION TYPE Properties window is displayed.
4. Click the **Columns** tab to view the column structure.

Display 10.1 *DIMENSION_TYPE Table Properties Window — Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	LANGUAGE_CD		Character	3	(None)	(None)	Yes	(None)	(None)
2	VALID_FROM_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
3	VALID_TO_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
4	DIMENSION_TYPE_CD		Character	32	(None)	(None)	Yes	(None)	(None)
5	TABLE_NM		Character	50	(None)	(None)	Yes	(None)	(None)
6	DIMENSION_TYPE_NM		Character	50	(None)	(None)	Yes	(None)	(None)
7	DIMENSION_TYPE_DESC		Character	255	(None)	(None)	Yes	(None)	(None)
8	ASSOC_TABLE_NM		Character	50	(None)	(None)	Yes	(None)	(None)
9	ASSOC_TYPE_TABLE_NM		Character	50	(None)	(None)	Yes	(None)	(None)
10	NLS_TABLE_NM		Character	50	(None)	(None)	Yes	(None)	(None)
11	BASE_FACT_COLUMN_NM		Character	50	(None)	(None)	Yes	(None)	(None)
12	KEY_COLUMN_NM		Character	50	(None)	(None)	Yes	(None)	(None)
13	MISC_COLUMN_NM		Character	50	(None)	(None)	Yes	(None)	(None)
14	MISC_FLG_COLUMN_NM		Character	50	(None)	(None)	Yes	(None)	(None)

The DIMENSION_TYPE table contains the following columns:

Column	Description
LANGUAGE_CD	Code that identifies the language and locale for names and descriptions. The code must be defined in the CODE_LANGUAGE table. An example is "en" for English.
VALID_FROM_DTTM	Defines the beginning of the lifespan of the record with a valid from datetime value.
VALID_TO_DTTM	Defines end of the lifespan of the record with a valid to datetime value.
DIMENSION_TYPE_CD	Code for a dimension type that is provided as input to the job that loads this table.
TABLE_NM	Name of the table that contains members that belong to the dimension type that is identified in the DIMENSION_TYPE_CD column. Some examples of member tables are ANALYSIS, GL_ACCOUNT, and INTERNAL_ORG.
DIMENSION_TYPE_NM	Name of the dimension type that is identified in the DIMENSION_TYPE_CD column.
DIMENSION_TYPE_DESC	Description of the dimension type that is identified in the DIMENSION_TYPE_CD column.
ASSOC_TABLE_NM	Name of the table that details the parent-child relationships of hierarchies that belong to the dimension type that is identified in the DIMENSION_TYPE_CD column. Some examples of parent-child tables are ANALYSIS_ASSOC, GL_ACCOUNT_ASSOC, and INTERNAL_ORG_ASSOC.

Column	Description
ASSOC_TYPE_TABLE_NM	Name of the table that identifies the hierarchies that belong to the dimension type that is identified in the DIMENSION_TYPE_CD column. Some examples of hierarchy identification tables are ANALYSIS_ASSOC_TYPE, GL_ACCOUNT_ASSOC_TYPE, and INTERNAL_ORG_ASSOC_TYPE.
NLS_TABLE_NM	Name of dimension's NLS table.
BASE_FACT_COLUMN_NM	Name of the column of the dimension type's member table that contains the member codes.
KEY_COLUMN_NM	Name of the column of the dimension type's member table that contains the member codes.
MISC_COLUMN_NM	Name of the column in fact tables other than the GL_TRANSACTION_SUM table that identifies members that belong to the dimension type that is identified in the DIMENSION_TYPE_CD column.
MISC_FLG_COLUMN_NM	Name of the FLG column in the SUPP_SCHEDULE_FACT table that identifies members as (D) Dimension or (P) Property that belong to the dimension type that is identified in the DIMENSION_TYPE_CD column.

5 Click **OK** to close the DIMENSION_Type Properties window.

To create a new dimension type, the DIMENSION_TYPE table must be updated. To update the table, run the solnsvc_0100_create_a_new_dimension_type job.

Creating a New Dimension Type in the Staging Tables

The solnsvc_0100_create_a_new_dimension_type job creates a new dimension type in staging tables.

When you run the solnsvc_0100_create_a_new_dimension_type job, the job performs the following:

- In the SOURCE_DIMENSION_TYPE table, it places a row that describes a specified new dimension type. The SOURCE_DIMENSION_TYPE table is a supplementary source table for the solnsvc_0100_create_a_new_dimension_type job. The physical table name and the metadata name is SOURCE_DIMENSION_TYPE.
- It creates the four staging tables that you use to load members and hierarchies into a dimension that belongs to the new dimension type. For information about loading members and hierarchies into a dimension, see

Chapter 7, “Loading Members and Hierarchies into a Dimension,” on page 33.

- (Optional) It adds a column that holds member codes that belong to the new dimension type to the following staging tables:
  - GL_TRANSACTION_SUM
  - GL_JRNL_DETAILS
  - MISC_RATE
  - CURRENCY_COMPLEX_EXCH_RATE
  - SUPP_SCHEDULE_FACT

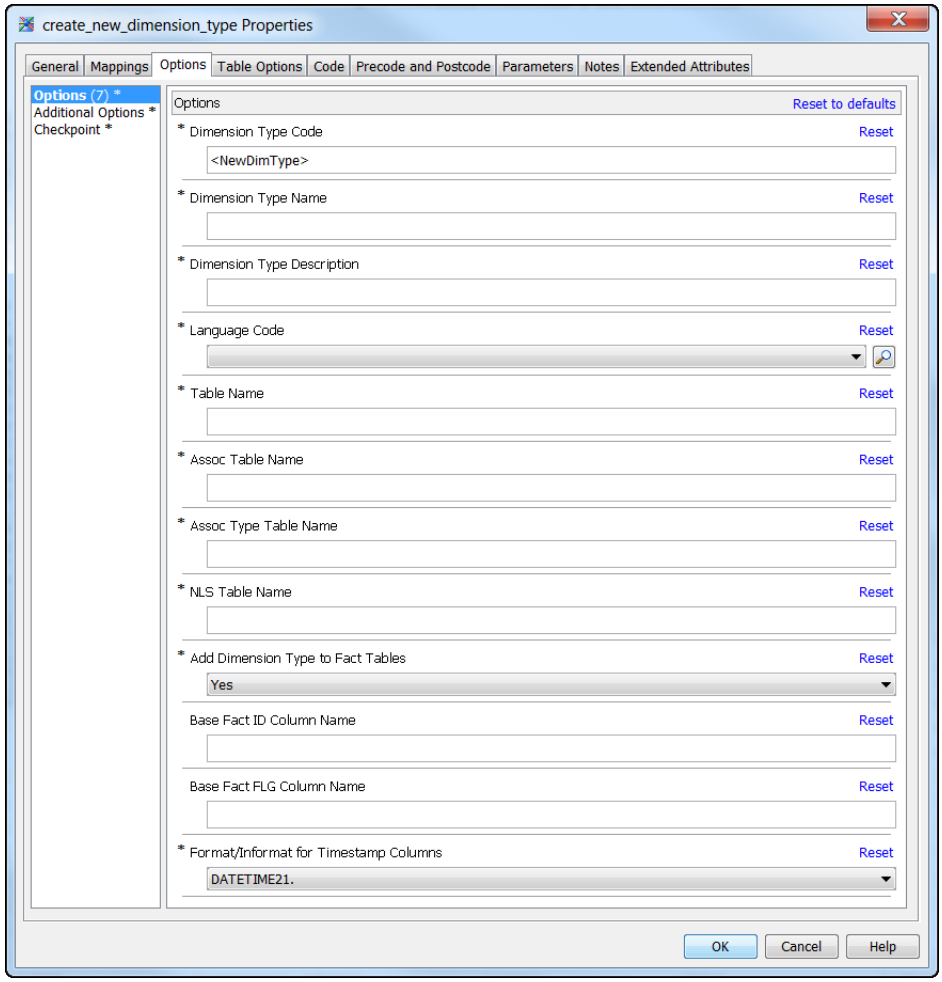
  Note: These tables need the additional columns only if the new dimension type is used to describe financial accounting data for SAS Financial Management.

Before running solnsvc_0100_create_a_new_dimension_type job, configure the job options as described in the following steps:

1. In SAS Data Integration Studio, select the **Folders** tab.
2. In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.
3. Make a copy of the solnsvc_0100_create_a_new_dimension_type job.

   Note: It is a good practice to create and maintain a separate, appropriately named job for each set of option values. Changing the value of a job’s options is possible, but likely to generate confusion.
4. Double-click **solnsvc_0100_create_a_new_dimension_type** in the list of jobs.
5. In the Job Editor window, right-click the **create_newdimension_type transformation** and select **Properties** from the pop-up menu.
6. In the **Properties** window, select the **Options** tab.

Display 10.2 *create_new_dimension_type Properties Window — Options View*



7 Enter values for the following options:

Option	Description
Dimension Type Code	Code of the new dimension type. This code can be up to 32 characters long, and it can include special characters.
Dimension Type Name	Name of the new dimension type. The name should identify the dimension in a way that is helpful to users.
Dimension Type Description	Description of the new dimension type. The description should identify the dimension in a way that is helpful to users.
Language Code	One of the language codes in the CODE_LANGUAGE table. Select the appropriate language code for the dimension name and description that you have provided. For information about loading language codes, see Chapter 4, “Loading Language Codes and Data Locale Codes,” on page 17.

Option	Description
Table Name	Name of the primary member table for the new dimension type. To use the naming convention of the predefined dimension types, make this table name identical to the dimension type code.
Assoc Table Name	Name of the hierarchy structure table for the new dimension type. To use the naming convention of the predefined dimension types, make this table name identical to *code*_ASSOC, where *code* is the dimension type code. The table name must be 32 characters or less, and it cannot contain special characters.
Assoc Type Table Name	Name of the hierarchy identification table for the new dimension type. To use the naming convention of the predefined dimension types, make this table name identical to *code*_ASSOC_TYPE, where *code* is the dimension type code. The table name must be 32 characters or less, and it cannot contain special characters.
NLS Table Name	Name of the secondary member table for the new dimension type. To use the naming convention of the predefined dimension types, make this table name identical to *code*_NLS, where *code* is the dimension type code. The table name must be 32 characters or less, and it cannot contain special characters.
Add Dimension Type to Fact Tables	Specifies whether to add a column for the new dimension type to the GL_TRANSACTION_SUM and GL_JRNL_DETAILS tables and their corresponding staging tables. In this case, you must specify name for these columns by using the Base Fact Column Name and Business ID Column Name options. The value for this field is a Yes/No flag. If you select **No**, then no column is added to these tables. Select **Yes** if you want the new dimension type to be used to describe financial accounting data for use in SAS Financial Management.
Base Fact ID Column Name	Name of the column that is added to the GL_TRANSACTION_SUM, GL_JRNL_DETAILS, MISC_RATE, CURRENCY_ COMPLEX_EXCH_RATE, and SUPP_SCHEDULE_FACT tables. To use the naming convention of the predefined dimension types, make this table name identical to *code*_ID, where *code* is the dimension type code. The column name must be 32 characters or less, and it cannot contain special characters. If you select **No** for the **Add Dimension Type to Fact Tables** option, then leave this option blank.
Base Fact FLG Column Name	This column is added to the SUPP_SCHEDULE_FACT table and is used to specify whether the dimension ID specified is a dimension member or a property. This field is required if the add dimension type to fact tables is selected.
Format/Informat for Timestamp Columns	Determines the format to use for time stamps in the four tables that hold member and hierarchy data for the new dimension type.

8 Click **OK** to save your changes and close the Properties window.

9 Select **File** ▸ **Save**.

10 In the Job Editor window, click **Run**.

11 When the job displays as completed in the Status column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

12 Select **View** ▸ **Refresh** to refresh the metadata so that the tables for the new dimension type appear.

If you specified PRODUCT as the value of the Dimension Type Code option, then you should see the following tables in the **Folders** ▸ **Products** ▸ **SAS Financial Management** ▸ **StageFM** folder:

- PRODUCT
- PRODUCT_ASSOC
- PRODUCT_ASSOC_TYPE
- PRODUCT_NLS

Loading a Dimension Type into the Staging Table

The solnsvc_0200_load_stagefm_dimension_type_table job loads a dimension type into the staging table.

To load a dimension type into a staging table, complete the following steps:

1 In SAS Data Integration Studio, select the **Folders** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.

Make a copy of the solnsvc_0200_load_stagefm_dimension_type_table job.

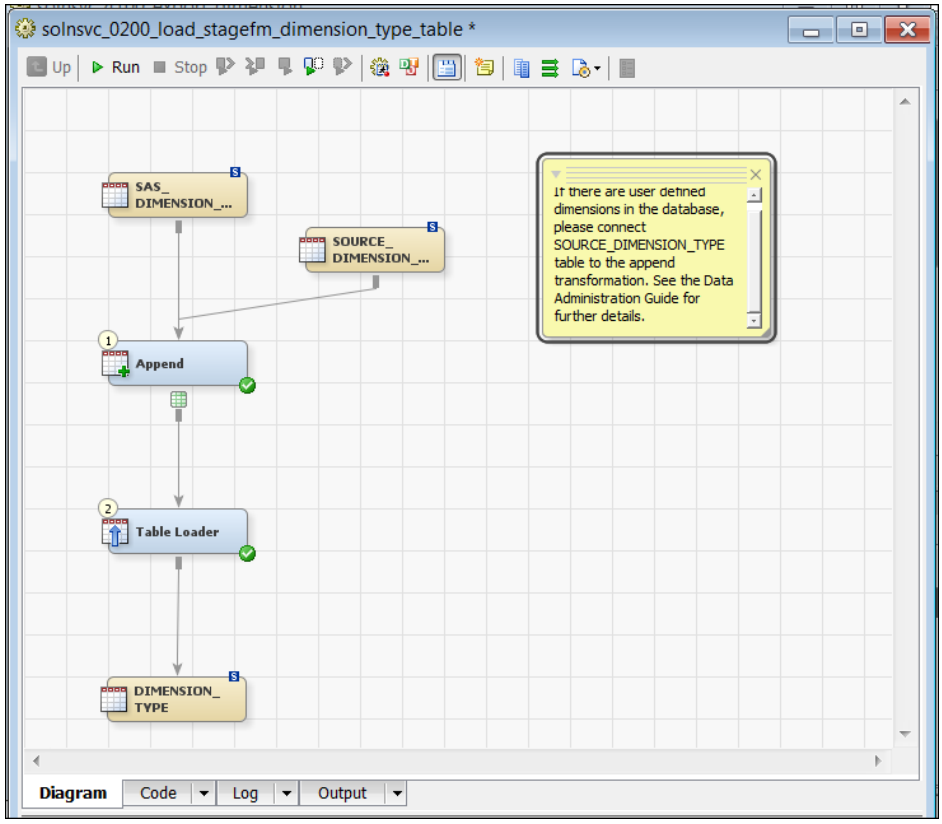
Note: It is a good practice to create and maintain a separate, appropriately named job for each set of option values. Changing the value of a job's options is possible, but likely to generate confusion.

3 Double-click **solnsvc_0200_load_stagefm_dimension_type_table** in the list of jobs. The solnsvc_0200_load_stagefm_dimension_type_table process diagram is displayed in the Job Editor window.

4 In the **Folders** tree, expand the StagedFM folder.

5 Add the SOURCE_DIMENSION_TYPE table as a second source by dragging and dropping the table onto the process diagram and connecting it to the Append transformation.

Display 10.3 *solnsvc_0200_load_stagefm_dimension_type_table Process Diagram*



6 Ensure that the columns in the SOURCE_DIMENSION_TYPE table are mapped to the output table in the **Append** transformation by completing the following steps.

   1 In the process diagram, right-click the Append transformation and select **Properties** from the pop-up menu. The Append Properties window is displayed.

   2 Select the **Mappings** tab.

   3 Click the **Map all columns** icon.

   4 Click **OK** to save your changes and close the Append Properties window.

7 Select **File** ▸ **Save**.

8 In the Job Editor window, click **Run**.

9 When the job displays as completed in the Status column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

10 Verify that all of the rows of data that the solnsvc_0200_load_dimension_type_table job placed in the SOURCE_DIMENSION_TYPE table are now in the DIMENSION_TYPE table.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

Loading New Dimension Types into the Data Mart

The solnsvc_2000_load_dimension_types job loads new dimension types into the Data Mart.

To load new dimension types into the Data Mart, complete the following steps:

1. In SAS Data Integration Studio, select the **Folders** tab.
2. In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.
3. Make a copy of the solnsvc_0200_load_dimension_types job.

   Note: It is a good practice to create and maintain a separate, appropriately named job for each set of option values. Changing the value of a job's options is possible, but likely to generate confusion.
4. Double-click **solnsvc_0200_load_dimension_types** in the list of jobs.
5. In the Job Editor window, click **Run**.
6. When the job displays as completed in the Status column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management : System Administration Guide*.

Creating Dimensions in a New Dimension Type

The methods that you use to create a dimension in a new dimension type are the same as the methods that you use to create new dimension in existing dimension types and predefined dimension types.

For information about creating dimensions, see Chapter 6, "Creating a Dimension," on page 25.

Loading Members and Hierarchies into a Dimension That Belongs to a New Dimension Type

The procedure that you use to load members and hierarchies into a dimension is the same as the procedure that you use to load members and hierarchies into new dimension types and predefined dimension types.

For information about loading members and hierarchies into a Dimension, see Chapter 7, “Loading Members and Hierarchies into a Dimension,” on page 33.

11

Creating a Stored Process

Overview

This chapter describes the following topics:

- stored processes
- creating a stored process
- editing a stored process
- registering a stored process

.

About Stored Processes

You can make any SAS Data Integration Studio job available as a *stored process* that users can run from the SAS Portal. This feature enables the ability to run the code to a larger set of users, which might be appropriate in some cases.

Before you create a stored process from a SAS Data Integration Studio job, ensure that you have made all of the appropriate modifications to the job. These modifications include specifying the appropriate values for any job options.

Creating a Stored Process

To create a stored process from a SAS Data Integration Studio job, complete the following steps:

1. In SAS Data Integration Studio, select the **Folders** tab.
2. In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.
3. Double-click the job in the list of jobs that you want to make available as a stored process.
4. In the Job Editor window, select the **Code** tab to display the job's code.
5. Select **File** ▸ **Save to File** ▸ **Local**.
6. In the Save File window, specify the target location and name for the file in which to save the job code, and click **Save**.

   Save the file in a location such as the following: ***SAS-config-dir*** **\Lev1\SASApp\SASEnvironmentData\FinancialManagement\UserDefined**.

   Note: Create the **UserDefined** directory if it does not already exist.

Note: For information about additional methods that you can use to create a stored process, see the *SAS Data Integration Studio User's Guide* (see "Related Documentation" on page 6).

Editing a Stored Process

To edit a stored process, complete the following steps:

1. Using a text editor, open the saved file.
2. At the beginning of the file, add the following statement:

   ```
   %rptinit;
   ```

3. At the end of the file, add the following statements:

   ```
   %include
   sasautos(etlstatus.sas);
   %stpend;
   ```

   Note: The %INCLUDE statement creates a job status report, which displays the status of jobs that are executed in SAS Data Integration Studio.

4. Save your changes and close the file.

Registering a Stored Process

Register the stored process in SAS Management Console. For information about registering a stored process in SAS Management Console, see "Working with Stored Processes" in the *SAS Financial Management: Customization Guide.* (see "Related Documentation" on page 6).

12

Loading Exchange Rates into a SAS Financial Management Exchange Rate Set

Overview

This chapter describes the following topics:

- exchange rates
- loading exchange rates into staging tables
- loading exchange rates from the staging area to the Data Mart
- exporting exchange rates

About Exchange Rates

Exchange Rate Types

Every currency exchange rate must belong to one of the predefined exchange rate types in the CURRENCY_EXCH_RATE_TYPE table.

The exchange rate types in the CURRENCY_EXCH_RATE_TYPE table are divided into two groups:

- *Complex exchange rate types*—Exchange rates that vary with time period and with the members of at least one other dimension type. For example, Account or Organization belong to a complex exchange rate type. The Historic and Derived exchange rate types are complex exchange rate types.
- *Simple exchange rate types*—Exchange rates that vary with time period but do not vary with the members of any other dimension type belong to a simple exchange rate type. All exchange rates types except for Historic and Derived are simple exchange rate types.

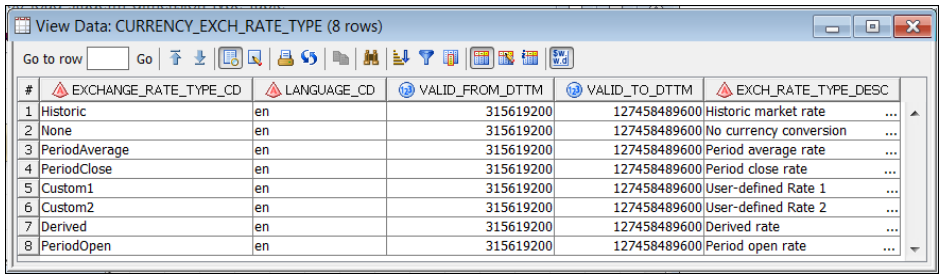
To view the rows of data in the CURRENCY_EXCH_RATE_TYPE table, complete the following steps:

1 In SAS Data Integration Studio, select the **Folders** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.

3 To view the rows of data in the CURRENCY_EXCHANGE_RATE_TYPE table, right-click on the table in the list of tables and select **Open** from the pop-up menu.

Display 12.1 *CURRENCY_EXCH_RATE_TYPE—Rows View*


View Data: CURRENCY_EXCH_RATE_TYPE (8 rows)

Go to row Go

#	EXCHANGE_RATE_TYPE_CD	LANGUAGE_CD	VALID_FROM_DTTM	VALID_TO_DTTM	EXCH_RATE_TYPE_DESC
1	Historic	en	315619200	127458489600	Historic market rate
2	None	en	315619200	127458489600	No currency conversion
3	PeriodAverage	en	315619200	127458489600	Period average rate
4	PeriodClose	en	315619200	127458489600	Period close rate
5	Custom1	en	315619200	127458489600	User-defined Rate 1
6	Custom2	en	315619200	127458489600	User-defined Rate 2
7	Derived	en	315619200	127458489600	Derived rate
8	PeriodOpen	en	315619200	127458489600	Period open rate

4 Click **OK** to close the window.

Exchange Rate Sets

In SAS Financial Management, every exchange rate belongs to a SAS Financial Management *exchange rate set*.

You define SAS Financial Management exchange rate sets in the **Rates** workspace of SAS Financial Management Studio. In the **Rates** workspace, you can define exchange rates codes, names, and descriptions.

In the SAS Data Integration Studio staging area, every exchange rate belongs to a staging area exchange rate set. You define staging area exchange rate sets in the CURRENCY_EXCH_RATE_SET table.

Exchange Rate Sources

Each exchange rate that you load is extracted from a source. You must define codes for the sources from which you extract exchange rates and load these codes into CURRENCY_EXCH_RATE_SRC table.

Loading Exchange Rates into Staging Tables

The following staging tables are defined in the StageFM library for exchange rate data:

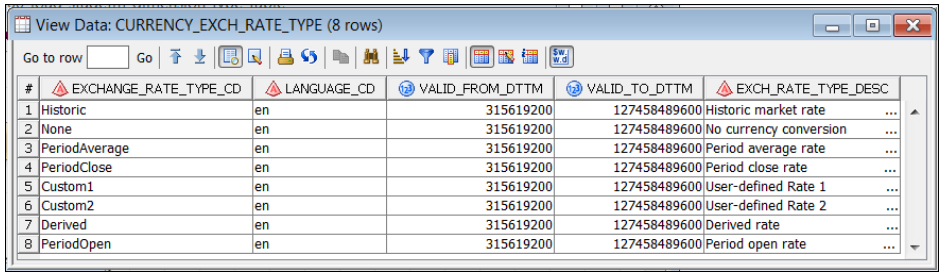
- CURRENCY_EXCH_RATE—Contains the numerical exchange rates between pairs of currencies, for different time periods and exchange rate types.
- CURRENCY_EXCH_RATE_SET—Defines the exchange rate sets that are loaded in the SAS Financial Management Data Mart.
- CURRENCY_EXCH_RATE_SRC—Defines codes that identify the sources from which you extract numerical exchange rates.
- CURRENCY_COMPLEX_EXCH_RATE—Contains the numerical exchange rates between pairs of currencies, defined by crossings of various dimension members and exchange rate types.
- CURRENCY_EXCH__RATE_TYPE—Defines the currency exchange rate type such as Derived, Historic, None, PeriodAverage, PeriodClose, and PeriodOpen.

To load the tables with data, you must write and run a job that loads the data into each table. Before you write a job to load data into a table, review the column structure of the table to ensure that the job that you write places the correct data in the correct columns.

To view the column structure of the exchange rate tables, complete the following steps:

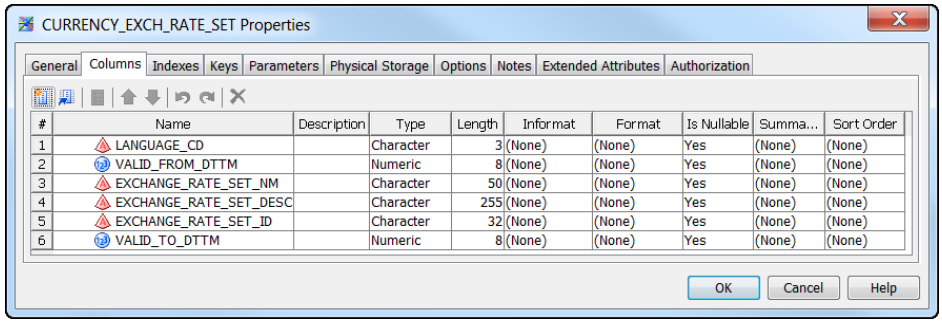
1. In the SAS Data Integration Studio, select the **Folders** tab.
2. In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.
3. Double-click the appropriate table for the exchange rates that you are loading in the list of tables. The Properties window for the table is displayed.
4. Select the **Columns** tab to view the column structure of the table.

Display 12.2 *CURRENCY_EXCH_RATE_TYPE — Rows View*



#	EXCHANGE_RATE_TYPE_CD	LANGUAGE_CD	VALID_FROM_DTTM	VALID_TO_DTTM	EXCH_RATE_TYPE_DESC
1	Historic	en	315619200	127458489600	Historic market rate
2	None	en	315619200	127458489600	No currency conversion
3	PeriodAverage	en	315619200	127458489600	Period average rate
4	PeriodClose	en	315619200	127458489600	Period close rate
5	Custom1	en	315619200	127458489600	User-defined Rate 1
6	Custom2	en	315619200	127458489600	User-defined Rate 2
7	Derived	en	315619200	127458489600	Derived rate
8	PeriodOpen	en	315619200	127458489600	Period open rate

Display 12.3 *CURRENCY_EXCH_RATE_SET Properties Window — Columns View*

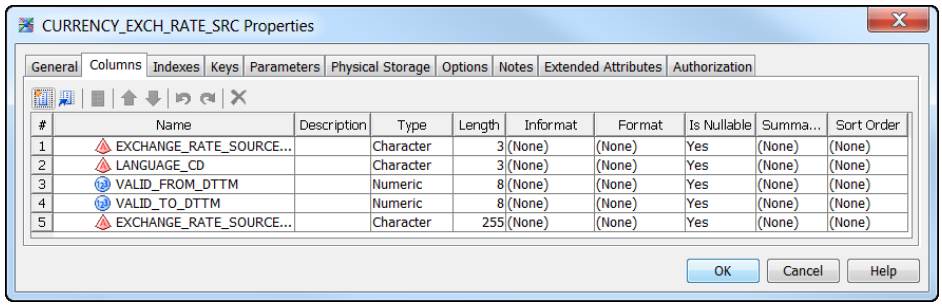


#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	LANGUAGE_CD		Character	3	(None)	(None)	Yes	(None)	(None)
2	VALID_FROM_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
3	EXCHANGE_RATE_SET_NM		Character	50	(None)	(None)	Yes	(None)	(None)
4	EXCHANGE_RATE_SET_DESC		Character	255	(None)	(None)	Yes	(None)	(None)
5	EXCHANGE_RATE_SET_ID		Character	32	(None)	(None)	Yes	(None)	(None)
6	VALID_TO_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)

Before building records for the CURRENCY_EXCH_RATE_SET table, note the following:

- The VALID_FROM_DTTM column and VALID_TO_DTTM column define the lifespan of the record.
- Ensure that you maintain a one-to-one correlation between staging area exchange rate sets and SAS Financial Management exchange rate sets. In addition, ensure that you coordinate the codes, names, and descriptions of the corresponding pairs.
- You must load the definitions of the exchange rate sets into the staging area before you load exchange rates that belong to those exchange rate sets into the staging area.

Display 12.4 *CURRENCY_EXCH_RATE_SRC Properties Window — Columns View*



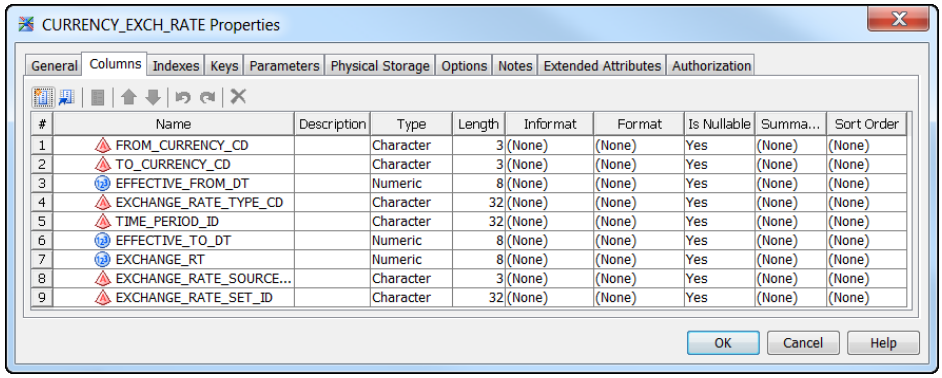
#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	EXCHANGE_RATE_SOURCE...		Character	3	(None)	(None)	Yes	(None)	(None)
2	LANGUAGE_CD		Character	3	(None)	(None)	Yes	(None)	(None)
3	VALID_FROM_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
4	VALID_TO_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
5	EXCHANGE_RATE_SOURCE...		Character	255	(None)	(None)	Yes	(None)	(None)

Before building records for the CURRENCY_EXCH_RATE_SRC table, note the following:

- If exchange rates are extracted from a single source, or if you are not interested in tracking source information for exchange rates, the CURRENCY_EXCH_RATE_SRC table can contain a single record.

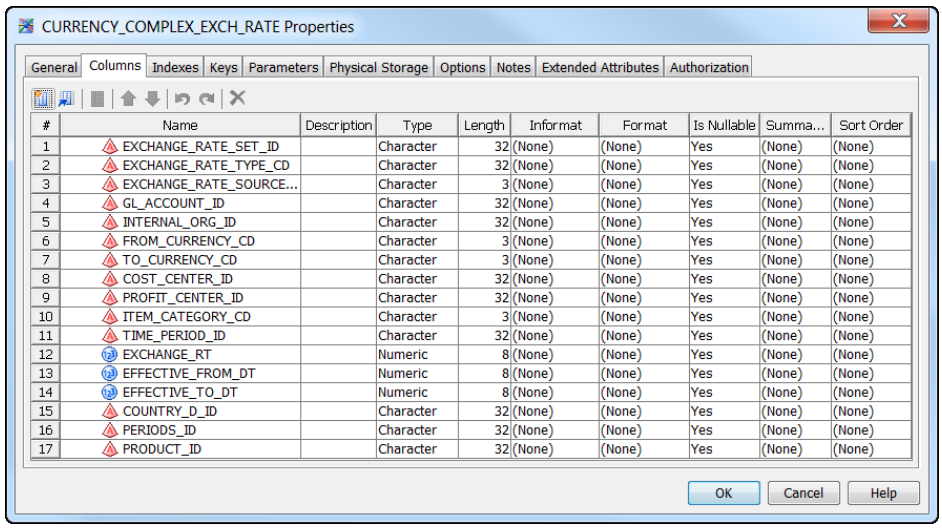
- You must load the definitions of your exchange rate sources into the staging area before you load exchange rates that belong to those exchange rate sources into the staging area.

Display 12.5 *CURRENCY_EXCH_RATE Properties Window — Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	FROM_CURRENCY_CD		Character	3	(None)	(None)	Yes	(None)	(None)
2	TO_CURRENCY_CD		Character	3	(None)	(None)	Yes	(None)	(None)
3	EFFECTIVE_FROM_DT		Numeric	8	(None)	(None)	Yes	(None)	(None)
4	EXCHANGE_RATE_TYPE_CD		Character	32	(None)	(None)	Yes	(None)	(None)
5	TIME_PERIOD_ID		Character	32	(None)	(None)	Yes	(None)	(None)
6	EFFECTIVE_TO_DT		Numeric	8	(None)	(None)	Yes	(None)	(None)
7	EXCHANGE_RT		Numeric	8	(None)	(None)	Yes	(None)	(None)
8	EXCHANGE_RATE_SOURCE...		Character	3	(None)	(None)	Yes	(None)	(None)
9	EXCHANGE_RATE_SET_ID		Character	32	(None)	(None)	Yes	(None)	(None)

Display 12.6 *CURRENCY_COMPLEX_EXCH_RATE Properties Window — Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	EXCHANGE_RATE_SET_ID		Character	32	(None)	(None)	Yes	(None)	(None)
2	EXCHANGE_RATE_TYPE_CD		Character	32	(None)	(None)	Yes	(None)	(None)
3	EXCHANGE_RATE_SOURCE...		Character	3	(None)	(None)	Yes	(None)	(None)
4	GL_ACCOUNT_ID		Character	32	(None)	(None)	Yes	(None)	(None)
5	INTERNAL_ORG_ID		Character	32	(None)	(None)	Yes	(None)	(None)
6	FROM_CURRENCY_CD		Character	3	(None)	(None)	Yes	(None)	(None)
7	TO_CURRENCY_CD		Character	3	(None)	(None)	Yes	(None)	(None)
8	COST_CENTER_ID		Character	32	(None)	(None)	Yes	(None)	(None)
9	PROFIT_CENTER_ID		Character	32	(None)	(None)	Yes	(None)	(None)
10	ITEM_CATEGORY_CD		Character	3	(None)	(None)	Yes	(None)	(None)
11	TIME_PERIOD_ID		Character	32	(None)	(None)	Yes	(None)	(None)
12	EXCHANGE_RT		Numeric	8	(None)	(None)	Yes	(None)	(None)
13	EFFECTIVE_FROM_DT		Numeric	8	(None)	(None)	Yes	(None)	(None)
14	EFFECTIVE_TO_DT		Numeric	8	(None)	(None)	Yes	(None)	(None)
15	COUNTRY_D_ID		Character	32	(None)	(None)	Yes	(None)	(None)
16	PERIODS_ID		Character	32	(None)	(None)	Yes	(None)	(None)
17	PRODUCT_ID		Character	32	(None)	(None)	Yes	(None)	(None)

Before building records for the two exchange rate staging tables, note the following:

Column	Description
TO_CURRENCY_CD	Code of the base currency of the target exchange rate set. This code should be the same for all records that belong to a given staging area exchange rate set, as indicated in the Exchange Rate Set ID column. When you load data into an exchange rate set in the Data Mart, records whose To Currency Code does not match the base currency of the target exchange rate set are ignored.
FROM_CURRENCY_CD	Code of the other currency that is involved in the exchange rate.

Column	Description
EXCHANGE_RATE_TYPE_CD	In the CURRENCY_EXCH_RATE table, must be one of the simple exchange rate type codes in the SAS_CURRENCY_EXCH_RATE_TYPE table. In the CURRENCY_COMPLEX_EXCH_RATE table, must be one of the complex exchange rate type codes in the SAS_CURRENCY_EXCH_RATE_TYPE table. For more information about exchange rate types, see "Exchange Rate Types" on page 80.
EXCHANGE_RATE_SET_ID	Indicates which staging area exchange rate set the exchange rate belongs to. It must be one of the values in the CURRENCY_EXCH_RATE_SET table. or more information about exchange rate sets, see "Exchange Rate Sets" on page 80.
EXCHANGE_RATE_SOURCE_CD	Indicates where the exchange rate was extracted from. It must be one of the values in the CURRENCY_EXCH_RATE_SRC table. For more information about exchange rate sources, see "Exchange Rate Sources" on page 81.
EXCHANGE_RT	Numeric exchange rate. The numeric exchange rate must be the number by which you multiply a value expressed in the From Currency to yield the equivalent value expressed in the To Currency. For example, if the From Currency is U.S. dollars and the To Currency is Japanese yen, then the numeric exchange rate is in the approximately 100. However, if the From Currency is Japanese yen and the To Currency is U.S. dollars, then the numeric exchange rate is in the approximately 0.01.
TIME_PERIOD_ID	Code of the time period that the exchange rate applies to. The CURRENCY_COMPLEX_EXCH_RATE table has columns for member codes from other dimension types that the exchange rates depend on.
EFFECTIVE_FROM_DT	Must contain a distinct date for each time period. For example, you can use the first day of each time period.
EFFECTIVE_TO_DT	Not used. Therefore, you can leave this field blank.

5 Click **OK** to close the Properties window.

To load exchange rate data into the exchange rate staging tables, complete the following steps:

1 Create a SAS Data Integration Studio job that loads data from its source into the appropriate table in the StageFM library.

2 Click **OK** to save the job.

3 Run the job and validate that the table loaded successfully.

Note: For detailed information about creating and working with SAS Data Integration Studio jobs to extract data from a source table and load it into a target table, see the *SAS Data Integration User's Guide*.

Loading Exchange Rates into the Data Mart

About Loading Exchange Rates into the Data Mart

When loading exchange rates, note the following:

- You can use SAS Data Integration Studio to load all exchange rates, both simple and complex.
- You can also load all exchange rates by using the Load Exchange Rates wizard in the Rates workspace of SAS Financial Management Studio.
- The Rates workspace of SAS Financial Management Studio also enables you to enter exchange rates manually. However, given the volume of data that is involved and the importance of avoiding errors, ensure that you load exchange rates from a reliable source by using either SAS Data Integration Studio or SAS Financial Management Studio.

You load exchange rates from the staging area into a SAS Financial Management exchange rate set in the Data Mart by using the following methods:

- the Load Exchange Rates wizard in SAS Financial Management Studio
- a SAS Data Integration Studio job
- a SAS macro

When choosing a method of loading exchange rates from the staging area into the Data Mart, note the following:

- Each time you run the SAS Data Integration Studio job, it loads exchange rates for only one combination of an exchange rate type and a time period.
- The Load Exchange Rates wizard and the SAS macro can handle many combinations of exchange rate types and time periods in a single run. Because there is substantial overhead associated with each run, as the number of combinations of exchange rate types and time periods increases, using the wizard and the SAS macro becomes increasingly advantageous.
- When you load exchange rates into an exchange rate set, the exchange rates in the target exchange rate set for the specified time periods and exchange rate types are deleted before the new exchange rates are loaded.

Loading Exchange Rates into the Data Mart by Using the Load Exchange Rates Wizard

To load exchange rates by using the Load Exchange Rates wizard, complete the following steps:

1 In SAS Financial Management Studio, select the **Rates** workspace.

2 In the Exchange Rate Sets view, select the exchange rate set into which you want to load exchange rates.

3 Select **Load Exchange Rates**. The Load Exchange Rates wizard launches.

4 Work through the wizard, consulting the online Help as necessary.

Loading Exchange Rates into the Data Mart by Using a Job

The fm_1300_exchange_rates job loads exchange rates into the data mart.

To load exchange rates by using a SAS Data Integration Studio job, complete the following steps:

1 In SAS Data Integration Studio, select the **Folder** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.

3 Make a copy of the fm_1300_exchange_rates job.

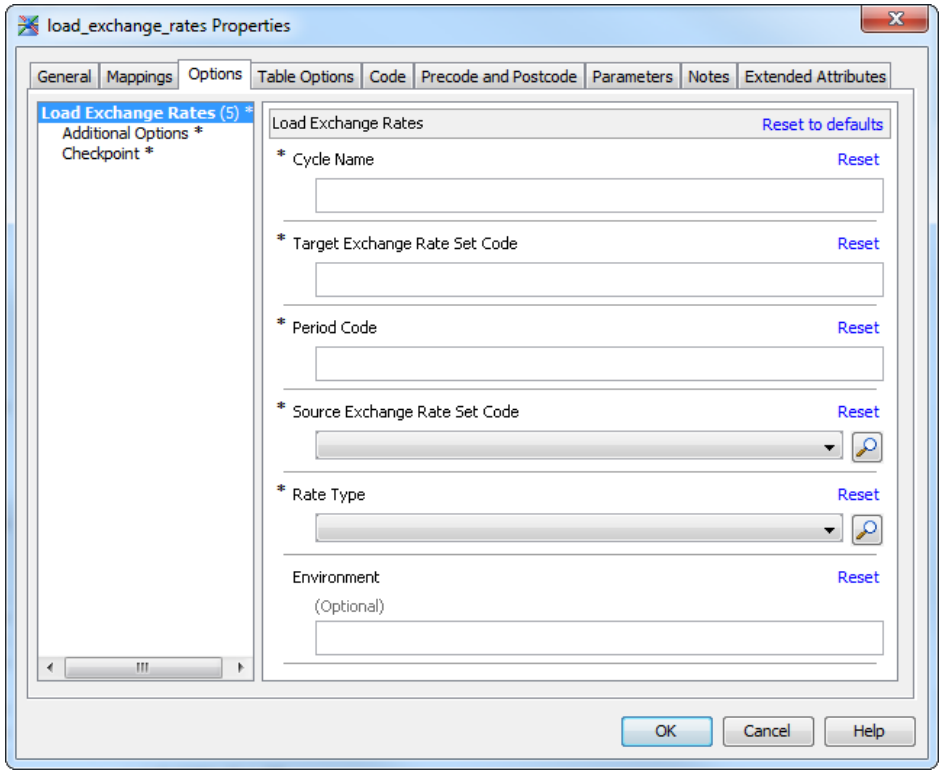
Note: It is a good practice to create and maintain a separate, appropriately named job for each set of option values. Changing the value of a job's options is possible, but likely to generate confusion.

4 Double-click the **fm_1300_exchange_rates** in the list of jobs.

5 In the Job Editor window, right-click the transformation and select **Properties** from the pop-up menu.

6 Select the **Options** tab.

Display 12.7 *load_exchange_rates Properties Window — Options View*



7 Enter values for the following options:

Option	Description
Cycle Name	Name of the cycle to which the target exchange rate set belongs.
Target Exchange Rate Set Code	Code of the target exchange rate set.
Period Code	Code of the time period for which you are loading exchange rates.
Source Exchange Rate Set Code	Code of the source exchange rate set in the staging area. Use the drop-down list to select a valid code.
Rate Type	Exchange rate type for which you are loading exchange rates. Use the drop-down list to select a valid exchange rate type. If you select a simple exchange rate type, then the job gets the exchange rates from the CURRENCY_EXCH_RATE table. If you select a complex exchange rate type, —then the job gets the exchange rates from the CURRENCY_COMPLEX_ EXCH_RATE table.
Environment	(Optional) Any environment that is defined in your sas-environment.xml file. If you leave this field empty, then the environment “default” is used.

8 Click **OK** to save your changes and close the Properties window.

9 Select **File** ▸ **Save**.

10 In the Job Editor window, click **Run**.

11 When the job displays as completed in the Status column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

Loading Exchange Rates into the Data Mart by Using a SAS Macro

To load exchange rates by using a SAS macro, complete the following steps:

1 Create a SAS data set that specifies the combinations of exchange rate types and time periods for which you want to load exchange rates.

2 Run the etlxrteb.sas macro file.

When using the etlxtreb.sas macro to load exchange rates into the Data Mart, note the following:

- Detailed instructions on using the etlxrteb.sas macro are inside the macro file, which is located on the data tier server.
- On a Windows server, the etlxrteb.sas macro file is at the following location: `SASROOT\finance\sasmacro`
- On a UNIX server, the etlxrteb.sas macro file is at the following location: `SASROOT/sasautos`

Note: To run the macro, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide* (see "Related Documentation" on page 6.

Exporting Exchange Rates

You can export driver rates by using a SAS Financial Management Studio wizard or by using a SAS Data Integration Studio job.

Exporting Exchange Rates by Using the Export Exchange Rates Wizard

To export exchange rates by using the Export Exchange Rates wizard, complete the following steps:

1 In SAS Financial Management Studio, open the cycle from which you want to export data.

2 In the Rates workspace, select the **Exchange Rates** tab.

3 In the list of exchange rate sets, select the exchange rate set containing the exchange rates that you want to export.

4 Select **Export exchange rates for this exchange rate set** to launch the Export Driver Rates wizard.

5 Work through the wizard, referring to the online Help for the individual wizard pages if necessary.

Exporting Exchange Rates by Using a Job

The fm_1310_export_exchange_rates job exports exchange rates.

To export exchange rates by using the fm_1310_export_exchange_rates job, complete the following steps:

1 In SAS Data Integration Studio, select the **Folder** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.

3 Make a copy of the fm_1310_export_exchange_rates job.

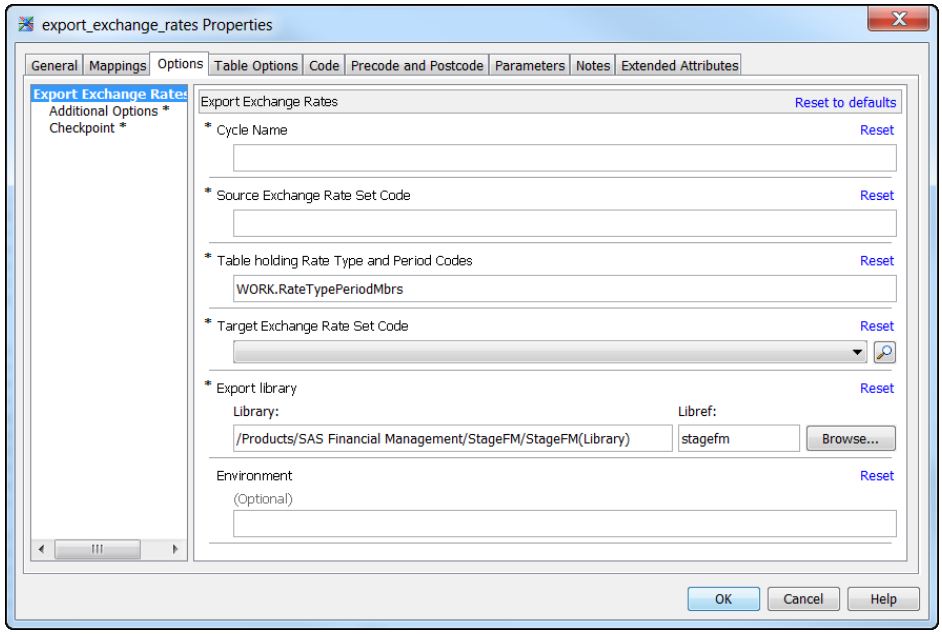
Note: It is a good practice to create and maintain a separate, appropriately named job for each set of option values. Changing the value of a job's options is possible, but likely to generate confusion.

4 Double-click **fm_1310_export_exchange_rates** in the list of jobs.

5 In the Job Editor window, select the **export_exchange_rates** transformation and select **Properties** from the pop-up menu. The export_exchange_rates Properties window is displayed.

6 Select the **Options** tab.

Display 12.8 *export_exchange_rates Properties Window — Options View*



7 Enter values for the following options:

Option	Description
Cycle Name	Add description.
Source Exchange Rate Set Code	Add description.
Table holding Rate Type and Period Codes	Add description.
Target Exchange Rate Code	Add description.
Export Library	Add description.
Environment	(Optional)

8 Click **OK** to save your changes and close the Properties window.

9 Select **File** ▸ **Save**.

10 In the Job Editor window, click **Run**.

11 When the job displays as completed in the Status column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administrator's Guide* (see "Related Documentation" on page 6).

13

Loading Driver Rates into a SAS Financial Management Driver Rate Set

Overview

This chapter describes the following topics:

- about driver rates
- loading driver rates into their staging table
- loading driver rates from the staging area to the Data Mart
- exporting driver rates

About Driver Rates

Driver Rate Types

Every driver rate that you load must belong to a driver rate type that you define in the Rates workspace of SAS Financial Management Studio.

Like the relationship between exchange rates and exchange rate types, driver rates must belong to a driver rate type.

The key difference between exchange rate types and driver rate types is that exchange rate types are predefined in a SAS Data Integration Studio table and driver rate types are not. Therefore, you must load driver rate types into the MISC_RATE_TYPE table into the staging area before you can load driver rates that belong to those driver rate types into the staging area.

To define a driver rate type, complete the following steps:

1 In SAS Financial Management Studio, select the Rates workspace.

2 Select **Tools** ▸ **Driver Rate Type**.

3 Click **New Driver Rate Type**.

The staging table defined in the StageFM library for driver rate type is the MISC_RATE_TYPE table.

To load the table with data, you must write and run a job that loads data into the table. Before you write a job that loads data into the MISC_RATE_TYPE, review the column structure of the table to ensure that the job that you write places the correct data in the correct columns.

To view the column structure of the MISC_RATE_TYPE table, complete the following steps:

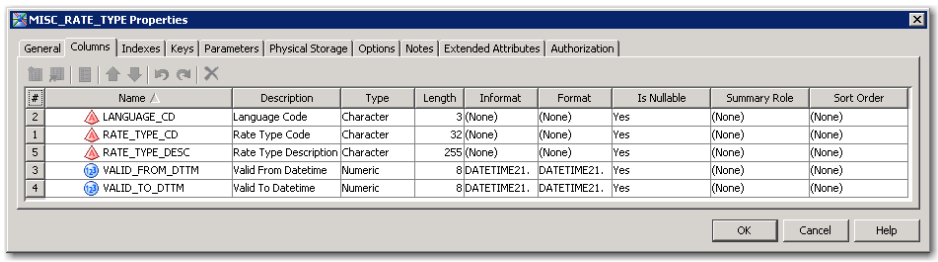
1 In SAS Data Integration Studio, select the **Folders** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.

3 Double-click **MISC_RATE_TYPE** in the list of tables. The MISC_RATE_TYPE Properties window is displayed.

4 Click the **Columns** tab to view the column structure of the table.

Display 13.1 *MISC_RATE_TYPE Properties Window — Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summary Role	Sort Order
2	LANGUAGE_CD	Language Code	Character	3	(None)	(None)	Yes	(None)	(None)
1	RATE_TYPE_CD	Rate Type Code	Character	32	(None)	(None)	Yes	(None)	(None)
5	RATE_TYPE_DESC	Rate Type Description	Character	255	(None)	(None)	Yes	(None)	(None)
3	VALID_FROM_DTTM	Valid From Datetime	Numeric	8	DATETIME21.	DATETIME21.	Yes	(None)	(None)
4	VALID_TO_DTTM	Valid To Datetime	Numeric	8	DATETIME21.	DATETIME21.	Yes	(None)	(None)

The MISC_RATE_TYPE table contains the following columns:

Column	Description
LANGUAGE_CD	Language code that is used in staging tables. Typically, the language code is one of the two-character codes in the ISO0639_LANGUAGE_CD column of the SAS_LANGUAGE_ISO0639 table. One exception is if you need two or more records that represent variants of the same language. For example, if you have a record for French as used in France and another record for French as used in Canada, then you might use language codes **`frf`** and **`frc`**, respectively. **Note:** Do not use the same language code in two records.
RATE_TYPE_CD	Unique code for a type of rate. This column is used for data validation in jobs.
RATE_TYPE_DESC	Names that describe the types of rates.
VALID_FROM_DTTM	Moment that begins the time period during which a row of data is valid..
VALID_TO_DTTM	Moment that ends the time period during which a row of data is valid..

5 Click **OK** to close the Properties window.

To load data into MISC_RATE_TYPE table, complete the following steps:

1 Create a SAS Data Integration Studio job that loads data from its source into the appropriate table in the StageFM library.

2 Click **OK** to save the job.

3 Run the job and validate that the table loaded successfully.

Note: For detailed information about creating and working with SAS Data Integration Studio jobs to extract data from a source table and load it into a target table, see the *SAS Data Integration User's Guide*.

Driver Rate Sets

In SAS Financial Management, every driver rate belongs to a driver rate set. You define SAS Financial Management driver rate sets in the **Rates** workspace of SAS Financial Management Studio. In the **Rates** workspace, you define codes, names, and descriptions.

In the staging area, every driver rate belongs to a staging area driver rate set. The staging table defined in the StageFM library for driver rate sets is the MISC_RATE_SET table.

To load the table with data, you must write and run a job that loads data into the table. Before you write a job that loads data into the MISC_RATE_SET, review the column structure of the table to ensure that the job that you write places the correct data in the correct columns.

To view the column structure of the MISC_RATE_SET table, complete the following steps:

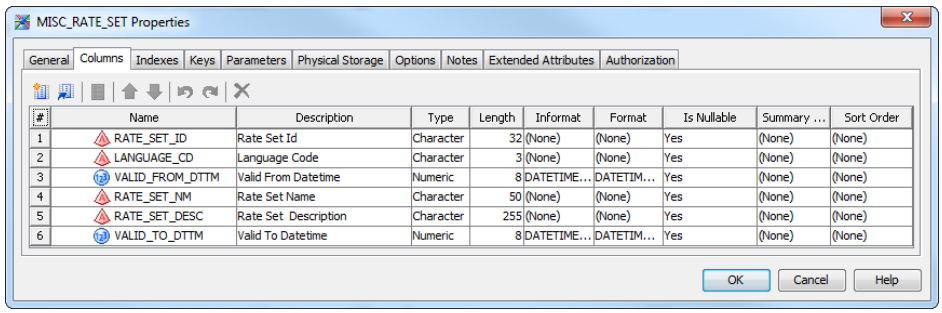
1 In SAS Data Integration Studio, select the **Folders** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.

3 Double-click **MISC_RATE_SET** in the list of tables. The MISC_RATE_SET Properties window is displayed.

4 Click the **Columns** tab to view the column structure of the table.

Display 13.2 *MISC_RATE_SET Properties Window — Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summary ...	Sort Order
1	RATE_SET_ID	Rate Set Id	Character	32	(None)	(None)	Yes	(None)	(None)
2	LANGUAGE_CD	Language Code	Character	3	(None)	(None)	Yes	(None)	(None)
3	VALID_FROM_DTTM	Valid From Datetime	Numeric	8	DATETIME...	DATETIM...	Yes	(None)	(None)
4	RATE_SET_NM	Rate Set Name	Character	50	(None)	(None)	Yes	(None)	(None)
5	RATE_SET_DESC	Rate Set Description	Character	255	(None)	(None)	Yes	(None)	(None)
6	VALID_TO_DTTM	Valid To Datetime	Numeric	8	DATETIME...	DATETIM...	Yes	(None)	(None)

Before building records for the MISC_RATE_SET table, note the following:

- The VALID_FROM_DTTM column and the VALID_TO_DTTM column define the lifespan of the record.
- Ensure that you maintain a one-to-one correspondence between staging area driver rate sets and SAS Financial Management driver rate sets. In addition, ensure that you coordinate the codes, names, and descriptions of the corresponding pairs.
- You must load the definitions of the driver rate sets into the staging area before you load driver rates that belong to those driver rate sets into the staging area.

5 Click **OK** to close the Properties window.

To load data into MISC_RATE_SET table, complete the following steps:

1 Create a SAS Data Integration Studio job that loads data from its source into the appropriate table in the StageFM library.

2 Click **OK** to save the job.

3 Run the job and validate that the table loaded successfully.

Note: For detailed information about creating and working with SAS Data Integration Studio jobs to extract data from a source table and load it into a target table, see the *SAS Data Integration User's Guide*.

Loading Driver Rates into the Staging Table

The staging table defined in the StageFM library for driver rates is the MISC_RATE table.

To load the table with data, you must write and run a job that loads data into the table. Before you write a job that loads data into the MISC-RATE, review the column structure of the table to ensure that the job that you write places the correct data in the correct columns.

To view the column structure of the MISC_RATE_SET table, complete the following steps:

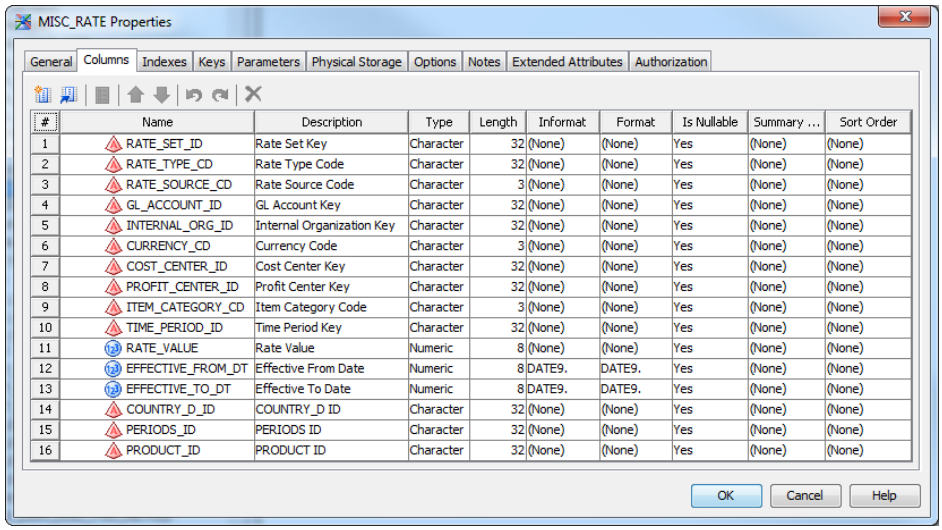
1 In the SAS Data Integration Studio, select the **Folders** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.

3 Double-click MISC_RATE in the list of tables. The table MISC_RATE Properties window is displayed.

4 Select the **Columns** tab to view the column structure of the table.

Display 13.3 *MISC_RATE Properties Window — Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summary ...	Sort Order
1	RATE_SET_ID	Rate Set Key	Character	32	(None)	(None)	Yes	(None)	(None)
2	RATE_TYPE_CD	Rate Type Code	Character	32	(None)	(None)	Yes	(None)	(None)
3	RATE_SOURCE_CD	Rate Source Code	Character	3	(None)	(None)	Yes	(None)	(None)
4	GL_ACCOUNT_ID	GL Account Key	Character	32	(None)	(None)	Yes	(None)	(None)
5	INTERNAL_ORG_ID	Internal Organization Key	Character	32	(None)	(None)	Yes	(None)	(None)
6	CURRENCY_CD	Currency Code	Character	3	(None)	(None)	Yes	(None)	(None)
7	COST_CENTER_ID	Cost Center Key	Character	32	(None)	(None)	Yes	(None)	(None)
8	PROFIT_CENTER_ID	Profit Center Key	Character	32	(None)	(None)	Yes	(None)	(None)
9	ITEM_CATEGORY_CD	Item Category Code	Character	3	(None)	(None)	Yes	(None)	(None)
10	TIME_PERIOD_ID	Time Period Key	Character	32	(None)	(None)	Yes	(None)	(None)
11	RATE_VALUE	Rate Value	Numeric	8	(None)	(None)	Yes	(None)	(None)
12	EFFECTIVE_FROM_DT	Effective From Date	Numeric	8	DATE9.	DATE9.	Yes	(None)	(None)
13	EFFECTIVE_TO_DT	Effective To Date	Numeric	8	DATE9.	DATE9.	Yes	(None)	(None)
14	COUNTRY_D_ID	COUNTRY_D ID	Character	32	(None)	(None)	Yes	(None)	(None)
15	PERIODS_ID	PERIODS ID	Character	32	(None)	(None)	Yes	(None)	(None)
16	PRODUCT_ID	PRODUCT ID	Character	32	(None)	(None)	Yes	(None)	(None)

Before building records for the MISC_RATE staging table, note the following:

Column	Description
RATE_TYPE_CD	Code of a driver rate type that you define in the MISC_RATE_TYPE table. For information about driver rate types, see “Driver Rate Types” on page 92.

Column	Description
RATE_SET_KEY	Indicates to which staging area driver rate set the driver rate belongs. The value for this column must be one of the values in the MISC_RATE_SET table. For information about driver rate sets, see "Driver Rate Sets" on page 93.
RATE_SOURCE_CD	Indicates from what source the driver rate was extracted from. This column is not validated and can be left blank.
RATE_VALUE	Driver rate that you specify in a numeric value.
GL_ACCOUNT_ID (and other Key columns)	Member codes that the driver rate depends on. In each record, at least one of these columns must contain a member code.
EFFECTIVE_FROM_DT	Effective To Date—Not used. Therefore, you can leave this field blank.
EFFECTIVE_TO_DT	Not used. Therefore, you can leave this field blank.

5 Click **OK** to close the Properties window.

To load cell visibility rule data into the MISC_RATE table, complete the following steps:

1 Create a SAS Data Integration Studio job that loads data from its source into the appropriate table in the StageFM library.

2 Click **OK** to save the job.

3 Run the job and validate that the table loaded successfully.

Note: For detailed information about creating and working with SAS Data Integration Studio jobs to extract data from a source table and load it into a target table, see the *SAS Data Integration User's Guide*.

Loading Driver Rates into the Data Mart

About Loading Driver Rates into the Data Mart

You can load driver rates from the staging area into a SAS Financial Management driver rate set in the Data Mart by using the following methods:

- The Load Driver Rates wizard in SAS Financial Management Studio.
- A SAS Data Integration Studio job.
- A SAS macro.

When choosing a method of loading driver rates from the staging area into the Data Mart, note the following:

- Each time you run the SAS Data Integration Studio job, it loads driver rates for only one driver rate type.
- The Load Rates wizard and the SAS macro can handle many driver rate types in a single run. Because there is substantial overhead associated with each run, as the number of driver rate types increases, the wizard and the SAS macro become increasingly advantageous.
- When you load driver rates into a driver rate set, the driver rates in the target driver rate set and for the specified driver rate types are deleted before the new driver rates are loaded.

Loading Driver Rates into the Data Mart by Using the Load Driver Rates Wizard

To load driver rates using the Load Driver Rates wizard, complete the following steps:

1. In SAS Financial Management Studio, select the Rates workspace.
2. In the Driver Rate Sets view, select the driver rate set into which you want to load driver rates.
3. Select **Load Driver Rates**. The Load Driver Rates wizard launches.
4. Work through the wizard, consulting the online Help as necessary.

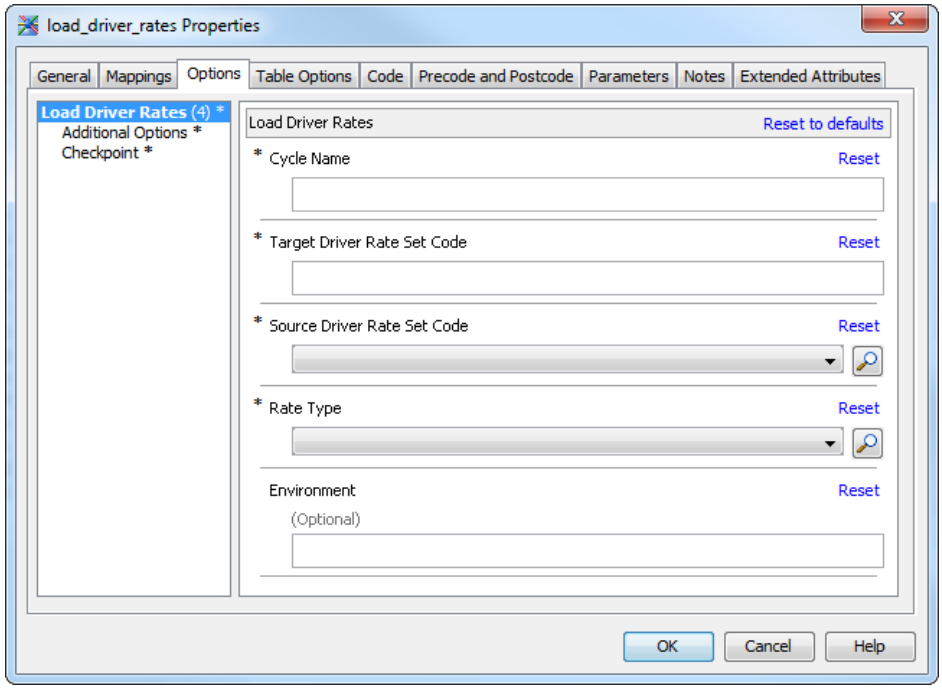
Loading Driver Rates into the Data Mart by Using a Job

To load driver rates by using a SAS Data Integration Studio job, complete the following steps:

1. In SAS Data Integration Studio, select the **Folder** tab.
2. In the Folders tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.
3. Make a copy of the fm_1500_load_driver_rates job.

   Note: It is a good practice to create and maintain a separate, appropriately named job for each set of option values. Changing the value of a job's options is possible, but likely to generate confusion.
4. Double-click fm_1500_load_driver_rates in the list of jobs.
5. In the Job Editor window, right-click the load_driver_rates transformation and select **Properties** from the pop-up menu.
6. Select the **Options** tab.

Display 13.4 load_driver_rates Properties Window — Options View



7 Enter values for the following options:

Option	Description
Cycle Name	Name of the cycle to which the target driver rate set belongs.
Target Driver Rate Set Code	Code of the target driver rate set.
Source Driver Rate Set Code	Code of the source driver rate set in the SAS Financial Management staging area. Use the drop-down list to select a valid code.
Rate Type	Driver rate type for which you are loading driver rates. Use the drop-down list to select a valid driver rate type.
Environment	(Optional) Any environment that is defined in your sas-environment.xml file. If you leave this field empty, then the environment “default” is used.

8 Click **OK** to save your changes and close the Properties window.

9 Select **File** ▸ **Save**.

10 In the Job Editor window, click **Run**.

11 When the job displays as completed in the Status column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

Loading Driver Rates into the Data Mart by Using a SAS Macro

To load driver rates by using a SAS macro, complete the following steps:

1. Create a SAS data set that specifies the driver rate types for which you want to load driver rates.
2. Run the etldrteb.sas macro file.

When using the etldrteb.sas macro to load driver rates into the Data Mart, note the following:

- Detailed instructions on using the etlxrteb.sas macro are inside the macro file, which is on the data tier server.
- On a Windows server, the etldrteb.sas macro file is at the following location: `!SASROOT\finance\sasmacro`
- On a UNIX server, the etldrteb.sas macro file is at the following location: `!SASROOT/sasautos`

Note: To run the macro, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

Exporting Driver Rates

You can export driver rates by using a SAS Financial Management Studio wizard or by using a SAS Data Integration Studio job.

Exporting Driver Rates by Using the Export Driver Rates Wizard

To export driver rates by using the Export Driver Rates wizard, complete the following steps:

1. In SAS Financial Management Studio, open the cycle from which you want to export data.
2. In the Rates workspace, select the **Driver Rates** tab.
3. In the list of driver rate sets, select the driver rate set containing the driver rates that you want to export.
4. Select **Export driver rates for this driver rate set** to launch the Export Driver Rates wizard.
5. Work through the wizard, referring to the online Help for the individual wizard pages if necessary.

Exporting Driver Rates by Using a Job

You can use the fm_1510_export_driver_rates job to export driver rates.

To export driver rates by using the fm_1510_export_driver_rates job, complete the following steps:

1 In SAS Data Integration Studio, select the **Folder** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.

3 Make a copy of the fm_1510_export_driver_rates job.

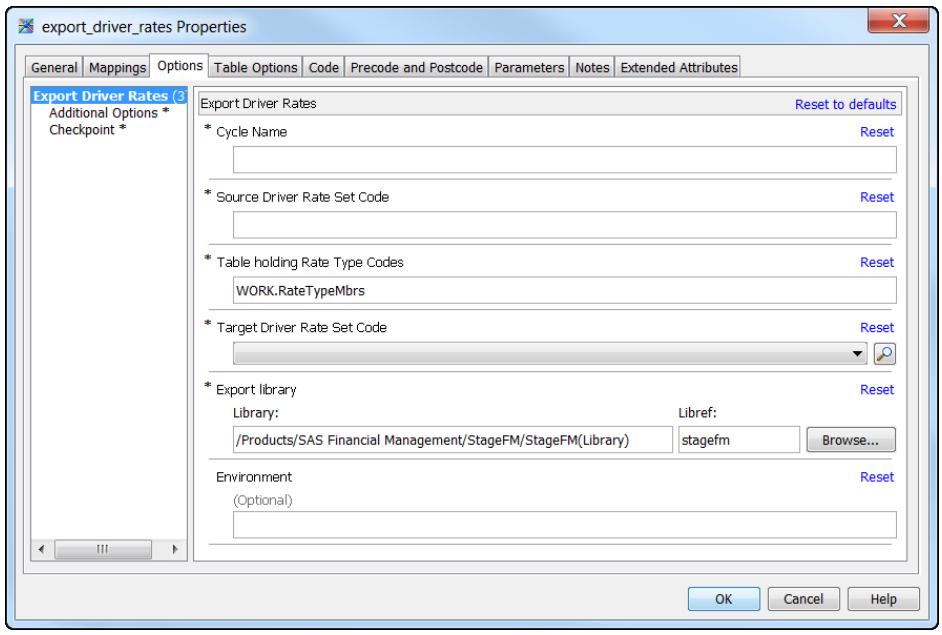
Note: It is a good practice to create and maintain a separate, appropriately named job for each set of option values. Changing the value of a job's options is possible, but likely to generate confusion.

4 Double-click **fm_1510_driver_rates** in the list of jobs.

5 In the Job Editor window, select the **export_driver_rates** transformation and select **Properties** from the pop-up menu. The export_exchange_rates Properties window is displayed.

6 Select the **Options** tab.

Display 13.5 *export_driver_rates Properties Window — Options View*



7 Enter values for the following options:

Option	Description
Cycle Name	Add description.
Source Driver Rate Set Code	Add description.

Option	Description
Table holding Rate Type Codes	Add description.
Target Exchange Rate Code	Add description.
Export Library	Add description.
Environment	(Optional)

8 Click **OK** to save your changes and close the Properties window.

9 Select **File** ▸ **Save**.

10 In the Job Editor window, click **Run**.

11 When the job displays as completed in the Status column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administrator's Guide*.

14

Loading Cell Protection Rules for a Model

Overview

This chapter describes the following topics:

- cell protection rules
- loading cell protection rules for a model
- loading cell protection rules into staging tables
- loading cell protection rules from the staging tables into the Data Mart
- exporting cell protection rules

About Cell Protection Rules for a Model

Note: For information about defining cell protection rules for a model, see the online Help for the Excel add-in or the *SAS Financial Management: User's Guide*.

You can protect cell crossings in a data-entry form by creating one or more rules that apply to the dimensions in the data-entry table.

When you apply a cell protection rule, cells are protected from the following actions:

- manual data entry
- spread
- automatic allocation (applies only to forms in a bottom-up workflow)

However, the values of protected cells can still change as the result of indirect actions, including the following:

- calculations
- changes in the values of descendants that roll up to the protected cell
- changes in cell protection rules
- changes in previous periods when frequency is To Date (for example, Year To Date or Quarter To Date)
- data that is loaded by using SAS Data Integration Studio jobs
- data that was seeded from other models
- rules-based adjustments and allocations
- manual adjustments

SAS Financial Management applies cell protection rules in the following order:

1. Rules that are defined in a model. These rules are inherited by every form set that uses the model.
2. Rules that are defined in a form template. These rules, as well as the rules from the model, are inherited by all forms in the form set.
3. Cell protection that is set in a data-entry form. This protection applies only to the form in which it is defined. You must set form-based cell protection in Microsoft Excel, but the protected cells are visible (and honored) in the Web-based Form Editor as well.

Note: A form cannot override the protection that was set in the form set or the model, and a form set cannot override the protection that was set in the model. For example, if the model rules protect a specific crossing, the form set and its forms cannot undo that protection. However, both the form template and individual forms can define additional cell protection.

Loading Cell Protection Rules into Staging Tables

The following two staging tables are defined in the StageFM library for cell protection rules:

- APP_CELL_PROTECTION_RULE—Defines the rules.
- APP_DIM_TYPE_MEMBER_SELECTOR—Selects the members to which each rule applies. Each rule can apply to one or more members of one or more dimensions.

To load the tables with data, you must write and run a job that loads the data into each table. Before you write a job to load data into the APP_CELL_PROTECTION RULES table and the APP_DIM_TYPE_MEMBER_SELECTOR table, review the column structure of

the tables to ensure that the jobs that you write place the correct data in the correct columns.

To view the column structure of the APP_CELL_PROTECTION_RULE table, complete the following steps:

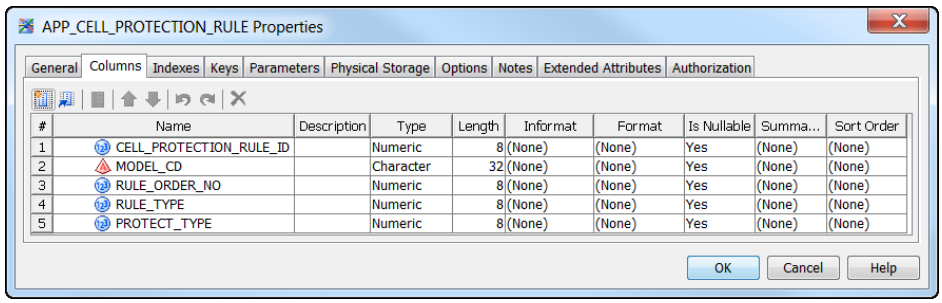
1 In SAS Data Integration Studio, select the **Folders** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.

3 Double-click **APP_CELL_PROTECTION_RULE** in the list of tables. The APP_CELL_PROTECTION_RULE Properties window is displayed.

4 Click the **Columns** tab to view the column structure of the table.

Display 14.1 *APP_CELL_PROTECTION_RULE Properties Window — Columns View*



The APP_CELL_PROTECTION_RULE table contains the following columns:

Column	Description
CELL_PROTECTION_RULE_ID	The rule ID. It must correspond to CELL_PROTECTION_RULE_ID in the APP_DIM_TYPE_MEMBER_SELECTOR table, in a one-to-many relationship.
MODEL_CD	The model code.
RULE_ORDER_NO	The sequence (starting with 1) in which rules are applied for this model.
RULE_TYPE	The type of rule: 0 (protect) or 1 (unprotect).
PROTECT_TYPE	The type of protection: 1 (protection) or 2 (visibility).

5 Click **OK** to close the Properties window.

To view the column structure of the APP_DIM_TYPE_MEMBER_SELECTOR table, complete the following steps:

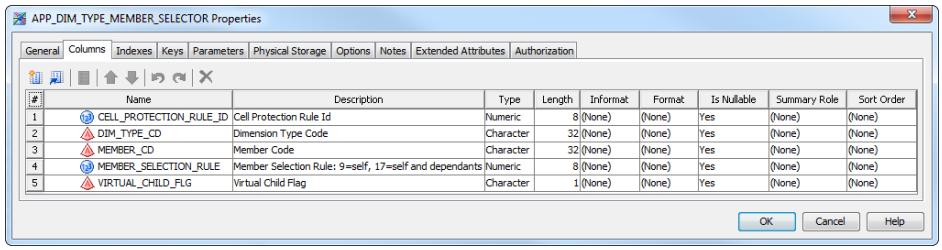
1 In the SAS Data Integration Studio, select the **Folders** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.

3 Double-click **APP_DIM_TYPE_MEMBER_SELECTOR** in the list of tables.

4 Select the **Columns** tab to view the column structure of the table.

Display 14.2 *APP_DIM_TYPE_MEMBER_SELECTOR Properties Window — Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summary Role	Sort Order
1	CELL_PROTECTION_RULE_ID	Cell Protection Rule Id	Numeric	8	(None)	(None)	Yes	(None)	(None)
2	DIM_TYPE_CD	Dimension Type Code	Character	32	(None)	(None)	Yes	(None)	(None)
3	MEMBER_CD	Member Code	Character	32	(None)	(None)	Yes	(None)	(None)
4	MEMBER_SELECTION_RULE	Member Selection Rule: 9=self, 17=self and dependants	Numeric	8	(None)	(None)	Yes	(None)	(None)
5	VIRTUAL_CHILD_FLG	Virtual Child Flag	Character	1	(None)	(None)	Yes	(None)	(None)

The APP_DIM_TYPE_MEMBER_SELECTOR table contains the following columns:

Column	Description
CELL_PROTECTION_RULE_ID	The rule ID. It must correspond to the CELL_PROTECTION_RULE_ID column in table APP_CELL_PROTECTION_RULE.
DIM_TYPE_CD	The dimension type code.
MEMBER_CD	The member code.
MEMBER_SELECTION_RULE	The following lists the valid values for this column: ■ 0: The rule does not apply to the member or any of its descendants. ■ 2: The rule applies only to the leaf descendants. ■ 4: The rule applies only to the immediate subordinate members. ■ 8: The rule applies only to all of the subordinate members. ■ 9: The rule applies only to the specified member. ■ 11: The rule applies to the member and the leaf descendants. ■ 13: The rule applies to the member and all of its immediate subordinate members. ■ 17: The rule applies to the member and all of its descendants.
VIRTUAL_CHILD_FLG	Specifies whether the member is a virtual child. Valid values are N and Y.

5 Click **OK** to close the Properties window.

To load data into the APP_CELL_PROTECTION_RULE table and the APP_DIM_TYPE_MEMBER_SELECTOR table, complete the following steps:

1 Create a SAS Data Integration Studio job that loads data from its source into the appropriate table in the StageFM library.

2 Click **OK** to save the job.

3 Run the job and validate that the table loaded successfully.

Note: For detailed information about creating and working with SAS Data Integration Studio jobs to extract data from a source table and load it into a target table, see the *SAS Data Integration User's Guide*.

Loading Cell Protection Rules into the Data Mart

You can load cell protection rules for a model from the staging tables into the Data Mart by using a SAS Data Integration Studio job or by using a SAS macro. Each time you run the SAS Data Integration Studio job or the macro, it loads cell protection rules for the specified model. The job or macro deletes any rules that previously existed for that model and loads the rules that are defined in the staging tables. However, loading an empty data set is not supported, and attempting to do so results in a failure.

Loading Cell Protection Rules into the Data Mart by Using a Job

To load cell protection rules by using SAS Data Integration Studio job, complete the following steps:

1 In SAS Data Integration Studio, select the **Folder** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.4 Jobs**.

3 Make a copy of the fm_2100_import_cell_protection_rules job.

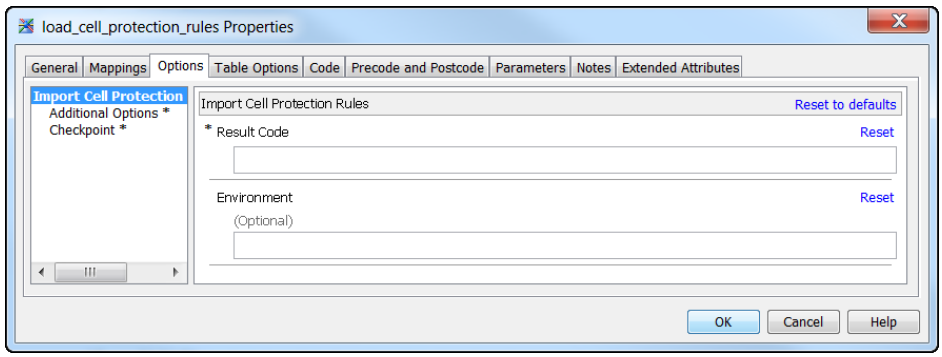
Note: It is a good practice to create and maintain a separate, appropriately named job for each set of option values. Changing the value of a job's options is possible, but likely to generate confusion.

4 Double-click the job in the list of jobs.

5 In the Job Editor window, select the load_cell_protection_rules transformation and select **Properties** from the pop-up menu. The load_cell_protection_rules Properties window is displayed.

6 Select the **Columns** tab.

Display 14.3 *load_cell_protection_rules Properties Window — Options View*



7 Enter values for the following options:

Option	Description
Result Code	Code for the model whose rules you are loading.
Environment	(Optional) Environment that is defined in your EnvironmentFactory.xml file (for authentication purposes). If you leave this field empty, then the default environment is used. (The default value is `default`.

8 Click **OK** to save your changes and close the Properties window.

9 In the Job Editor window, click **Run**.

10 When the job displays as completed in the Status column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

Loading Cell Protection Rules into the Data Mart by Using a SAS Macro

The %ETLLDCPR SAS macro loads cell protection rules for a specified model from the staging tables to the Data Mart.

To load cell protection rules from the staging tables into the Data Mart by using a SAS macro, run the macro as follows:

ETLLDCPR(resultCode, <environment>)

where,

- *resultCode*—Code for the model whose rules you are loading. Only rules for the specified model are loaded.
- *environment*—(Optional) Name of the middle-tier environment (for authentication purposes). The default value is `default`.

When loading cell protection rules into the Data Mart by using the %ETLLDCPR macro, note the following:

- You can invoke the macro from an interactive SAS session, or you can write a stored process that calls the macro.
- On a Windows server, the etlldcpr.sas macro file is located in the following directory: `!SASROOT\finance\sasmacro`.
- On a UNIX server, the etlldcpr.sas macro file is located in the following directory: `!SASROOT/sasautos`.

Note: To run the macro, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

Exporting Cell Protection Rules

About Exporting Cell Protection Rules

You can export cell protection rules for a model by using one of the following methods:

- A SAS Data Integration Studio job.
- A SAS macro.

Exporting Cell Protection Rules by Using a Job

To export cell protection rules by using a SAS Data Integration Studio job, complete the following steps:

1 In SAS Data Integration Studio, select the **Folder** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.

3 Make a copy of the fm_2300_export_cell_protection_rules job.

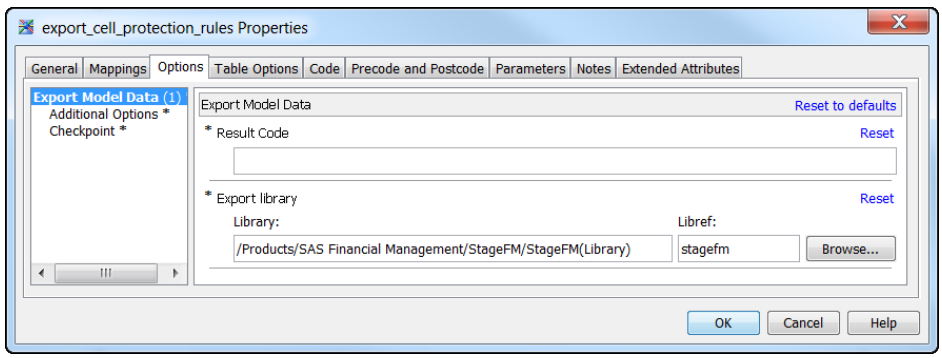
Note: It is a good practice to create and maintain a separate, appropriately named job for each set of option values. Changing the value of a job's options is possible, but likely to generate confusion.

4 Double-click to select the job in the list of jobs.

5 In the Job Editor window, select the export_cell_protection_rules transformation and select **Properties** from the pop-up menu. The export_cell_protection_rules Properties window is displayed.

6 Select the **Options** tab.

Display 14.4 *export_cell_protection_rules Properties Window — Options View*



7 Enter values for the following options:

Option	Description
Result Code	Code for the model whose rules you are exporting.

Option	Description
Export library	Target data library to where you want to export the cell protection rules. Select from the available data libraries. The default value is the StageFM library.

8 Click **OK** to save your changes and close the window.

9 Select **File** ▸ **Save**.

10 In the Job Editor window, click **Run**.

11 When the job displays as completed in the Status column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

Exporting Cell Protection Rules by Using a SAS Macro

The %ETLCPREX SAS macro exports cell protection rules for a specified model from the staging tables to the Data Mart.

To export cell protection rules to the staging tables into the Data Mart by using a SAS macro, run the macro as follows:

ETLCPREX(resultCode, <exportLib>)

where,

- *resultCode*—Code for the model whose rules you are exporting. Only rules for the specified model are loaded.
- *exportLib*—Target data library to where to export the cell protection rules. Select from the available data libraries.

When exporting cell protection rules from the Data Mart by using the %ETLCPREX macro, note the following:

- You can invoke the macro from an interactive SAS session, or you can write a stored process that calls the macro. Before you run the macro from a SAS session, the target data library has to be assigned in the SAS session.
- On a Windows server, the etlcprex.sas macro file is located in the following directory: `!SASROOT\finance\sasmacro`.
- On a UNIX server, the etlcprex.sas macro file is located in the following directory: `!SASROOT/sasautos`.

Note: To run the macro, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

15

Loading Cell Visibility Rules for a Model

Overview

This chapter describes the following topics:

- cell visibility rules
- loading cell visibility rules for a model
- loading cell visibility rules into staging tables
- loading cell visibility rules from staging tables into the Data Mart
- exporting cell visibility rules

About Cell Visibility Rules

Note: For information about defining cell visibility rules for a model, see the online Help for the Excel add-in or the *SAS Financial Management: User's Guide*.

Using cell visibility rules, you can choose which cells are visible and which cells are hidden in a data-entry forms and reports. You apply cell visibility rules to the dimensions in the data-entry table. Cell visibility rules hide data that is not necessary or not of interest. Visibility rules do not provide data security.

Note: With SAS Financial Management 5.4, system filters are implemented by using cell visibility rules. When migrating from a prior release, system filters are converted to visibility rules during the migration process. For more information about the migration process, see the *SAS Financial Management: Migration Guide*.

Even though a cell is not visible, the value of cell can still change as the result of indirection actions, including the following:

- calculations
- changes in the values of descendants that roll up to the hidden cell
- data that is loaded by using SAS Data Integration Studio jobs
- data that was seeded from other models
- rules-based adjustments and allocations
- manual adjustments

Cell visibility rules are applied in the following order:

1. Rules that are defined in a model. These rules are inherited by every form that uses the model.
2. Rules that are defined in a form template. These rules, as well as the rules from the model, are inherited by all forms in the form set.
3. Cell visibility that is set in a data-entry form. This visibility applies only to the form in which it is defined. You must set form-based cell visibility in Microsoft Excel, but the hidden cells are visible (and honored) in the Web-based Form Editor as well.

Note: A form cannot override the visibility that was set in the form set or the model, and a form cannot override the visibility that was set in the model. For example, if the model rules hide a specific crossing, the form set and its forms cannot unhide, or reveal, the crossing. However, both the form template and individual forms can define additional cell visibility.

Loading Cell Visibility Rules into Staging Tables

The following two staging tables are defined in the StageFM library for cell visibility rules:

- APP_CELL_PROTECTION_RULE—Defines the cell protection or cell visibility rules.
- APP_DIM_TYPE_MEMBER_SELECTOR—Selects the members to which each rule applies. Each rule can apply to one or more members of one or more dimensions.

To load the tables with data, you must write and run a job that loads the data into each table. Before you write a job to load data into the APP_CELL_PROTECTION_RULE table and the APP_DIM_TYPE_MEMBER_SELECTOR table, review the column structure of the tables to ensure that the jobs that you write place the correct data in the correct columns.

To view the column structure of the APP_CELL_PROTECTION_RULE table, complete the following steps:

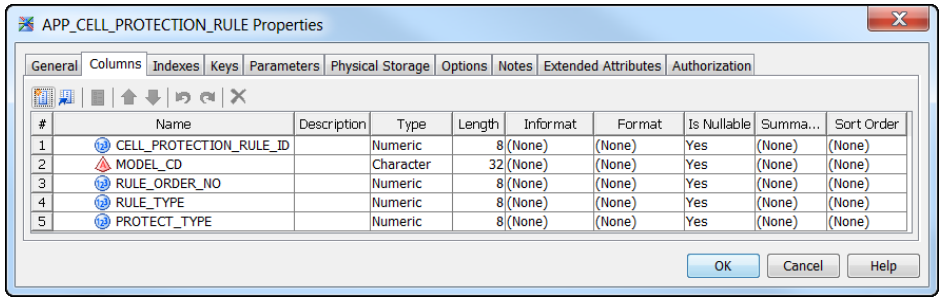
1. In the SAS Data Integration Studio, select the **Folders** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.

3 Double-click **APP_CELL_PROTECTION_RULE** in the list of tables. The APP_CELL_PROTECTION_RULE Properties window is displayed

4 Select the **Columns** tab to view the column structure of the table.

Display 15.1 *APP_CELL_PROTECTION_RULE Properties Window — Columns View*



The APP_CELL_PROTECTION_RULE table contains the following columns:

Column	Description
CELL_PROTECTION_RULE_ID	The rule ID. It must correspond to CELL_PROTECTION_RULE_ID in the APP_DIM_TYPE_MEMBER_SELECTOR table, in a one-to-many relationship.
MODEL_CD	The model code.
RULE_ORDER_NO	The sequence (starting with 1) in which rules are applied for this model.
RULE_TYPE	The type of rule: 0 (protect) or 1 (unprotect).
PROTECT_TYPE	The type of protection: 1 (protection) or 2 (visibility).

5 Click **OK** to close the Properties window.

To view the column structure of the APP_DIM_TYPE_MEMBER_SELECTOR table, complete the following steps:

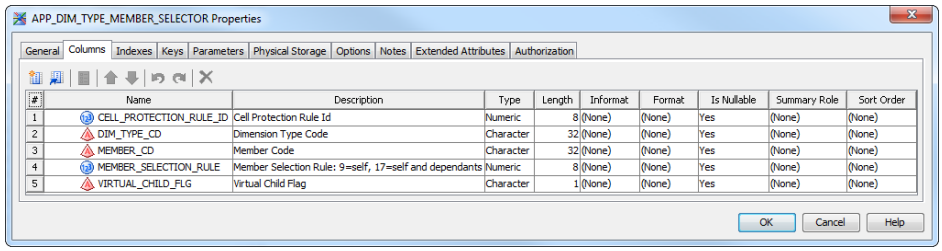
1 In the SAS Data Integration Studio, select the **Folders** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.

3 Double-click **APP_DIM_TYPE_MEMBER_SELECTOR** in the list of tables.

4 Select the **Columns** tab to view the column structure of the table.

Display 15.2 *APP_DIM_TYPE_MEMBER_SELECTOR Properties Window — Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summary Role	Sort Order
1	CELL_PROTECTION_RULE_ID	Cell Protection Rule Id	Numeric	8	(None)	(None)	Yes	(None)	(None)
2	DIM_TYPE_CD	Dimension Type Code	Character	32	(None)	(None)	Yes	(None)	(None)
3	MEMBER_CD	Member Code	Character	32	(None)	(None)	Yes	(None)	(None)
4	MEMBER_SELECTION_RULE	Member Selection Rule: 9=self, 17=self and dependants	Numeric	8	(None)	(None)	Yes	(None)	(None)
5	VIRTUAL_CHILD_FLG	Virtual Child Flag	Character	1	(None)	(None)	Yes	(None)	(None)

The APP_DIM_TYPE_MEMBER_SELECTOR table has the following columns:

Column	Description
CELL_PROTECTION_RULE_ID	Rule ID. It must correspond to the CELL_PROTECTION_RULE_ID column in table APP_CELL_PROTECTION_RULE.
DIM_TYPE_CD	Dimension type code.
MEMBER_CD	Member code.
MEMBER_SELECTION_RULE	Specifies how the rule is applied. The following lists the valid values for this column: ■ 0: The rule does not apply to the member or any of its descendants. ■ 2: The rule applies only to the leaf descendants. ■ 4: The rule applies only to the immediate subordinate members. ■ 8: The rule applies only to all of the subordinate members. ■ 9: The rule applies only to the specified member. ■ 11: The rule applies to the member and the leaf descendants. ■ 13: The rule applies to the member and all of its immediate subordinate members. ■ 17: The rule applies to the member and all of its descendants.
VIRTUAL_CHILD_FLG	Specifies whether the member is a virtual child. Valid values are N and Y.

5 Click **OK** to close the Properties window.

Note: Each rule can apply to one or more members of one or more dimensions.

To load cell visibility rule data into the APP_CELL_PROTECTION_RULE table and the APP_DIM_TYPE_MEMBER_SELECTOR table, complete the following steps:

1 Create a SAS Data Integration Studio job that loads data from its source into the appropriate table in the StageFM library.

2 Click **OK** to save the job.

3 Run the job and validate that the table loaded successfully.

Note: For detailed information about creating and working with SAS Data Integration Studio jobs to extract data from a source table and load it into a target table, see the *SAS Data Integration User's Guide*.

Loading Cell Visibility Rules into the Data Mart

You can load cell visibility rules for a model from the staging tables into the Data Mart by using the fm_2150_import_cell_visibility_rules job.

Note: When you run the SAS Data Integration Studio job, it loads cell visibility rules for the specified model. The job deletes any rules that previously existed for that model and loads the rules that are defined in the staging tables.

To load cell visibility rules by using a SAS Data Integration Studio job, complete the following steps:

1 In SAS Data Integration Studio, select the **Folder** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.

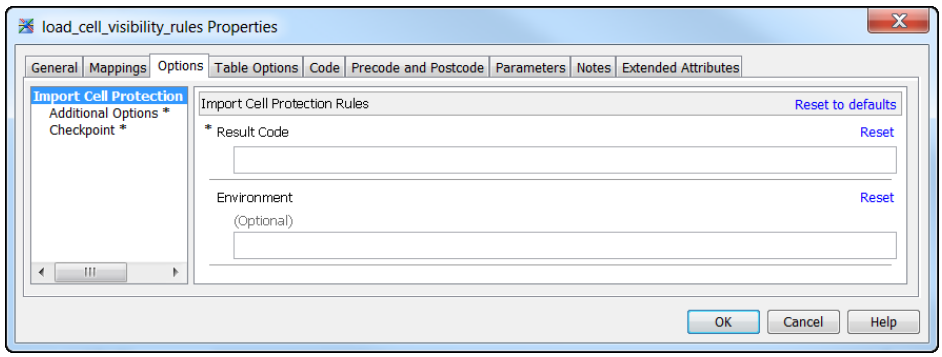
3 Make a copy of the fm_2150_import_cell_visibility_rules job.

Note: It is a good idea to create and maintain a separate, appropriately named job for each set of option values. Changing the option values of a job occasionally is possible, but it is likely to generate confusion.

4 In the Job Editor window, select the load_cell_visibility_rules transformation and select **Properties** from the pop-up menu. The load_cell_visibility_rules Properties window is displayed.

5 Select the **Options** tab.

Display 15.3 *load_cell_visibility_rules Properties Window — Options View*



6 Enter values for the following options:

Option	Description
Result Code	Code for the model whose rules you are loading.
Environment	(Optional) Name of the middle-tier environment (for authentication purposes). The default value is `default`.

7 Click **OK** to save your changes and close the Properties window.

8 Select **File** ▸ **Save**.

9 In the Job Editor window, click **Run**.

10 When the job displays as completed in the **Status** column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

Exporting Cell Visibility Rules

The fm_2350_export_cell_visibility_rules job exports cell visibility rules.

To export cell visibility rules by using fm_2350_export_cell_visibility_rules job, complete the following steps:

1 In SAS Data Integration Studio, select the **Folder** tab.

2 In the Folders tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.

3 Make a copy of the fm_2350_export_cell_visibility_rules job.

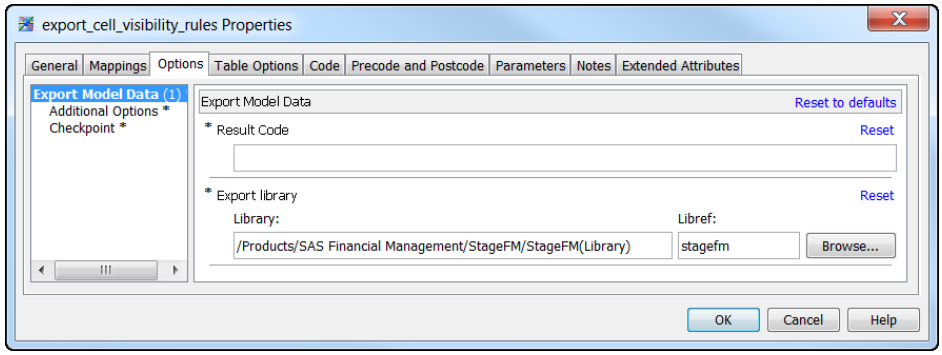
Note: It is a good idea to create and maintain a separate, appropriately named job for each set of option values. Changing the option values of a job occasionally is possible, but it is likely to generate confusion.

4 Double-click to select the job.

5 In theJob Editor window, select the export_cell_visibility_rules transformation and select **Properties** from the pop-up menu. The export_cell_visibility_rules Properties is displayed.

6 Select the **Options** tab.

Display 15.4 *export_cell_visibility_rules Properties Window — Options View*



7 Enter values for the following options:

Option	Description
Result Code	Code for the model whose rules you are loading.
Export library	Target data library to where you want to export the cell visibility rules. Select from the available data libraries. The default value is the StageFM library.

8 Click **OK** to save your changes and close the Properties window.

9 Select **File** ▸ **Save**.

10 In the Job Editor window, click **Run**.

11 When the job displays as completed in the Status column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

16

Loading Data Validation Rules for a Model

Overview

This chapter describes the following topics:

- data validation rules
- loading data validation rules for a model
- loading data validation rules into staging tables
- loading data validation rules from the staging tables into the Data Mart
- exporting data validation rules

Data Validation Rules

Data validation ensures that values in a data-entry table comply with certain constraints. For example, a company might want to make sure that employee bonuses do not exceed a specified percentage, or that new hiring does not exceed specified limits.

Note: Validation rules are defined in SAS Financial Management Studio at the model level or the form set level.

Loading Data Validation Rules for a Model

You load data validation rules for a model by using one of the following methods:

- In SAS Data Integration Studio, you can run a job that loads the data validation rules for the selected model from the staging tables to the Data Mart.
- In the Models workspace of SAS Financial Management Studio, you can select a model and then select **Show data validation rules for this model**. The Data Validation window appears. In the Data Validation window, you can define data validation rules for the model.

  For information about defining data validation rules for a model, see the SAS Financial Management Studio online Help or the “Working with Forms and Form Sets” in the *SAS Financial Management: User's Guide*.

  Note: If you subsequently load the Data Mart database via a job, the rules that you defined in SAS Financial Management Studio are deleted.

Loading Data Validation Rules into Staging Tables

The following two staging tables are defined in the StageFM library for data validation rules:

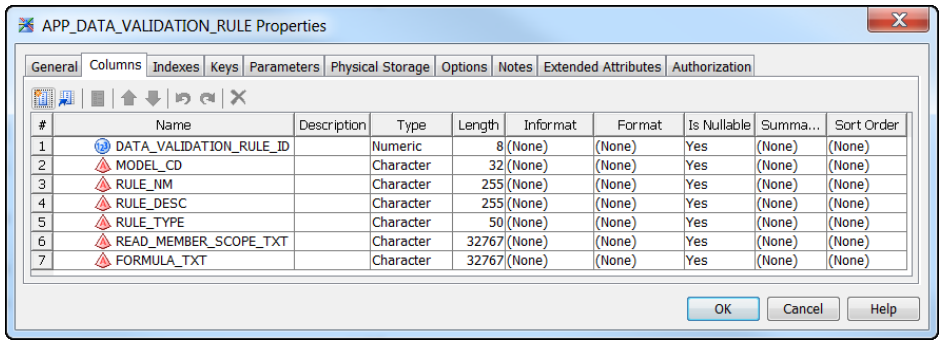
- APP_DATA_VALIDATION_RULE—Defines the rules.
- APP_DATA_VALIDATION_RULE_NLS—Selects the members to which each rule applies. Each rule can apply to one or more members of one or more dimensions.

To load the tables with data, you must write and run a job that loads the data into each table. Before you write a job to load data into the APP_DATA_VALIDATION_RULE table and the APP_DATA_VALIDATION_RULE_NLS table, review the column structure of the tables to ensure that the jobs that you write place the correct data in the correct columns.

To view the column structure of the APP_DATA_VALIDATION_RULE table, complete the following steps:

1. In the SAS Data Integration Studio, select the **Folders** tab.
2. In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.
3. Double-click **APP_DATA_VALIDATION_RULE** in the list of tables. The APP_DATA_VALIDATION_RULE Properties window is displayed.
4. Select the **Columns** tab to view the column structure of the table.

Display 16.1 *APP_DATA_VALIDATION_RULE Properties Window — Columns View*



The APP_DATA_VALIDATION_RULE table has the following columns:

Column	Description
DATA_VALIDATION_RULE_ID	Rule ID.
MODEL_CD	Model code.
RULE_NM	Name of the rule.
RULE_TYPE	Type of rule: ERROR/ERROR_COMMENT/WARNING.
RULE_DESC	Description of the rule.
READ_MEMBER_SCOPE_TXT	Rule read member scope used to limit where the rule executes.
FORMULA_TXT	Data validation rule.

5 Click **OK** to close the Properties window.

To view the column structure of the APP_DATA_VALIDATION_RULE_NLS table, complete the following steps.

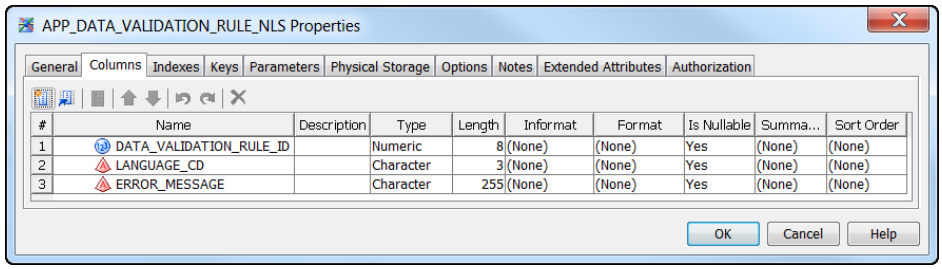
1 In the SAS Data Integration Studio, select the **Folders** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.

3 Double-click **APP_DATA_VALIDATION_RULE_NLS** from the list of tables.

4 Select the **Columns** tab to view the column structure of the table.

Display 16.2 *APP_DATA_VALIDATION_RULE_NLS Properties Window — Columns View*



The APP_DATA_VALIDATION_RULE_NLS table contains the following columns:

Column	Description
DATA_PROTECTION_RULE_ID	Rule ID.
LANGUAGE_CD	Language code.
ERROR_MESSAGE	Error message that is displayed if the rule finds invalid data.

5 Click **OK** to close the Properties window.

Note: Each rule can apply to one or more members of one or more dimensions.

To load data validation rule data into the APP_DATA_VALIDATION_RULE table and the APP_DATA_VALIDATION_RULE_NLS table, complete the following steps:

1 Create a SAS Data Integration Studio job that loads data from its source into the appropriate table in the StageFM library.

2 Click **OK** to save the job.

3 Run the job and validate that the table loaded successfully.

Note: For detailed information about creating and working with SAS Data Integration Studio jobs to extract data from a source table and load it into a target table, see the *SAS Data Integration User's Guide*.

Loading Data Validation Rules into the Data Mart

The fm_2400_import_data_validation_rules job loads data validation rules from the staging area into the Data Mart.

To load data validation rules by using the fm_2400_import_data_validation_rules job, complete the following steps:

1 In SAS Data Integration Studio, select the **Folder** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.

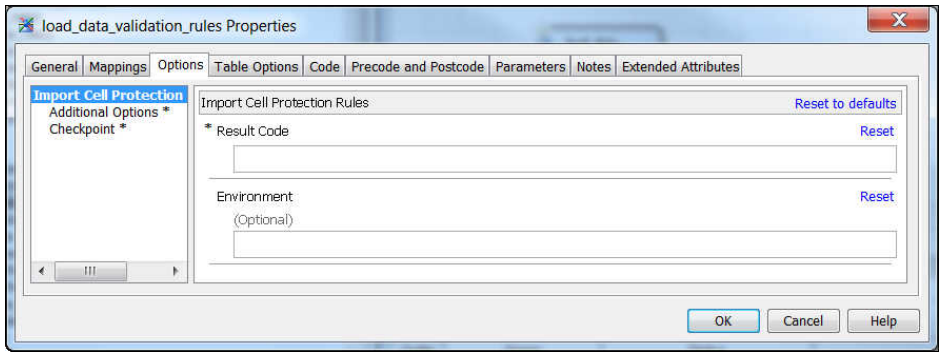
3 Make a copy of the fm_2400_import_data_validation_rules job.

Note: It is a good idea to create and maintain a separate, appropriately named job for each set of option values. Changing the option values of a job occasionally is possible, but it is likely to generate confusion.

4 In the Job Editor window, select the **load_data_validation_rules** transformation and select **Properties** from the pop-up menu.

5 Select the **Options** tab.

Display 16.3 *load_data_validation_rules Properties Window — Options View*



6 Enter values for the following options:

Option	Description
Result Code	Code for the model whose rules you are loading.
Environment	(Optional) Name of the middle-tier environment (for authentication purposes). The default value is `default`.

7 Click **OK** save your changes and closer the Properties window.

8 Select **File** ▸ **Save**.

9 In the Job Editor window, click **Run**.

10 When the job displays as completed in the Status column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

Exporting Data Validation Rules

To export data validation rules, you use the fm_2410_export_data_validation_rules job.

To export data validation rules by using the fm_2410_export_data_validation_rules job, complete the following steps:

1 In SAS Data Integration Studio, select the **Folder** tab.

2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.

3 Make a copy of the fm_2410_export_data_validation_rules job.

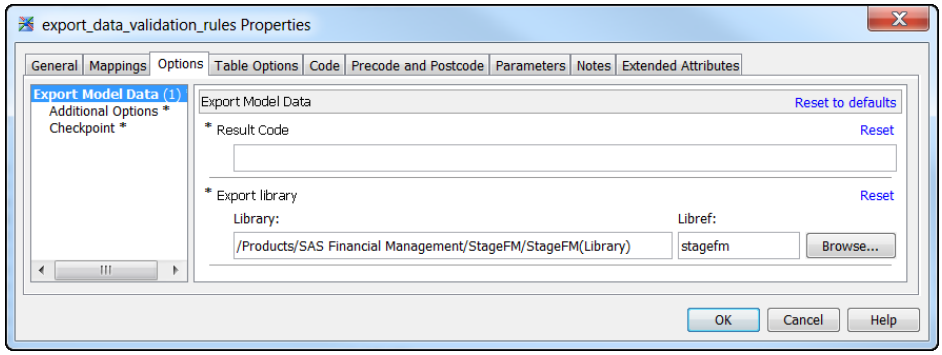
Note: It is a good idea to create and maintain a separate, appropriately named job for each set of option values. Changing the option values of a job occasionally is possible, but it is likely to generate confusion.

4 Double-click to select the job.

5 In the Job Editor window, select the export_data_validation_rules transformation and select **Properties** from the pop-up menu. The export_data_validation_rules Properties window is displayed.

6 Select the **Options** tab.

Display 16.4 *export_data_validation_rules Properties Window — Options View*



7 Enter values for the following options:

Option	Description
Result Code	Code for the model whose rules you are exporting.
Export library	Target data library to where you want to export the data validation rules. Select from the available data libraries. The default value is the StageFM library.

8 Click **OK** to save your changes and close the Properties window.

9 Select **File** ▸ **Save**.

10 In the Job Editor window, click **Run**.

11 When the job displays as completed in the Status column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

17

Loading Base Data into a Financial Cycle

Overview

This chapter describes the following topics:

- base financial data
- working with base financial data staging tables
- loading base financial data into staging tables
- loading base financial data from the staging tables into the Data Mart
- deleting facts from a cycle

About Base Financial Data

As a SAS Financial Management data administrator, you load facts into *cycles* in SAS Financial Management. A cycle is a structured pool of stored data that is open for input and modification at a specific time and locked against changes at

a specific later time. General ledgers and Enterprise Resource Planning (ERP) systems are common sources of external data that you can load to SAS Financial Management. You can also load facts as balances or in terms of activity.

Note: If you load both general ledger data and journal data, ensure that you do not load journal data that is already in the general ledger data that you are loading. Loading duplicate data would lead to double-counting of the same financial transactions.

Working with Base Financial Data Staging Tables

Columns That Hold Members

The following columns must contain a valid member code in every record because they represent dimension types that are automatically included in every cycle:

- Initiating Internal Organization ID (Organization)
- Affected Internal Organization ID (Trader)
- GL Account ID
- Analysis ID
- Currency Code
- Time Period ID

The SOURCE_INTERNAL_ORG_ID column must contain a valid member code in every record. This column indicates the organization that is the source of the data record. In many cases, this is the same organization that you place in the INITIATING_INTERNAL_ORG_ID column, which indicates the organization that the record describes.

The member codes in a record must satisfy the following constraints:

- The Organization code must not be ALL or EXT.
- The Organization code and the Trader code must be different.
- If the account that is specified in the GL Account ID column has a value of *N* for its Intercompany Account Flag, then the value of Affected Internal Organization ID (Trader) must be EXT. (This constraint applies only if you load the data into a cycle for which the "Non-intercompany accounts must be associated with the external trading member" property is set.)
- If the specified account has a value of *Y* for its Intercompany Account Flag, then the value of Affected Internal Organization ID (Trader) must not be EXT. (This constraint applies only if you load the data into a cycle for which the "Intercompany accounts must be associated with an intercompany trading partner" property is set.)

Leave the following columns empty if these dimension types are not used to describe the data that you are loading:

- Cost Center ID

- Profit Center ID
- External Organization
- Item Category Code

If you add other dimension types to your data model, then they are represented by additional columns in GL_TRANSACTION_SUM and GL_JRNL_DETAILS that are not shown here. You must provide valid member codes for any dimension type that is included in the cycle that is the destination of the data. For a discussion of adding dimension types, see Chapter 10, “Adding a Dimension Type,” on page 65.

Columns That Specify the Numeric Values

About Numeric Values

In both GL_TRANSACTION_SUM and GL_JRNL_DETAILS, the Transaction Amount column holds the base numeric values.

In GL_TRANSACTION_SUM, the interpretation of the Transaction Amount values is affected by the Transaction Amount Year-to-Date Flag column. For each record, you must load this column with either a *Y* or an *N*. If you leave the Transaction Amount Year-to-Date Flag column empty, then the record is ignored.

GL_JRNL_DETAILS does not have a Transaction Amount Year-to-Date Flag column. Every record in that table is processed in the same manner as an *N* record in GL_TRANSACTION_SUM.

The explanation of the *Y/N* choice for the Transaction Amount Year-to-Date Flag column follows.

Setting the Year-To-Date Flag

For a Revenue or Expense account, the value that is stored in the Data Mart must represent the revenue received or expense incurred during the designated time period. For an Asset, Liability, or Equity account, the value that is stored in the Data Mart must represent the change in the value of the asset, liability, or equity item from the previous time period to the designated time period. SAS Financial Management computes the values of Asset, Liability, and Equity accounts by summing up a history of stored changes in value. All the numeric values that are stored in the Data Mart are called *period activity* values.

For each record of GL_TRANSACTION_SUM, if you load the period activity value that is required by the Data Mart into the Transaction Amount column, then you should place *N* in the Transaction Amount Year-to-Date Flag column. Thus, use *N* for the flag in the following cases:

- The record concerns a Revenue account and the Transaction Amount is the revenue received during the designated time period.
- The record concerns an Expense account and the Transaction Amount is the expense incurred during the designated time period.
- The record concerns an Asset, Liability, or Equity account. The Transaction Amount is the change in the value of the asset, liability, or equity item from the previous time period to the designated time period.

You should use *Y* for the flag in the following cases:

- The record concerns a Revenue account and the Transaction Amount is the cumulative year-to-date revenue through the designated time period.
- The record concerns an Expense account and the Transaction Amount is the cumulative year-to-date expense through the designated time period.
- The record concerns an Asset, Liability, or Equity account and the Transaction Amount is the value of the asset, liability, or equity item in the designated time period.

For Statistical accounts, the Year-to-Date Flag is ignored, but still you must specify either *Y* or *N*.

How Year-To-Date Transaction Amounts Are Processed

For Statistical accounts, transaction amounts are always loaded without change into the Data Mart, whether the Year-to-Date Flag is *Y* or *N*.

For all other account types, if the year-to-date flag is *Y*, then the period activity value that is placed in the Data Mart is generally calculated as the Transaction Amount in GL_TRANSACTION_SUM for the same time period minus the Transaction Amount in GL_TRANSACTION_SUM for the previous time period. For example, a March year-to-date transaction amount of 100 and a February year-to-date transaction amount of 94 together yield a March period activity value of 6 in the Data Mart.

There are two important exceptions to this rule. A year-to-date Transaction Amount in GL_TRANSACTION_SUM is carried forward without change to the Data Mart if either of the following conditions is true:

- GL_TRANSACTION_SUM does not contain a corresponding record for the previous time period.
- The record concerns a Revenue or Expense account and the designated time period is the first period of a fiscal year, as determined by the relevant time hierarchy.

Note that the difference between the year-to-date values for two consecutive time periods can be calculated only if the table contains records for both time periods. This is so, even if the year-to-date value for one of the time periods is zero. If you set the year-to-date flag to *Y*, then be sure to include records for an unbroken sequence of time periods, including records with a Transaction Amount of zero where necessary.

How Multiple Records for the Same Combination of Members Are Processed

It is likely that your staging tables contain at most one record for a given combination of members. However, this is not a requirement. You can create as many data records as you want for the same combination of members. You can even create a mix of year-to-date and non-year-to-date data records for the same combination of members. That would be pointless and confusing in most cases, but the software can handle it.

Suppose that you create many data records for the same combination of members, possibly including a mix of year-to-date and non-year-to-date records. The period activity values that are loaded into the Data Mart are computed as follows:

1. All year-to-date transaction amounts for a given combination of members are summed, yielding a net year-to-date amount for that combination of members.
2. The net year-to-date amount for one time period is subtracted from the net year-to-date amount for the following time period. This calculation yields a period activity value that is based solely on the year-to-date amounts.
3. All non-year-to-date transaction amounts for a given combination of members are summed, yielding a period activity value for that combination of members that is based solely on non-year-to-date amounts.
4. For each combination of members, the period activity value that is based solely on year-to-date amounts is added to the period activity value that is based solely on non-year-to-date amounts. This yields the final period activity value that is loaded into the Data Mart.

Loading Base Financial Data into Staging Tables

The following staging tables are defined in the StageFM library for base financial data:

- GL_TRANSACTION_SUM
- GL_JRNL
- GL_JRNL_DETAILS

To load the tables with data, you must write and run a job that loads the data into each table. Before you write a job to load data into the base financial data staging tables, review the column structure of the tables to ensure that the jobs that you write place the correct data in the correct columns.

Before building records for the base financial data staging tables, note the following:

- The column layout for general ledger data is similar to the column layout for journal data.
- In the case of general ledger data, the GL_TRANSACTION_SUM table contains all the columns.
- In the case of journal data, the GL_JRNL table identifies journal entries. Each entry can include several data records in the GL_JRNL_DETAILS table. GL_JRNL contains the columns that must have the same value for all the data records that belong to a given journal entry.
- GL Journal ID must have a unique value for each record in GL_JRNL. You can generate the unique values in any way that you find convenient. In GL_JRNL_DETAILS, the combination of GL Journal ID and GL Journal Line Item Number must be unique for each record.
- The Schema ID column in GL_TRANSACTION_SUM and GL_JRNL is not used. Leave this column blank.

Note: Ensure that you have reviewed the guidelines for working with base financial data staging tables before you write and run the job to load a staging

table. For information about the base financial data staging table guidelines, see “Working with Base Financial Data Staging Tables” on page 128.

To view the column structure of a base financial data table, complete the following steps:

1 In the SAS Data Integration Studio, select the **Folders** tab.

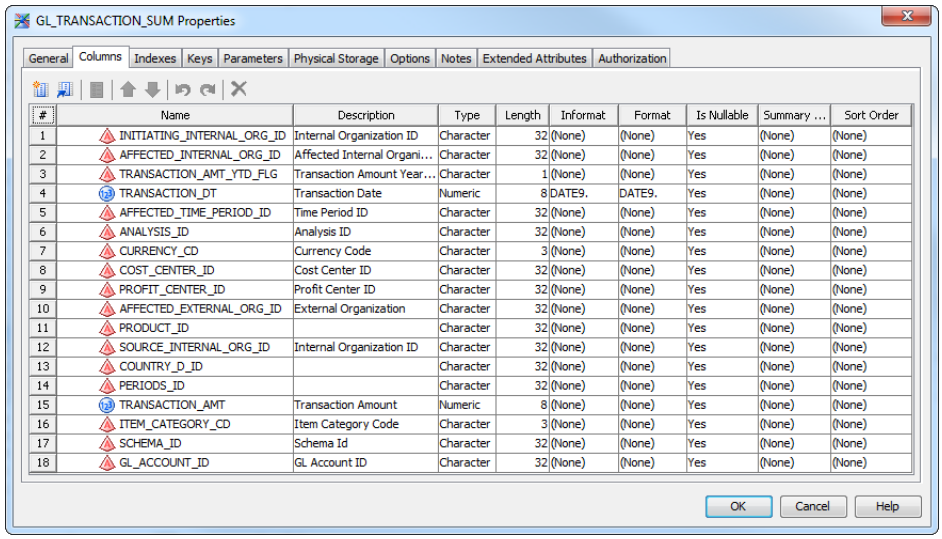
2 In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.

3 From the list of stables, select the base financial data table for which you want to view the column structure.

The Properties window for the table is displayed.

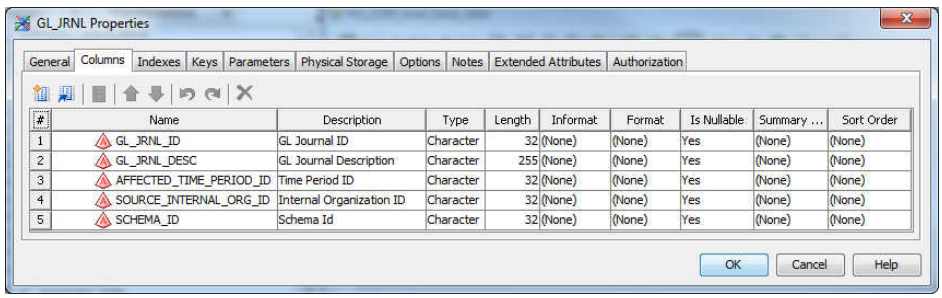
4 Select the **Columns** tab to view the column structure of the table.

Display 17.1 *GL_TRANSACTION_SUM Properties Window — Columns View*



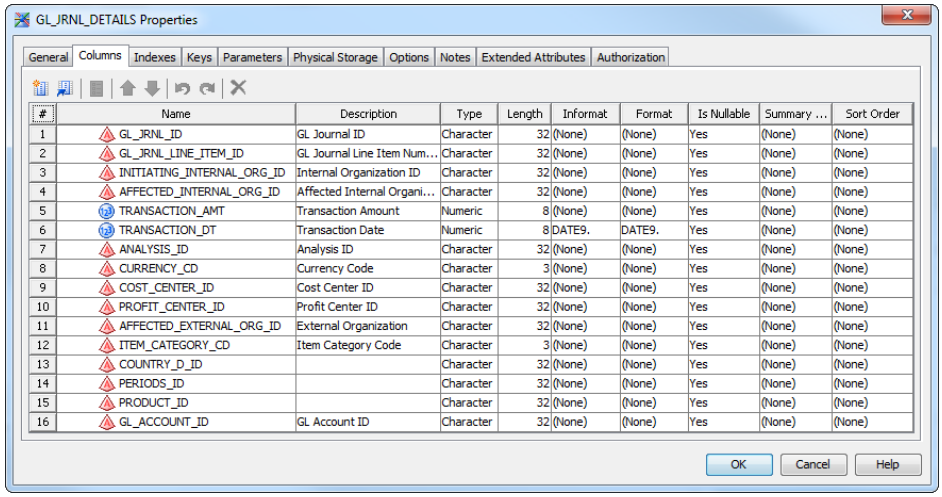
#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summary ...	Sort Order
1	INITIATING_INTERNAL_ORG_ID	Internal Organization ID	Character	32	(None)	(None)	Yes	(None)	(None)
2	AFFECTED_INTERNAL_ORG_ID	Affected Internal Organi...	Character	32	(None)	(None)	Yes	(None)	(None)
3	TRANSACTION_AMT_YTD_FLG	Transaction Amount Year...	Character	1	(None)	(None)	Yes	(None)	(None)
4	TRANSACTION_DT	Transaction Date	Numeric	8	DATE9.	DATE9.	Yes	(None)	(None)
5	AFFECTED_TIME_PERIOD_ID	Time Period ID	Character	32	(None)	(None)	Yes	(None)	(None)
6	ANALYSIS_ID	Analysis ID	Character	32	(None)	(None)	Yes	(None)	(None)
7	CURRENCY_CD	Currency Code	Character	3	(None)	(None)	Yes	(None)	(None)
8	COST_CENTER_ID	Cost Center ID	Character	32	(None)	(None)	Yes	(None)	(None)
9	PROFIT_CENTER_ID	Profit Center ID	Character	32	(None)	(None)	Yes	(None)	(None)
10	AFFECTED_EXTERNAL_ORG_ID	External Organization	Character	32	(None)	(None)	Yes	(None)	(None)
11	PRODUCT_ID		Character	32	(None)	(None)	Yes	(None)	(None)
12	SOURCE_INTERNAL_ORG_ID	Internal Organization ID	Character	32	(None)	(None)	Yes	(None)	(None)
13	COUNTRY_D_ID		Character	32	(None)	(None)	Yes	(None)	(None)
14	PERIODS_ID		Character	32	(None)	(None)	Yes	(None)	(None)
15	TRANSACTION_AMT	Transaction Amount	Numeric	8	(None)	(None)	Yes	(None)	(None)
16	ITEM_CATEGORY_CD	Item Category Code	Character	3	(None)	(None)	Yes	(None)	(None)
17	SCHEMA_ID	Schema Id	Character	32	(None)	(None)	Yes	(None)	(None)
18	GL_ACCOUNT_ID	GL Account ID	Character	32	(None)	(None)	Yes	(None)	(None)

Display 17.2 *GL_JRNL Properties Window — Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summary ...	Sort Order
1	GL_JRNL_ID	GL Journal ID	Character	32	(None)	(None)	Yes	(None)	(None)
2	GL_JRNL_DESC	GL Journal Description	Character	255	(None)	(None)	Yes	(None)	(None)
3	AFFECTED_TIME_PERIOD_ID	Time Period ID	Character	32	(None)	(None)	Yes	(None)	(None)
4	SOURCE_INTERNAL_ORG_ID	Internal Organization ID	Character	32	(None)	(None)	Yes	(None)	(None)
5	SCHEMA_ID	Schema Id	Character	32	(None)	(None)	Yes	(None)	(None)

Display 17.3 *GL_JRNL_Details Properties Window — Columns View*



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summary ...	Sort Order
1	GL_JRNL_ID	GL Journal ID	Character	32	(None)	(None)	Yes	(None)	(None)
2	GL_JRNL_LINE_ITEM_ID	GL Journal Line Item Num...	Character	32	(None)	(None)	Yes	(None)	(None)
3	INITIATING_INTERNAL_ORG_ID	Internal Organization ID	Character	32	(None)	(None)	Yes	(None)	(None)
4	AFFECTED_INTERNAL_ORG_ID	Affected Internal Organi...	Character	32	(None)	(None)	Yes	(None)	(None)
5	TRANSACTION_AMT	Transaction Amount	Numeric	8	(None)	(None)	Yes	(None)	(None)
6	TRANSACTION_DT	Transaction Date	Numeric	8	DATE9.	DATE9.	Yes	(None)	(None)
7	ANALYSIS_ID	Analysis ID	Character	32	(None)	(None)	Yes	(None)	(None)
8	CURRENCY_CD	Currency Code	Character	3	(None)	(None)	Yes	(None)	(None)
9	COST_CENTER_ID	Cost Center ID	Character	32	(None)	(None)	Yes	(None)	(None)
10	PROFIT_CENTER_ID	Profit Center ID	Character	32	(None)	(None)	Yes	(None)	(None)
11	AFFECTED_EXTERNAL_ORG_ID	External Organization	Character	32	(None)	(None)	Yes	(None)	(None)
12	ITEM_CATEGORY_CD	Item Category Code	Character	3	(None)	(None)	Yes	(None)	(None)
13	COUNTRY_D_ID		Character	32	(None)	(None)	Yes	(None)	(None)
14	PERIODS_ID		Character	32	(None)	(None)	Yes	(None)	(None)
15	PRODUCT_ID		Character	32	(None)	(None)	Yes	(None)	(None)
16	GL_ACCOUNT_ID	GL Account ID	Character	32	(None)	(None)	Yes	(None)	(None)

5 Click **OK** to close the Properties window.

To load the data into the base financial data staging tables, complete the following steps:

1 Create a SAS Data Integration Studio job that loads data from its source into the appropriate table in the StageFM library.

2 Click **OK** to save the job.

3 Run the job and validate that the table loaded successfully.

Note: For detailed information about creating and working with SAS Data Integration Studio jobs to extract data from a source table and load it into a target table, see the *SAS Data Integration User's Guide*.

Loading Base Financial Data from the Staging Tables into the Data Mart

About Loading Base Financial Data into the Data Mart

You can load base financial data from the staging area into a cycle in the Data Mart by using the following methods:

- the Load New Data wizard in SAS Financial Management Studio
- a SAS Data Integration Studio job
- a SAS macro

Note: When loading base financial data from the staging area into a cycle in the Data Mart, you must use all three of the staging area tables (GL_TRANSACTION_SUM,GL_JRNL, and GL_JRNL_DETAILS), unless one of the tables does not contain any relevant records.

Loading Base Financial Data into the Data Mart by Using the Load New Data Wizard

To load base financial data by using SAS Financial Management Studio, complete the following steps:

1. Open the cycle into which you want to load the data.
2. Select the Periods workspace.
3. Select the period or periods for which you want to load data.
4. Select **Load New Data**. The Load New Data wizard launches.
5. Work through the wizard, referring to the online Help as necessary.

Load Base Financial Data into the Data Mart by Using a Job

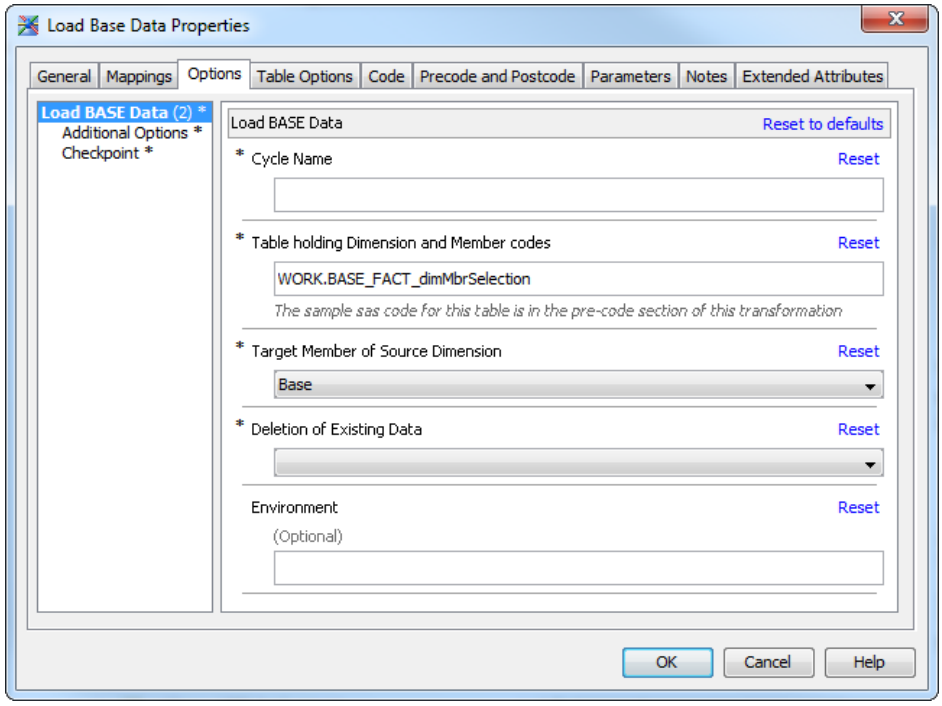
You can use the following SAS Data Integration Studio jobs to load base financial data into the Data Mart:

- fm_1100_load_base_data job—Loads base financial data into unlocked periods.
- fm_1100_load_base_data_unlock_periods—Unlocks locked target periods, loads base financial data, and locks the period that it unlocked.

To load base financial data by using a SAS Data Integration Studio job, complete the following steps:

1. In SAS Data Integration Studio, select the **Folder** tab.
2. In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.
3. Double-click to select the job in the list of jobs.
4. In the Job Editor window, select the Load Base Data transformation, and select **Properties** from the pop-up menu. The Properties window for the job that you selected is displayed.
5. Select the **Options** tab.

Display 17.4 *Load Base Data Properties Window — Options View*



6 Enter values for the following options:

Option	Description
Cycle Name	Name of the SAS Financial Management cycle into which you are loading the data.
Table holding Dimension and Member codes	Name of a SAS data set that specifies the dimension and member combinations to load data for. The **Precode** region on the **Precode and Postcode** tab contains sample code that builds a SAS data set with the required layout. By default, the job uses the SAS data set that is built by the Precode program. On the **Precode and Postcode** tab, complete one of the following tasks: ■ Ensure that the **Precode** check box is selected. Ensure that the name of the SAS data set that is specified for this option matches the name of the table specified in the Table holding Dimension and Member codes option. Modify the precode to build the table that you need. In this case, the precode runs before the job, and then the job uses the SAS data set that the precode builds. ■ Ensure that the **Precode** check box is not selected. Build the SAS data set that you need to use by using a method other than the precode. Ensure that the name of the SAS data set that is specified for this option matches the name of the SAS data set that you built for this purpose.

Option	Description
Target Member of Source Dimension	Specifies the relevant general ledger data in the GL_TRANSACTION_SUM table to associate with the Base member of the Source hierarchy or to associate with the BaseForm member of the Source hierarchy. ■ Select **Base** from the drop-down list to associate the relevant general ledger data in GL_TRANSACTION_SUM with the Base member of the Source hierarchy. ■ Select **BaseForm** from the drop-down list to associate the relevant general ledger data in GL_TRANSACTION_SUM with the BaseForm member of the Source hierarchy.

Option	Description
Deletion of Existing Data	Specifies what action to take on existing data.: select **Replace All** or **Replace Matching** from the drop-down list. ■ Select **Replace All** from the drop-down list, and select **Yes** or **No** in the **Preserve Data Entered via Web Form** option. ■ Select **Replace Matching** from the drop-down list, and select **Yes** or **No** in the **Ignore Currency Dimension** option. The four possible combinations specify the four available deletion policies: ■ Replace all, preserve form data—Data is deleted from all the crossings that you have specified as eligible to receive data. This does not include data that was entered through forms or stored computed values of driver formulas. In each case where data is loaded to a crossing that already has form-entered data or stored driver-formula values, the result is additive. It is possible that data will be deleted from some crossings that do not receive data in the load operation. ■ Replace all, do not preserve form data—Data is deleted from all the crossings that you have specified as eligible to receive data, without exception. Data that was loaded previously, data that was entered through forms, and stored computed values of driver formulas are all deleted. It is possible that data will be deleted from some crossings that do not receive data in the load operation. ■ Replace matching, ignore currency dimension—Data is deleted only from crossings that match a crossing in the data that you are loading. A crossing in the new data matches a crossing in the existing data if the two records match member-for-member in every dimension except Source and Currency. The Source and Currency members can match or not. ■ Replace matching, do not ignore currency dimension—Data is deleted only from crossings that match a crossing in the data that you are loading. A crossing in the new data matches a crossing in the existing data if either of the following conditions is met: □ The two crossings match member-for-member in every dimension except Source. The Source member can match or not. □ The existing record was created through form data entry and the two crossings match member-for-member in every dimension except Source and Currency. The Source and Currency members can match or not.
Environment	(Optional) Any environment that is defined in your sas-environment.xml file. If you leave this field empty, then the environment “default” is used.

7 Click **OK** to save your changes and close the Properties window.

8 In the Job Editor window, click **Run**.

9 When the job displays as completed in the Status column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

Loading Base Financial Data into the Data Mart by Using a SAS Macro

To load base financial data by using a SAS macro, complete the following steps:

1 Create a SAS data set that specifies the member combinations for which you want to load data.

2 Run the etlldfct.sas macro file.

When using the etlldfct.sas macro to load exchange rates into the Data Mart, note the following:

- Detailed instructions on using the etlldfct.sas macro are inside the macro file, which is on the data tier server.
- On a Windows server, the etlldfct.sas macro file is at the following location: `!SASROOT\finance\sasmacro`
- On a UNIX server, the etlldfct.sas macro file is at the following location: `!SASROOT/sasautos`

Note: To run the macro, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

Which Records Are Loaded?

A staging area record that contains base financial data is loaded only if the following conditions are met:

- The record contains a member for each dimension type (except Source) that is used by the target cycle.
- The record contains a member for no dimension type that is not used by the target cycle.
- Within each dimension type that is used by the record and the target cycle, the member in the record belongs to the dimension that is used by the target cycle.
- The record belongs to the subset of records containing a member that is defined in the wizard, job, or SAS macro that loads the data. In other words, for each dimension type for which one or more members have been defined in the wizard, job, or SAS macro, one of those members is in the record.

Checking for Errors

After you run a job that uses the Load Base Data transformation, review the log. If there were errors, then the job is terminated and the log lists the location of an

HTML error report. If the SAS macro is terminated, then the log lists the location of an HTML error report. Error reports for the job, the macro, and the Load New Data wizard are all available in SAS Financial Management Studio from the **History** page of the Properties window for the target cycle.

An error report is produced if any record violates any one of the following constraints:

- In the GL_TRANSACTION_SUM table, INITIATING_INTERNAL_ORG_ID must not be ALL or EXT.
- In the GL_TRANSACTION_SUM table, INITIATING_INTERNAL_ORG_ID and AFFECTED_INTERNAL_ORG_ID (Trader) must be different.
- If the account that is specified in the GL Account ID column of GL_TRANSACTION_SUM has a value of *N* for Intercompany Account Flag in the GL_ACCOUNT table, then the value of AFFECTED_INTERNAL_ORG_ID (Trader) in GL_TRANSACTION_SUM must be EXT. (This constraint applies only if you load the data into a SAS Financial Management cycle for which the “Non-intercompany accounts must be associated with the external trading member” property is set.)
- If the account that is specified in the GL Account ID column of GL_TRANSACTION_SUM has a value of *Y* for Intercompany Account Flag in the GL_ACCOUNT table, then the value of AFFECTED_INTERNAL_ORG_ID (Trader) in GL_TRANSACTION_SUM must not be EXT. (This constraint applies only if you load the data into a SAS Financial Management cycle for which the “Intercompany accounts must be associated with an intercompany trading partner” property is set.)
- Member IDs are also validated when they are imported into the Data Mart. A data record that has an ID value that does not also exist in the corresponding member table appears in the error report as follows:

Validation Errors
[NOTE: *stage ID* message indicates that a member ID value in the stage GL_TRANSACTION_SUM and/or stage GL_JRNL_DETAILS table(s) does not exist in the corresponding stage member table.]

Account_code	Analysis_code	Currency_code	Intorg_code	Source_code	Time_code	Trader_code	Value
stage ID A201	*stage ID* ACTUAL	*stage ID* EUR	*stage ID* ALDOMOVAR	*stage ID* Base	*stage ID* APR2004	*stage ID* EXT	-8.00000

Deleting Facts from a SAS Financial Management Cycle

The fm_3000_delete_fact_data job deletes facts from a SAS Financial Management cycle.

To delete facts from a cycle by using the fm_3000_delete_fact_data job, complete the following steps.

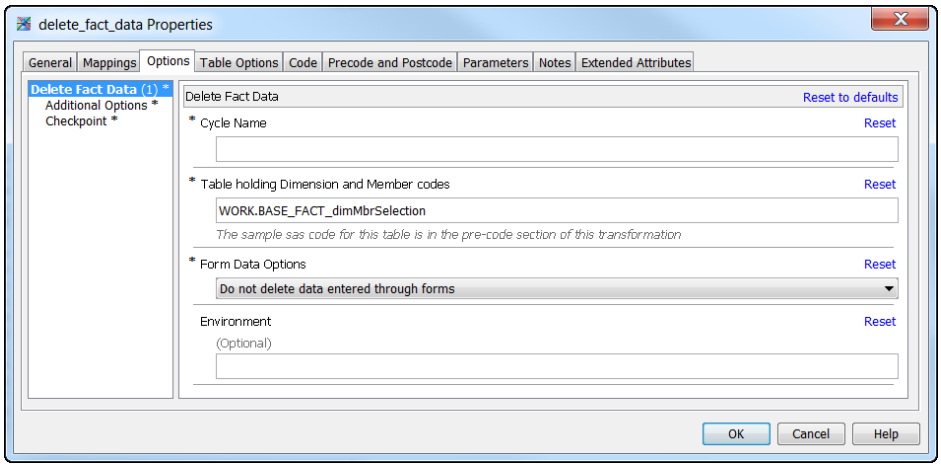
1. In SAS Data Integration Studio, select the **Folders** tab.
2. In the **Folders** tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.
3. Make a copy of the fm_3000_delete_fact_data job.

Note: It is a good practice to create and maintain a separate, appropriately named job for each set of option values. Changing the value of a job's options is possible, but likely to generate confusion.

4 Double-click **fm_3000_delete_fact_data** in the list of jobs.

5 In the Job Editor window, select the **delete_fact_data** transformation and select **Properties** from the pop-up menu. The delete_fact_data Properties window is displayed.

6 Select the **Options** tab.



7 Enter values for the following options:

Option	Description
Cycle Name	Name of the SAS Financial Management cycle from which you want to delete fact data.
Table holding Dimension and Member Codes	Name of the data set that specifies the dimension and member combination for which you want to delete data.
Form Data Options	Specifies whether to **Delete data entered through forms** or **Do not delete data entered through forms**.
Environment	(Optional) Environment that is defined in your EnvironmentFactory.xml file (for authentication purposes). If you leave this field empty, then the default environment is used. (The default value is `default`).

Note: The Precode region on the **Precode and Postcode** tab contains sample code that builds a SAS data set with the required layout. By default, the job uses the SAS data set that is built by the Precode program. Select the **Precode and Postcode** tab, and complete one of the following tasks:

- Ensure that the **Precode** check box is selected. Ensure that the name of the SAS data set that is specified for this option matches the name of the

SAS data set that is built by the precode. Modify the precode to build the table that you need. In this case, the precode runs before the job, and then the job uses the SAS data set that the precode builds.

- Ensure that the **Precode** check box is not selected. Build the SAS data set that you need to use by using another method other than the precode. Ensure that the name of the SAS data set that is specified for this option matches the name of the SAS data set that you built for this purpose.

8 Click **OK** to save your changes and to close the Properties window.

9 In the Job Editor window, click **Run**.

10 When the job displays as completed in the Status column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administrator's Guide*.

18

Exporting Financial Accounting Data

Overview

This chapter describes the following topics

- exporting accounting data
- using the Export Data Records Wizard to export accounting data
- using the Export Model Data Job to export accounting data
- using a SAS macro to export accounting data
- Viewing the details of the result
- possible obstacles to exporting data
- checking for errors

About Exporting Accounting Data

You can export data from a selected model in SAS Financial Management to a designated SAS library. The target library can be the StageFM library of staging tables or any other library that you set up to receive exported data.

If the target library is StageFM, then the exported data is placed in the following tables:

- GL_TRANSACTION_SUM
- GL_JRNL

- GL_JRNL_DETAILS

If you use any other target library, then the exported data is placed in copies of these staging tables that have been put in the target library.

From the target library, you can make the exported accounting data available to other products, such as SAS Web Report Studio.

In general, you should not export data from a model to the staging tables and then load it into a cycle. That procedure works, but you can achieve the same result more easily with the Load Model Data wizard in SAS Financial Management Studio.

There are three ways to export data from a selected model:

- Use the Export Data Records wizard in SAS Financial Management Studio.
- Use the fm_2000_export_model_data job.
- Use a SAS macro.

Exporting Accounting Data by Using the Export Data Records Wizard

To export data by using the Export Data Records wizard:

1. In SAS Financial Management Studio, open the cycle from which you want to export data.
2. In the Models workspace, select the source model.
3. Select **Export Data Records** to launch the Export Data Records wizard.
4. Work through the wizard, consulting the online Help for the individual wizard pages as necessary.

Export Accounting Data by Using the Export Model Data Job

The fm_2000_export_model_data job exports accounting data.

To export accounting data by using the fm_2000_export_model_data job, complete the following steps:.

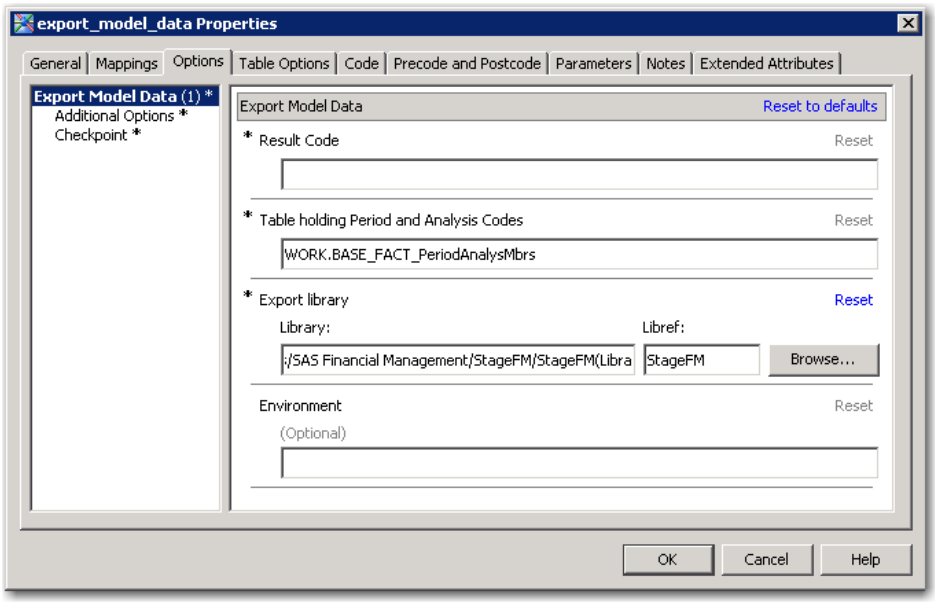
1. In SAS Data Integration Studio, select the **Folders** tab.
2. In the **Folders** tree, select **Products** ▸ **SAS Financial Managment** ▸ **5.5 Jobs**.
3. Make a copy of the fm_2000_export_model_data job.

   Note: It is a good practice to create and maintain a separate, appropriately named job for each set of option values. Changing the value of a job's options is possible, but likely to generate confusion.

4 Double-click **fm_2000_export_model_data** in the list of jobs. The job is displayed in the Job Editor window.

5 Right-click the export_model_data transformation and select **Properties** from the pop-up menu. The export_model data Properties window is displayed.

6 Select the **Options** tab.



7 Enter values for the following options:

Option	Description
Result Code	Code of the model that is the source of the data
Table holding Period and Analysis Codes	Name of a SAS data set that specifies the member combinations to export data for. The **Precode** region on the **Precode and Postcode** tab contains sample code that builds a SAS data set with the required layout. By default, the job uses the SAS data set that is built by the Precode program. Do one of the following: ■ On the **Precode and Postcode** tab, make sure that the **Precode** check box is selected. Make sure that the name of the SAS data set that is specified for this option matches the name of the SAS data set that is built by the precode. Modify the precode to build the table that you need. In this case, the precode runs before the job, and then the job uses the SAS data set that the precode builds. ■ On the **Precode and Postcode** tab, make sure that the **Precode** check box is not selected. Build the SAS data set that you need by using a means other than the precode. Make sure that the name of the SAS data set that is specified for this option matches the name of the SAS data set that you built for this purpose.

Option	Description
Export Library	SAS library that you are exporting the data to. Click **Browse** to select the target library. If you select a library other than **StageFM**, then make sure that the selected library satisfies the following conditions: ■ It contains copies of the GL_TRANSACTION_SUM, GL_JRNL, and GL_JRNL_DETAILS tables. ■ The Solutions Host User has operating system Read and Write access to it.
Environment	(Optional) Any environment that is defined in your sas-environment.xml file. If you leave this field empty, then the environment "default" is used.

8 Click **OK** to save your changes and close the Properties window.

9 Select **File** ▸ **Save**.

10 In the Job Editor window, click **Run**.

11 When the job displays as completed in the **Status** tab of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Note: To run the job, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

Exporting Accounting Data by Using a SAS Macro

To export accounting data by using a SAS macro, complete the following steps:

1 Create a SAS data set that specifies the combinations of analysis members and time periods for which you want to export accounting data.

2 Run the etlfctxp.sas macro file.

When using the etlfctxp.sas macro to export accounting data, note the following:

- Detailed instructions on using the etlfctxp.sas macro are inside the macro file, which is located on the data tier server.
- On a Windows server, the etlfctxp.sas macro file is at the following location: `SASROOT\finance\sasmacro`
- On a UNIX server, the etlfctxp.sas macro file is at the following location: `SASROOT/sasautos`

Note: To run the macro, SAS managed servers must be running on the middle-tier server. For more information about the managed servers, see the *SAS Financial Management: System Administration Guide*.

Details of the Result

The exported data is appended to whatever data is already in the target tables. If the target tables contain data that you do not want to mix with the data that you are exporting, then you must delete the data from the target tables before you begin the export process. To delete data from the target tables, write and run a suitable SAS program.

For each specified combination of a time period and an analysis member, the following data is exported:

- All data that is stored in the cycle that the model belongs to. This includes data that is associated with the following members of the Source hierarchy:
  - Base
  - BaseJourn
  - BaseForm
- All manual adjustments and all adjustments that are generated by adjustment rules that are part of the model. This includes data that is associated with the following members of the Source hierarchy:
  - Manual
  - Bal
  - Alloc
  - Reclass
  - CPO

Data that is associated with the BaseJourn member of the Source hierarchy is exported to the GL_JRNL and GL_JRNL_DETAILS tables or to copies of these tables that you place in another target library. All other exported data is exported to the GL_TRANSACTION_SUM table or to the copy of this table that you place in another target library.

Many numbers that you might see in a SAS Financial Management report that is based on the selected model are not exported. Numbers that are not exported include the following:

- elimination adjustments
- the computed values of accounts that belong to the Retained Earnings and CTA account types
- the computed values of hierarchical roll-ups
- the computed values of formulas

Possible Obstacles to Exporting Accounting Data

The Export Data Records wizard, the fm_2000_export_model_data job, and the etlfctxp.sas macro file can encounter various obstacles that prevent them from successfully exporting data.

Possible obstacles include the following:

- The Solutions Host User does not have operating system Read and Write access to the target data library.
- A target table does not exist. If the target data library is the staging area, this can happen if a table was accidentally deleted. For a target data library other than the staging area, this can happen if you neglected to copy one of the necessary tables into the target library.
- A column that represents a dimension type that is used by the data is either misnamed or missing from a target table. This can happen if the column was not added correctly when the dimension type was created.
- The DIMENSION_TYPE table contains an incorrect record for one of the dimension types that are used by the data. This can happen if an incorrect value was placed in the record when it was created.
- One of the target tables is open and locked. This can happen if someone is working with the table.
- The new destination library must contain copies of all of the tables that are needed to receive the exported data. These tables include the following:
  - Dimension-type-specific tables for each dimension type with which you are working.
  - For the Account dimension type, you need a copy of each of the following five tables: GL_ACCOUNT, GL_ACCOUNT_ASSOC_TYPE, GL_ACCOUNT_ASSOC, GL_ACCOUNT_NLS, and SOURCE_GL_ACCOUNT.

If any one of these obstacles is encountered, an appropriate message is displayed.

Checking for Errors

After you run a job that uses the export_model_data transformation, review the log. If there were errors, then the job is terminated and the log lists the location of an HTML report that contains information about the errors. If the SAS macro is terminated, then the log lists the location of an HTML report. Error reports for the job, the macro, and the Export Data Records wizard are all available in SAS Financial Management Studio. These error reports are accessible from the **History** page of the Properties window for the source model.

19

Loading Supplemental Schedule Detail and Fact Tables

Overview

This chapter describes the following topics:

- supplemental schedule detail and fact tables
- loading the supplemental schedule detail tables
- loading the supplemental schedule fact tables

About Supplemental Schedule Detail and Fact Tables

A supplemental schedule is an additional table that you can add to enable users to reference detailed information outside the model to use in reports and forms.

Loading the Supplemental Schedule Details and Facts into the Staging Tables

The following two staging tables are defined in the StageFM library for supplemental schedule details and facts outside the model:

- SUPP_SCHEDULE_DETAIL—Contains the details for a supplemental schedule table.
- SUPP_SCHEDULE_FACT—Contains the facts for a supplemental schedule table.

To load a supplemental schedule table, you must write and run a job that loads supplement schedule data from its source into the table.

Before you write a job to load data into a supplemental schedule table, review the column structure of the table to ensure that the job that you write places the correct data in the correct columns.

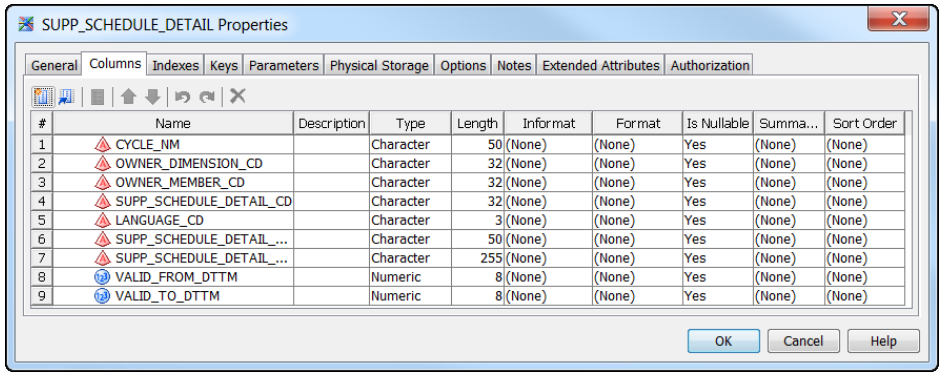
To view the column structure of the SUPP_SCHEDULE_DETAIL table, complete the following steps:

1 In SAS Data Integration Studio, select the **Folder** tab.

2 In the Folders tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.

3 Double-click **SUPP_SCHEDULE_DETAIL** in the list of tables. The SUPP_SCHEDULE_DETAIL Properties window is displayed.

4 Click the **Columns** tab to view the column structure of the staging table.



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	CYCLE_NM		Character	50	(None)	(None)	Yes	(None)	(None)
2	OWNER_DIMENSION_CD		Character	32	(None)	(None)	Yes	(None)	(None)
3	OWNER_MEMBER_CD		Character	32	(None)	(None)	Yes	(None)	(None)
4	SUPP_SCHEDULE_DETAIL_CD		Character	32	(None)	(None)	Yes	(None)	(None)
5	LANGUAGE_CD		Character	3	(None)	(None)	Yes	(None)	(None)
6	SUPP_SCHEDULE_DETAIL_...		Character	50	(None)	(None)	Yes	(None)	(None)
7	SUPP_SCHEDULE_DETAIL_...		Character	255	(None)	(None)	Yes	(None)	(None)
8	VALID_FROM_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)
9	VALID_TO_DTTM		Numeric	8	(None)	(None)	Yes	(None)	(None)

The SUPP_SCHEDULE_DETAIL table contains the following columns:

Column	Description
CYCLE_NM	Name of the cycle that the supplemental schedule uses.
OWNER_DIMENSION_CD	Dimension code for the owner of the detail record.
OWNER_MEMBER_CD	Member code for the owner of the detail record.
SUPP_SCHEDULE_DETAIL_CD	Code that uniquely identifies the detail.
LANGUAGE_CD	Code that identifies a language and locale. An example is "en" for English.
SUPP_SCHEDULE_DETAIL_NM	Name for the detail.
SUPP_SCHEDULE_DETAIL_DESC	Description for the detail.
VALID_FROM_DTTM	Moment that begins the time period during which a row of data is valid.

Column	Description
VALID_TO_DTTM	Moment that ends the time period during which a row of data is valid.

5 Click **OK** to close the Properties window.

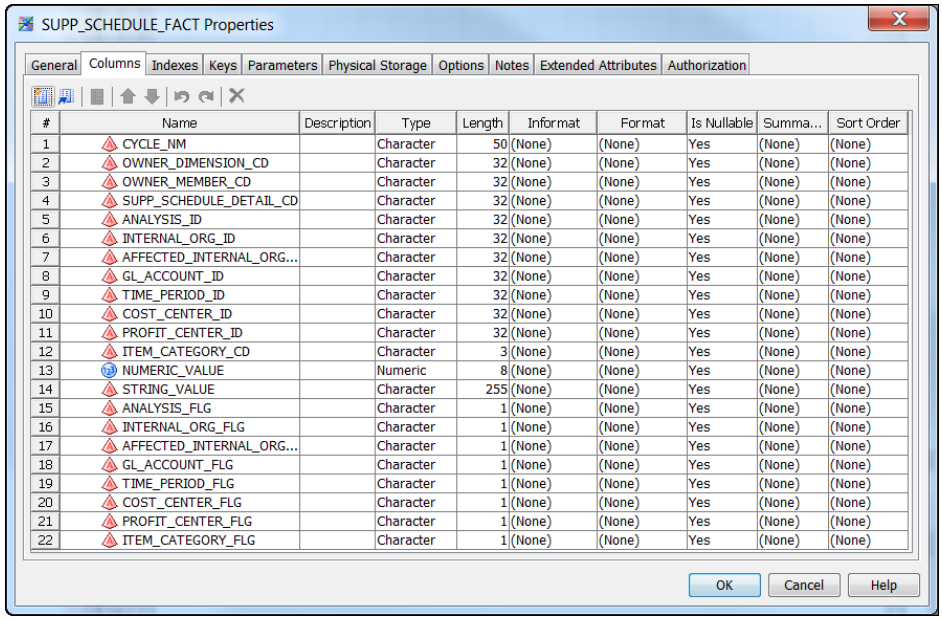
To view the column structure of the SUPP_SCHEDULE_FACT table, complete the following steps:

1 In SAS Data Integration Studio, select the **Folder** tab.

2 In the Folders tree, select **Products** ▸ **SAS Financial Management** ▸ **StageFM**.

3 Double-click **SUPP_SCHEDULE_FACT** in the list of tables. The SUPP_SCHEDULE_FACT Properties window is displayed.

4 Click the **Columns** tab to view the column structure of the staging table.



#	Name	Description	Type	Length	Informat	Format	Is Nullable	Summa...	Sort Order
1	CYCLE_NM		Character	50	(None)	(None)	Yes	(None)	(None)
2	OWNER_DIMENSION_CD		Character	32	(None)	(None)	Yes	(None)	(None)
3	OWNER_MEMBER_CD		Character	32	(None)	(None)	Yes	(None)	(None)
4	SUPP_SCHEDULE_DETAIL_CD		Character	32	(None)	(None)	Yes	(None)	(None)
5	ANALYSIS_ID		Character	32	(None)	(None)	Yes	(None)	(None)
6	INTERNAL_ORG_ID		Character	32	(None)	(None)	Yes	(None)	(None)
7	AFFECTED_INTERNAL_ORG...		Character	32	(None)	(None)	Yes	(None)	(None)
8	GL_ACCOUNT_ID		Character	32	(None)	(None)	Yes	(None)	(None)
9	TIME_PERIOD_ID		Character	32	(None)	(None)	Yes	(None)	(None)
10	COST_CENTER_ID		Character	32	(None)	(None)	Yes	(None)	(None)
11	PROFIT_CENTER_ID		Character	32	(None)	(None)	Yes	(None)	(None)
12	ITEM_CATEGORY_CD		Character	3	(None)	(None)	Yes	(None)	(None)
13	NUMERIC_VALUE		Numeric	8	(None)	(None)	Yes	(None)	(None)
14	STRING_VALUE		Character	255	(None)	(None)	Yes	(None)	(None)
15	ANALYSIS_FLG		Character	1	(None)	(None)	Yes	(None)	(None)
16	INTERNAL_ORG_FLG		Character	1	(None)	(None)	Yes	(None)	(None)
17	AFFECTED_INTERNAL_ORG...		Character	1	(None)	(None)	Yes	(None)	(None)
18	GL_ACCOUNT_FLG		Character	1	(None)	(None)	Yes	(None)	(None)
19	TIME_PERIOD_FLG		Character	1	(None)	(None)	Yes	(None)	(None)
20	COST_CENTER_FLG		Character	1	(None)	(None)	Yes	(None)	(None)
21	PROFIT_CENTER_FLG		Character	1	(None)	(None)	Yes	(None)	(None)
22	ITEM_CATEGORY_FLG		Character	1	(None)	(None)	Yes	(None)	(None)

The SUPP_SCHEDULE_FACT table contains the following columns:

Column	Description
CYCLE_NM	Name of the cycle that the supplemental schedule uses.
OWNER_DIMENSION_CD	Dimension code for the owner of the detail record.
OWNER_MEMBER_CD	Member code for the owner of the detail record.
SUPP_SCHEDULE_DETAIL_CD	Code that uniquely identifies the detail.

Column	Description
ANALYSIS_ID	Code that identifies a language and locale. An example is "en" for English.
INTERNAL_ORG_ID	Name for the detail.
AFFECTED_ INTERNAL_ORG_ID	Description for the detail.
GL_ACCOUNT_ID	Moment that begins the time period during which a row of data is valid..
FINISH TABLE	Find definition.

Note: For a complete list of the columns in the SUPP_SCHEDULE_FACT table, see the *SAS Financial Management: Data Model Reference*.

5 Click **OK** to close the Properties window.

To load supplemental schedule detail and fact data into the staging tables, complete the following steps:

1 Create a SAS Data Integration Studio job that loads data from its source into the tables for each of the supplementary schedule tables in the StageFM library.

2 Click **OK** to save the job.

3 Run the job and validate that the table loaded successfully.

Note: For detailed information about creating and working with SAS Data Integration Studio jobs to extract data from a source table and load it into a target table, see the *SAS Data Integration User's Guide*.

Loading the Supplemental Schedule Details and Facts into the Data Mart

The following two jobs load supplemental schedule detail and fact data into the Data Mart:

- fm_2200_load_ss_detail—Loads data in the SUPP_SCHEDULE_DETAIL table.
- fm_2210_load_ss_fact—Loads data in the SUPP_SCHEDULE_FACT table.

To load supplemental schedule detail data from the SUPP_SCHEDULE_DETAIL table into the Data Mart, complete the following steps:

1 In SAS Data Integration Studio, select the **Folder** tab.

2 In the Folders tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.

3 Make a copy of the fm_2200_load_ss_detail_job.

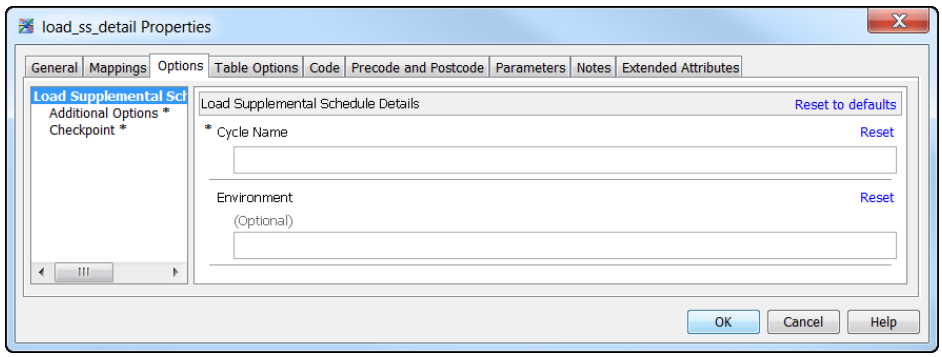
Note: It is a good idea to create and maintain a separate, appropriately named job for each set of option values. Changing the option values of a job occasionally is possible, but it is likely to generate confusion.

4 Double-click to select the job.

5 In the Job Editor window, right-click the load_ss_fact transformation, and select **Properties** from the pop-up menu.

6 Click the **Options** tab.

Display 19.1 *load_ss_detail Properties Window — Options View*



7 Enter values for the following options:

Option	Description
Cycle Name	Name of the cycle to which the supplemental schedule detail table applies. For more information about cycles, see the *SAS Financial Management: User's Guide*.
Environment	(Optional) Name of the middle-tier environment (for authentication purposes). The default value is default.

8 Click **OK** to save your changes and close the Properties window.

9 Select **File** ▸ **Save**.

10 In the Job Editor window, click **Run**.

11 When the job displays as completed in the Status column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

To load supplemental schedule facts from the SUPP_SCHEDULE_FACT table into the Data Mart, complete the following steps:

1 In SAS Data Integration Studio, select the **Folder** tab.

2 In the Folders tree, select **Products** ▸ **SAS Financial Management** ▸ **5.5 Jobs**.

3 Make a copy of the fm_2210_load_ss_fact job.

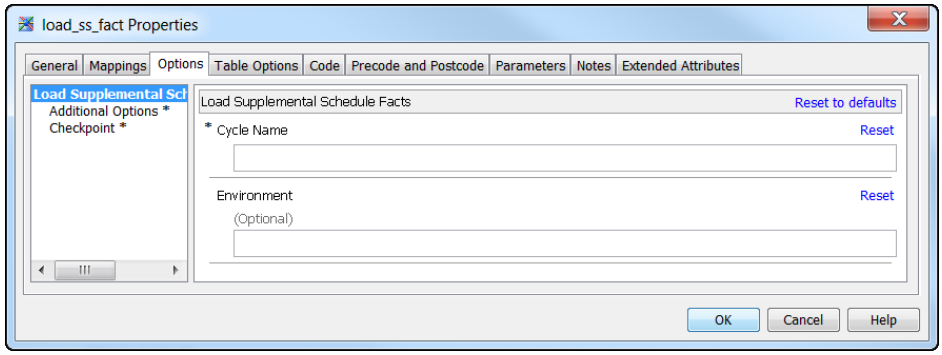
Note: It is a good idea to create and maintain a separate, appropriately named job for each set of option values. Changing the option values of a job occasionally is possible, but it is likely to generate confusion.

4 Double-click to select the job.

5 In the Job Editor window, right-click the load_ss_fact transformation, and select **Properties** from the pop-up menu.

6 In the Properties window, select the **Options** tab.

Display 19.2 *load_ss_fact Properties Window — Options View*



7 Enter values for the following options:

Option	Description
Cycle Name	Name of the cycle to which the supplemental schedule fact table applies. For information about cycles, see the *SAS Financial Management: User's Guide*.
Environment	(Optional) Name of the middle-tier environment (for authentication purposes). The default value is default.

8 Click **OK** to save the job.

9 In the Job Editor window, click **Run**.

10 When the job displays as completed in the Status column of the Details pane, select the **Log** tab to review the log. The log lists the location of an HTML report of the results.

Appendix 1

The Conform Area

Overview

This appendix describes the following topics:

- the conform area
- copying tables to the conform area

The Conform Area

Data that passes through the staging area can go from the staging area to the Data Mart directly or by way of an intermediate location known as the *conform* area.

The conform area and the SAS Financial Management staging area are concatenated to constitute the CONFORM library. Jobs that load data into the Data Mart refer to input tables in the CONFORM library. By default, the conform area points to the same location as the SAS Financial Management staging area:

```
..Lev1\SASApp\Data\FinancialManagement\StageFM
```

Creating a Separate Conform Area

To create a separate conform area, prepend the following path in the LIBNAME statement:

```
..\Lev1\SASApp\Data\FinancialManagement\ConformedDataMart
```

Note: The ConformedDataMart folder path must be prepended to the stageFM folder path. Do not be misled by the name of the conform area directory. The conform area is not a destination data mart.

The following LIBNAME statement is an example of creating a separate conform area:

```
LIBNAME Conform BASE
("C:\SAS\Config\Lev1\SASApp\Data\FinancialManagement\ConformedDataMart"
"C:\SAS\Config\Lev1\SASApp\Data\FinancialManagement\StageFM");
```

Copying Tables to the Conform Area

To copy tables to the conform area, complete the following steps:

1. In SAS Data Integration Studio, click the **Transformations** tab.
2. In the Transformations tree, expand the **SAS Financial Management**.
3. Use the copy_all_stagefm_tables_to_conform_library and copy_stagefm_table_to_conform_library transformations to write jobs that copy tables to the conform area

When copying tables to the conform area, note the following:

- You can recopy the tables at any time.
- Once you copy tables to the conform area, the SAS Financial Management Data Mart is loaded from the most recently copied versions of the tables.
- If you never copy these tables to the conform area, then the SAS Financial Management Data Mart is loaded from the tables in the SAS Financial Management staging area.

Index

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

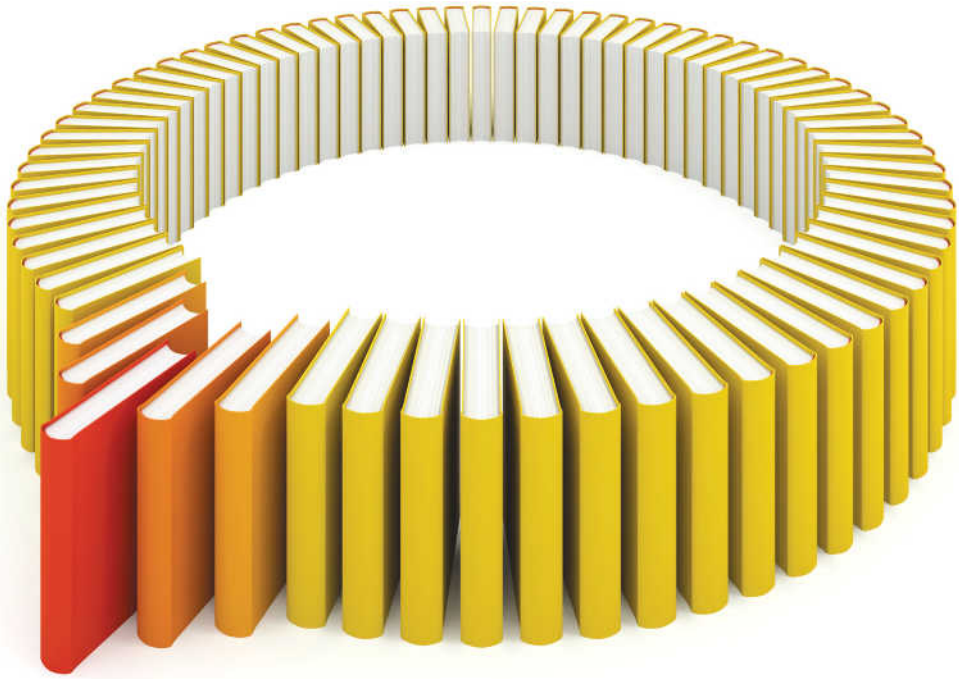
R

S

T

U

Y



Gain Greater Insight into Your SAS® Software with SAS Books.

Discover all that you need on your journey to knowledge and empowerment.

support.sas.com/bookstore
for additional books and resources.