SAS® Data Integration Studio 3.3

User's Guide

The Power to Know®

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2006. *SAS® Data Integration Studio 3.3: User's Guide*. Cary, NC: SAS Institute Inc.

SAS® Data Integration Studio 3.3: User's Guide

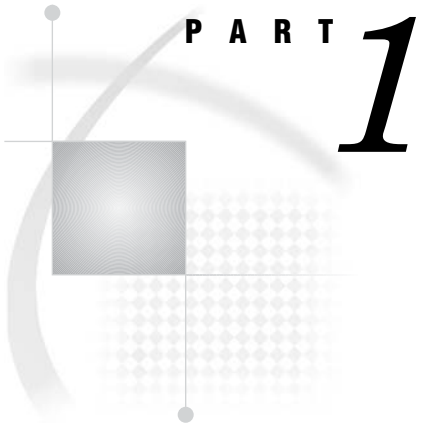
ISBN-13: 978-1-59047-869-1
ISBN-10: 1-59047-869-X

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

1st printing, March 2006

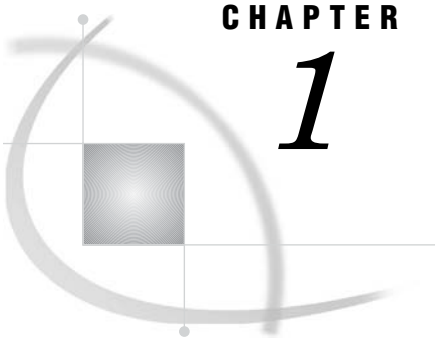
SAS Publishing provides a complete selection of books and electronic products to help customers use SAS software to its fullest potential. For more information about our e-books, e-learning products, CDs, and hard-copy books, visit the SAS Publishing Web site at **support.sas.com/pubs** or call 1-800-727-3228.

Contents



PART 1

Introduction



CHAPTER

1

Using This Manual

Purpose of This Manual

This manual explains how to use SAS Data Integration Studio to do the following tasks:

- specify metadata for data sources, such as tables in an operational system
- specify metadata for data targets, such as tables in a data warehouse
- create jobs that specify how data is extracted, transformed, and loaded from sources to targets

This manual also summarizes how to set up servers, libraries, and other resources that SAS Data Integration Studio requires. A data warehouse for a fictional company, Orion Star Sports & Outdoors, is used to illustrate these tasks.

Intended Audience for This Manual

This manual is intended for people who assume the following roles:

- SAS Data Integration Studio user—a person who uses SAS Data Integration Studio software to consolidate and manage enterprise data from a variety of source systems, applications, and technologies.
- SAS Data Integration Studio metadata administrator—a person who uses SAS Management Console software to maintain the metadata for servers, users, and other global resources that are required by SAS Data Integration Studio.

This manual is not intended for server administrators—people who install and maintain server hardware or software. However, some SAS Data Integration Studio tasks depend on tasks that the server administrator performs. A common scenario for SAS Data Integration Studio projects is as follows:

- A server administrator installs and starts servers. For details about maintaining these servers, administrators should see the documentation that came with the servers. See also the servers sections in the administering SAS Data Integration Studio chapter and the connecting to common data sources chapter in the *SAS Intelligence Platform: Administration Guide*.

- A metadata administrator uses SAS Management Console to define metadata for servers, users, libraries, and other global resources, as described in Chapter 6, "Main Tasks for Administrators," on page 53. For details about maintaining global metadata, see the online Help for SAS Management Console. See also the *SAS Management Console: User's Guide* and the assigning libraries and connecting to common data sources chapters in the *SAS Intelligence Platform: Administration Guide*.
- SAS Data Integration Studio users create process flows that consolidate and manage enterprise data from a variety of source systems, applications, and technologies, as described in "Main Tasks for Creating Process Flows" on page 96. Users are usually told which servers, user identities, libraries, and other global resources to use.

Quick Start with SAS Data Integration Studio

Administrators who want to begin work immediately should read Chapter 6, "Main Tasks for Administrators," on page 53. Users who want to begin work immediately should read Chapter 7, "Main Tasks for Users," on page 91.

SAS Data Integration Studio Online Help

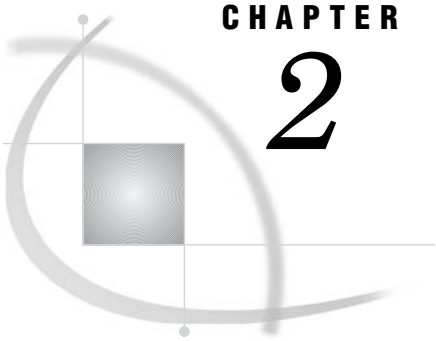
This manual is a companion to the online Help for SAS Data Integration Studio. The Help describes all windows in SAS Data Integration Studio, and it summarizes the main tasks that you can perform with the software. The Help includes examples for all source designer wizards, all target designer wizards, and all transformation templates in the Process Library tree. The Help also includes a What's New topic and a set of Usage Note topics for the current version of the software.

Follow these steps to display the main Help window for SAS Data Integration Studio.

1. Start SAS Data Integration Studio as described in "Starting SAS Data Integration Studio" on page 93.
2. From the menu bar, select **Help ▶ Contents**. The main Help window displays.

To display the Help for an active window or tab, click its **Help** button. If the window or tab does not have a **Help** button, press the **F1** key.

To search for topics about concepts or features that are identified by specific words, such as "application server," display the main Help window, then click the **Search** tab (magnifying glass icon). Enter the text to be found and press the **Enter** key.



CHAPTER

2

Introduction to SAS Data Integration Studio

The SAS Intelligence Platform

About the Platform Tiers

SAS Data Integration Studio is one component in the SAS Intelligence Platform, which is a comprehensive, end-to-end infrastructure for creating, managing, and distributing enterprise intelligence. The platform includes tools and interfaces that enable you to do the following:

- extract data from a variety of operational data sources on multiple platforms and build a data collection that integrates the extracted data
- store large volumes of data efficiently and in a variety of formats
- give business users at all levels the ability to explore data from the warehouse in a Web browser, to perform simple query and reporting functions, and to view up-to-date results of complex analyses
- use high-end analytic techniques to provide capabilities such as predictive and descriptive modeling, forecasting, optimization, simulation, and experimental design
- centrally control the accuracy and consistency of enterprise data

For more information about the SAS Intelligence Platform, see the *SAS Intelligence Platform: Overview*.

What Is SAS Data Integration Studio?

SAS Data Integration Studio is a visual design tool that enables you to consolidate and manage enterprise data from a variety of source systems, applications, and technologies. This software enables you to create process flows that accomplish the following tasks:

- extract, transform, and load (ETL) data for use in data warehouses and data marts
- cleanse, migrate, synchronize, replicate, and promote data for applications and business services

SAS Data Integration Studio enables you to integrate information from any platform that is accessible to SAS and from any format that is accessible to SAS.

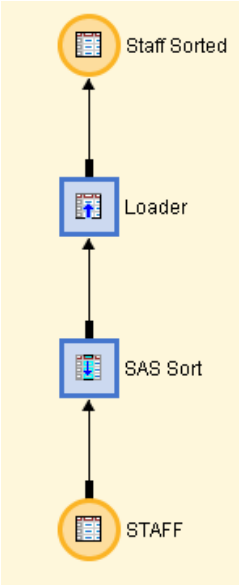
Note: SAS Data Integration Studio was formerly named SAS ETL Studio. △

Important Concepts

Process Flows and Jobs

In SAS Data Integration Studio, a job is a metadata object that specifies processes that create output. Each job generates or retrieves SAS code that reads data sources and creates data targets in physical storage. To generate code for a job, you create a process flow diagram that specifies the sequence of each source, target, and process in the job. For example, the following display shows the process flow for a job that will read data from a source table named STAFF, sort the data, then write the sorted data to a target table named Staff Sorted.

Display 2.1 Process Flow Diagram for a Job



Each process in the flow is specified by a metadata object that is called a transformation. In the previous figure, SAS Sort and Loader are transformations. A

transformation specifies how to extract data, transform data, or load data into data stores. Each transformation generates or retrieves SAS code. In most cases, you will want SAS Data Integration Studio to generate code for transformations and jobs, but you can specify user-written code for any transformation in a job, or for the entire job.

How Jobs Are Executed

In SAS Data Integration Studio, you can execute a job in the following ways:

- use the **`Submit Job`** option to submit the job for interactive execution
- use the **`Deploy for Scheduling`** option to generate code for the job and save it to a file; the job can be executed later in batch mode
- use the **`Stored Process`** option to generate a stored process for the job and save it to a file; the job can be executed later in batch mode by a stored process server

Identifying the Server That Executes a Job

In SAS Open Metadata Architecture applications such as SAS Data Integration Studio, a SAS Application Server is a metadata object that can provide access to several servers, libraries, schemas, directories, and other resources. An administrator typically defines this object and then tells the SAS Data Integration Studio user which object to select as the default.

Behind the scenes, when you submit a SAS Data Integration Studio job for execution, it is submitted to a SAS Workspace Server component of the relevant SAS Application Server. The relevant SAS Application Server is one of the following:

- the default server that is specified on the **`SAS Server`** tab in the Options window in SAS Data Integration Studio
- the SAS Application Server to which a job is deployed with the **`Deploy for Scheduling`** option

It is important for administrators to know which SAS Workspace Server or servers will execute a job in order to do the following tasks:

- store data where it can be accessed efficiently by the transformations in a SAS Data Integration Studio job, as described in "Supporting Multi-Tier (N-Tier) Environments" on page 64
- locate the SAS Work library where the job's intermediate files are stored by default
- specify SAS options that you want to apply to all jobs that are executed on a given server, as described in "Setting SAS Options for Jobs and Transformations" on page 189

To identify the SAS Workspace Server or servers that will execute a SAS Data Integration Studio job, administrators can use SAS Management Console to examine the metadata for the relevant SAS Application Server.

Intermediate Files for Jobs

Transformations in a SAS Data Integration Studio job can produce three kinds of intermediate files:

- procedure utility files that are created by the SORT and SUMMARY procedures, if these procedures are used in the transformation
- transformation temporary files that are created by the transformation as it is working

- transformation temporary output tables that are created by the transformation when it produces its result; the output for a transformation becomes the input to the next transformation in the flow

For example, suppose that you executed the job with the process flow that is shown in Display 2.1 on page 6. When the Sort transformation is finished, it creates a temporary output table. The default name for the output table is a two-level name with the Work libref and a generated member name, such as **work.W54KFYQY**. This output table becomes the input to the next step in the process flow.

By default, procedure utility files, transformation temporary files, and transformation temporary output tables are created in the Work library. You can use the WORK invocation option to force all intermediate files to a specified location, or you can use the UTILLOC invocation option to force only utility files to a separate location.

Knowledge of intermediate files helps you to do the following tasks:

- view or analyze the output tables for a transformation, and verify that the output is correct, as described in "Analyzing Transformation Output Tables" on page 192
- manage disk space usage for intermediate files, as described in "Managing Disk Space Use for Intermediate Files" on page 184

How Are Intermediate Files Deleted?

Procedure utility files are deleted by the SAS procedure that created them. Any transformation temporary files are deleted by the transformation that created them.

When a SAS Data Integration Studio job is executed in batch, transformation temporary output tables are deleted when the process flow ends or the current server session ends.

When a job is executed interactively in SAS Data Integration Studio, the temporary output tables for transformations are retained until the Process Designer window is closed or the current server session is ended in some other way (for example, by selecting **Process ▶ Kill** from the menu bar).

The temporary output tables for transformations can be used to debug the transformation, as described in "Analyzing Transformation Output Tables" on page 192. However, as long as you keep the job open in the Process Designer window, the output tables remain in the Work library on the SAS Workspace Server that executed the job. If this is not what you want, you can manually delete them, or you can close the Process Designer window and reopen it. This deletes the temporary output tables.

Features of SAS Data Integration Studio

Main Software Features

The next table describes the main features that are available in SAS Data Integration Studio.

Table 2.1 Main Features of SAS Data Integration Studio

Feature	Related Documentation
Capture source data from SAS, database management systems, and enterprise resource planning systems.	See "Registering Sources and Targets" on page 97.
Import or design metadata for targets in SAS, database management systems, and enterprise resource planning systems.	See "Registering Sources and Targets" on page 97 and "Importing and Exporting Metadata" on page 98.
Build process flows, view results, and capture run-time information.	See "Working With Jobs" on page 99 and "Analyzing Process Flow Performance" on page 187.
Provide a multi-user development environment.	See "Working with Change Management" on page 113.
Deploy completed process flows into a test environment or a production environment.	See "Deploying a Job for Scheduling" on page 102, "Generating a Stored Process for a Job" on page 103, "Metadata Administration" on page 71, and "Importing and Exporting Metadata" on page 98.
Manage large data collections such as data warehouses, receive logs and events, update metadata.	See " Importing Metadata with Change Analysis" on page 99, Chapter 11, "Optimizing Process Flows," on page 181, and "Updating Metadata" on page 105.

CHAPTER

3

About the Main Windows and Wizards

Overview of the Main Windows

The following table lists the main windows and components in SAS Data Integration Studio. Each component is briefly described in the sections that follow.

Table 3.1 SAS Data Integration Studio Interface

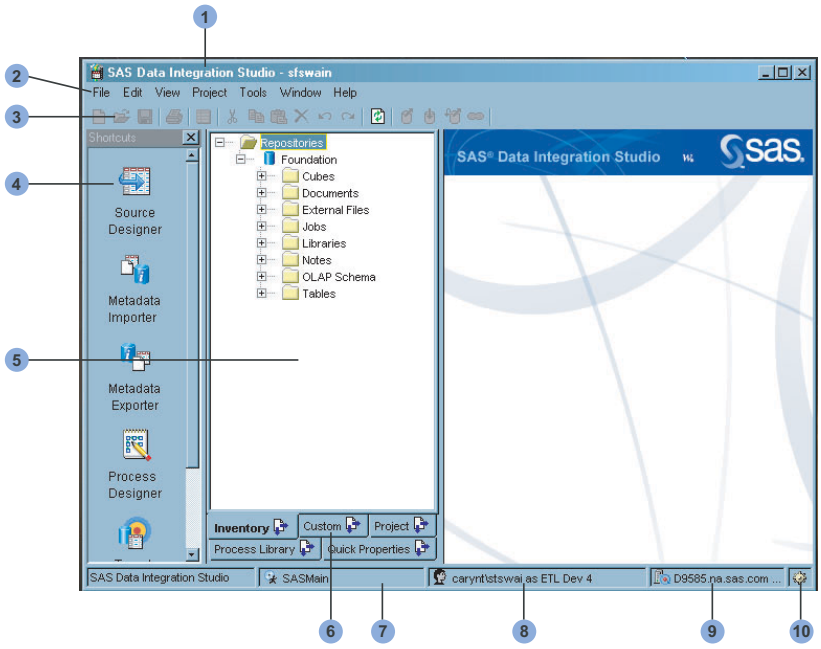
Window	Description
"About the Desktop" on page 13	Use to begin working with the metadata repositories that are specified in the current metadata profile.
"Expression Builder Window" on page 16	Use to create SAS expressions that aggregate columns, perform conditional processing, and perform other tasks.
"Job Properties Window" on page 17	Enables you to view or update the basic metadata for a job (metadata other than its process flow diagram).
"Open a Metadata Profile Window" on page 17	Use to open or maintain a metadata profile, which connects to a metadata server.
"Options Window" on page 19	Use to specify options for SAS Data Integration Studio.
"Process Designer Window" on page 20	Use to create process flow diagrams, to generate and submit code for jobs, and to perform related tasks.
"Process Library" on page 23	Use to drag and drop transformations into a process flow.
"Source Editor Window" on page 25	Use to edit SAS code.
"Table or External File Properties Window" on page 26	Enables you to view or update the metadata for a table.
"Transformation Properties Window" on page 27	Enables you to view or update the metadata for a transformation in the process flow diagram for a job.
"View Data Window" on page 27	Enables you to display data in a SAS table or view, in an external file, or in a DBMS table or view that is part of a SAS library for DBMS data stores.

About the Desktop

Overview of the Desktop

After you open a metadata profile, the SAS Data Integration Studio desktop displays.

Display 3.1 SAS Data Integration Studio Desktop



The desktop consists of the following main parts:

1 Metadata profile name
2 Menu bar
3 Toolbar
4 Shortcut bar
5 Tree view
6 Tree tabs
7 Default SAS application server
8 User ID and identity
9 Metadata server and port
10 Job status icon

Metadata Profile Name

The title bar in this window shows the name of the metadata profile that is currently in use. A metadata profile is a client-side definition of where the metadata server is located. This definition includes a machine name, a port number, and one or more metadata repositories. In addition, the metadata profile can contain a user's login

information and instructions for connecting to the metadata server automatically. Two parts—the user ID and identity, and the metadata server and port—are derived from the current metadata profile. For more information about metadata profiles, see "Creating a Metadata Profile (for Administrators)" on page 58 and "Creating a Metadata Profile (for Users)" on page 94.

Menu Bar

Use the menu bar to access the drop-down menus. The list of active options varies according to the current work area and the kind of object that you select. Inactive menu options are disabled or hidden.

Toolbar

The toolbar contains shortcuts for items on the menu bar. The list of active options varies according to the current work area and the kind of object that you select. Inactive menu options are disabled or hidden.

Shortcut Bar

The shortcut bar displays a pane of task icons on the left side of the SAS Data Integration Studio desktop. To display it, select **View ▶ Shortcut Bar** from the menu bar. Each icon displays a commonly used window, wizard, or a selection window for wizards.

Tree View

The tree view displays several different hierarchical lists or trees. Use the tabs at the bottom of this pane, such as **`Inventory`** and **`Custom`**, to display different trees. The following trees are the ones that you will use most of the time.

The Inventory tree displays the objects in the default metadata repository, as well as the objects from any repositories on which the default repository depends. The Inventory tree organizes metadata objects into a set of default groups, such as **`Tables`** for all table metadata in a repository, and **`Cubes`** for all cube metadata in a repository.

Like the Inventory tree, the Custom tree displays the objects in the default metadata repository, as well as the objects from any repositories on which the default repository depends. Unlike the Inventory tree, the Custom tree enables you to create user-defined groups (folders and subfolders) that you can use to organize metadata objects in a way that is convenient for your site. Use the Custom tree to keep similar kinds of metadata together. For example, you can create a user-defined group named **`Sales`** that contains the metadata for tables that contain sales information.

The Project tree becomes active when you open a metadata profile in which the default repository is a project repository. Use the Project tree to perform these tasks:

- use change management features to display objects that have been checked out from a change-managed repository
- display objects that have been fetched from a change-managed repository
- display objects that have been created in the Project repository and have not been checked in

Change management options are available from the **Project** drop-down menu and from pop-up menus.

See also "Process Library" on page 23.

Default SAS Application Server

This pane displays the default SAS application server, if you have selected one. The message **No Default Application Server** indicates that you need to select a default SAS Application server. If you double-click this pane, the Default SAS Application Server window opens. From that window, you can select a default server or test your connection to the default server. Alternatively, you can select the default SAS application server from the Options window, as described in "Options Window" on page 19. For more information about the impact of the default SAS application server, see "Selecting a Default SAS Application Server" on page 96.

User ID and Identity

This pane displays the domain, login user ID, and metadata identity that are specified in the current metadata profile.

Metadata Server and Port

This pane displays the name and port of the SAS Metadata Server that is specified in the current metadata profile.

Job Status Icon

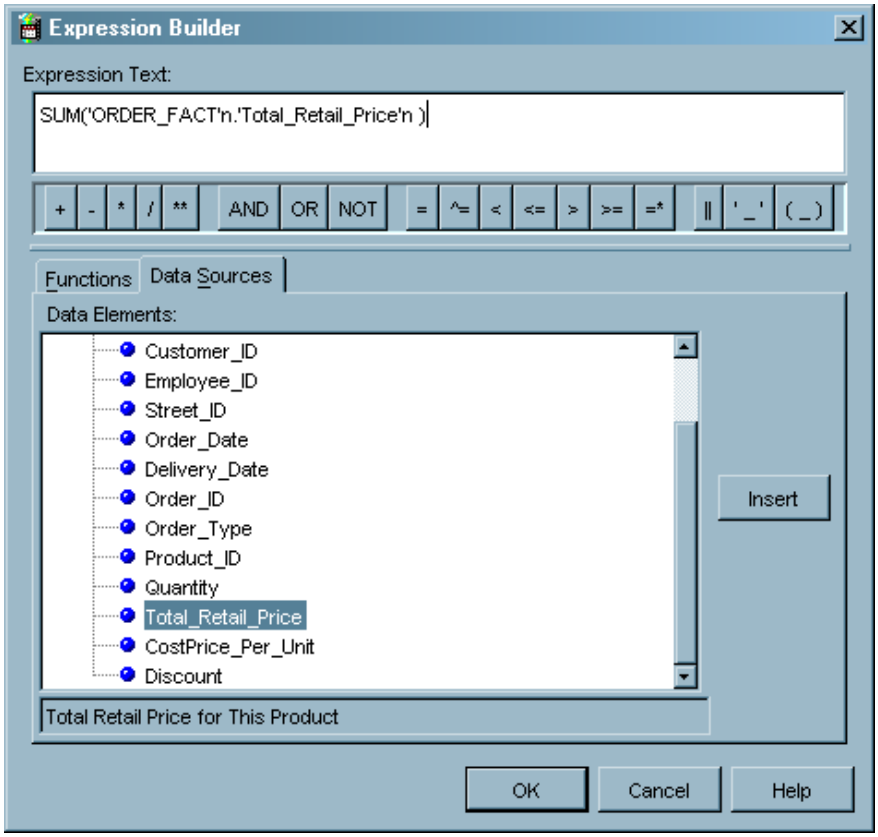
You can right-click this icon to display a list of the last five unique jobs that were submitted in the current SAS Data Integration Studio session and the status for each of them.

Note: The appearance of the icon changes to indicate changes in the status of the jobs that you submitted in the current session. The icon appearance returns to normal when you right-click the icon. △

Expression Builder Window

Use the Expression Builder to create SAS expressions that aggregate columns, perform conditional processing, and perform other tasks. For example, the following display shows an expression that sums the values in a column named Total_Retail_Price.

Display 3.2 Expression Builder Window with a SUM Expression



The Expression Builder is displayed from tabs in the property windows of many SAS Data Integration Studio transformations. For an example of how the Expression Builder can be displayed and used in the context of a job, see "Change One Column to a Calculated Column" on page 157.

The Expression Builder is used to add or update expressions in SAS, SQL, or MDX. The expression can transform columns, provide conditional processing, calculate new values, and assign new values. The expressions specify the following elements, among others:

- column names
- SAS functions
- constants (fixed values)
- sequences of operands (something to be operated on like a column name or a constant) and operators, which form a set of instructions to produce a value

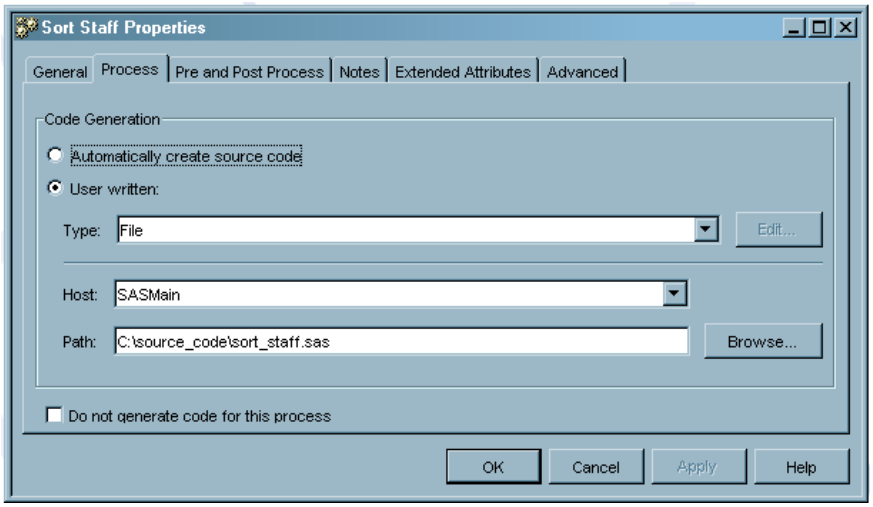
An expression can be as simple as a constant or a column name, or an expression can contain multiple operations connected by logical operators. For example, an expression to define how the values for the column COMMISSION are calculated could be **`amount`**

`* .01`. An example of conditional processing to subset data could be **`amount > 10000 and region = 'NE'`**. Other examples could be an expression to convert a character date into a SAS date or an expression to concatentate columns. For details about SAS expressions, see *SAS Language Reference: Concepts* in SAS OnlineDoc.

Job Properties Window

Use the properties window for a job to view or update its basic metadata. For example, you can specify whether the code for the current job will be generated by SAS Data Integration Studio or will be retrieved from a specified location. You can also use this window to specify code that should be run before or after a job executes. The following display shows a typical window.

Display 3.3 Job Properties Window



If you want to specify user-written code for the Sort Staff job that is described in "Process Flows and Jobs" on page 6, you can enter metadata that is similar to the metadata that is shown in the previous display. In the job properties window shown in the previous display, the **`User written`** option has been selected, and the physical path to a source code file has been specified.

If you wanted to execute code before or after the Sort Staff job is executed, you can click the **`Pre and Post Process`** tab and specify the code. For example, you might want to issue a SAS LIBNAME statement before the job is run.

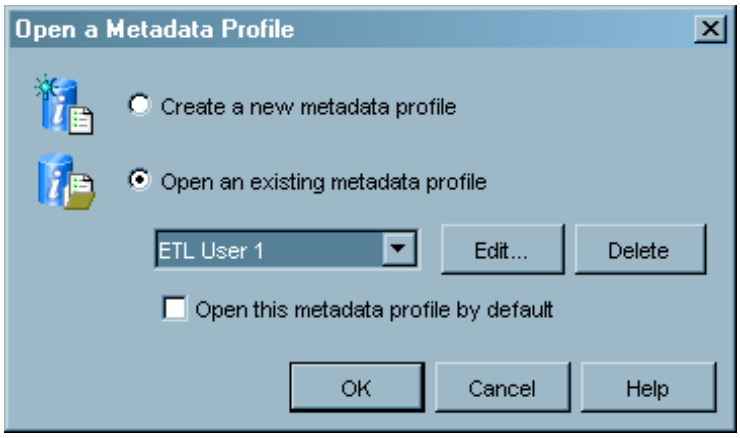
For a summary of how to use the job properties window, see "Updating Metadata for Jobs" on page 105 and "Viewing Metadata for Jobs" on page 111.

Open a Metadata Profile Window

A metadata profile is a client-side definition of where a metadata server is located. The definition includes a host name, a port number, and a list of one or more metadata repositories. In addition, the metadata profile can contain a user's login information and instructions for connecting to the metadata server either automatically or manually.

When you start SAS Data Integration Studio, the Open a Metadata Profile window displays in front of the desktop. The following display shows an example of this window.

Display 3.4 Open a Metadata Profile Window

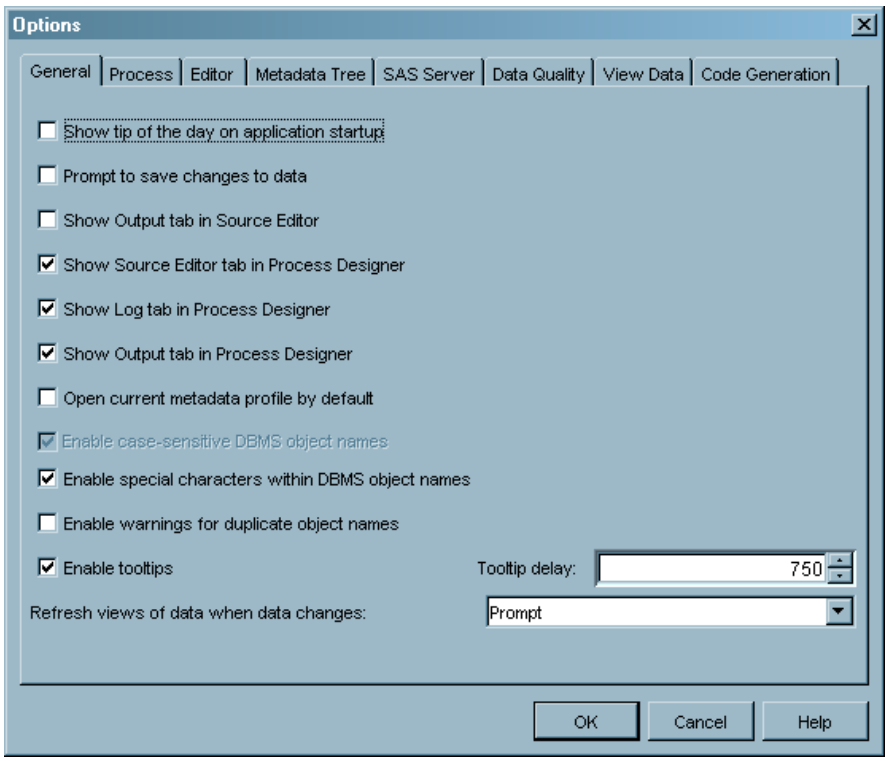


Use the Open a Metadata Profile window to open an existing metadata profile, edit an existing metadata profile, or add a new metadata profile. You must open a metadata profile before you can do any work in SAS Data Integration Studio.

Options Window

Use the Options window to specify options for SAS Data Integration Studio. The following display shows an example of this window.

Display 3.5 Options Window



The Options window contains the following tabs:

General	specifies general user interface options for SAS Data Integration Studio, such as desktop colors or whether to show the SAS Log tab in the Process Designer.
Process	specifies options for the Process Editor tab in the Process Designer. These options control how process flow diagrams display.
Editor	specifies options for the Source Editor tab in the Process Designer. These options control the appearance and behavior of the Source Editor.
Metadata Tree	specifies what types of metadata are displayed in the Metadata tree on the SAS Data Integration Studio desktop.
SAS Server	specifies the default SAS application server for SAS Data Integration Studio.
Data Quality	specifies options for Data Quality transformation templates that are available in the Process Library tree.
View Data	specifies options for the View Data window that is available for tables, external files, and selected transformations.
Code Generation	specifies options for code generation and parallel processing.

The following steps describe one way to view or update the options in the Options window:

1. From the SAS Data Integration Studio desktop, select **Tools** ► **Options** to display the Options window.
2. Select the tab that contains the options that you want to view or update.

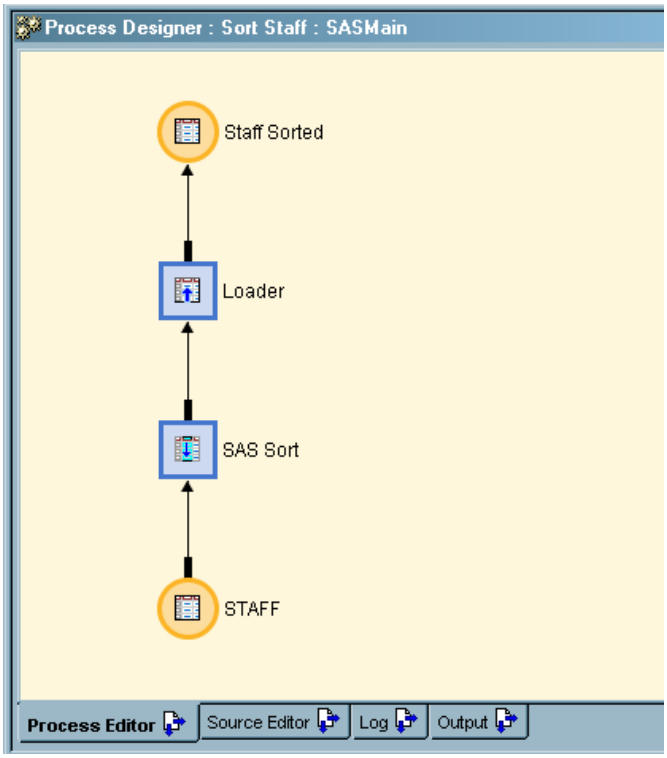
Process Designer Window

Use the Process Designer window to perform these tasks:

- Maintain the process flow diagram for the selected job.
- View or update the metadata for sources, targets, and transformations within the selected job.
- View or update the code that is generated for the entire selected job or for a transformation within that job.
- View a log that indicates whether code was successfully generated for the selected job or for one of its transformations (and was successfully executed, if the code was submitted for execution).
- View any output that the selected job or one of its transformations sends to the SAS output window.

The following display shows a typical view of this window.

Display 3.6 Process Designer Window



In the previous display, the Process Designer window contains the process flow diagram for the Sort Staff job that is described in “Process Flows and Jobs” on page 6. Note that the **`Process Editor`** tab is shown by default. You might need to use the Options

window to display the other tabs in the Process Designer window or to specify options for these tabs. For details, see "Options Window" on page 19.

The following steps describe one way to open an existing job in the Process Designer window:

1. From the SAS Data Integration Studio desktop, display the Inventory tree.
2. In the Inventory tree, expand the Jobs group.
3. Select the desired job, then select **View ▶ View Job** from the menu bar. The process flow diagram for the job displays in the **`Process Editor`** tab in the Process Designer window.

If the diagram is too large to view in the **`Process Editor`** tab, select **View ▶ Overview** from the menu bar. A small image of the complete process flow diagram displays in the Overview window.

To change the size or the orientation of the process flow diagram, select **Process ▶ Zoom** or **Process ▶ Layout** from the menu bar.

The tabs in the Process Designer window are described in the following sections. To display the online Help for each tab, select the tab and press the **`F1`** key.

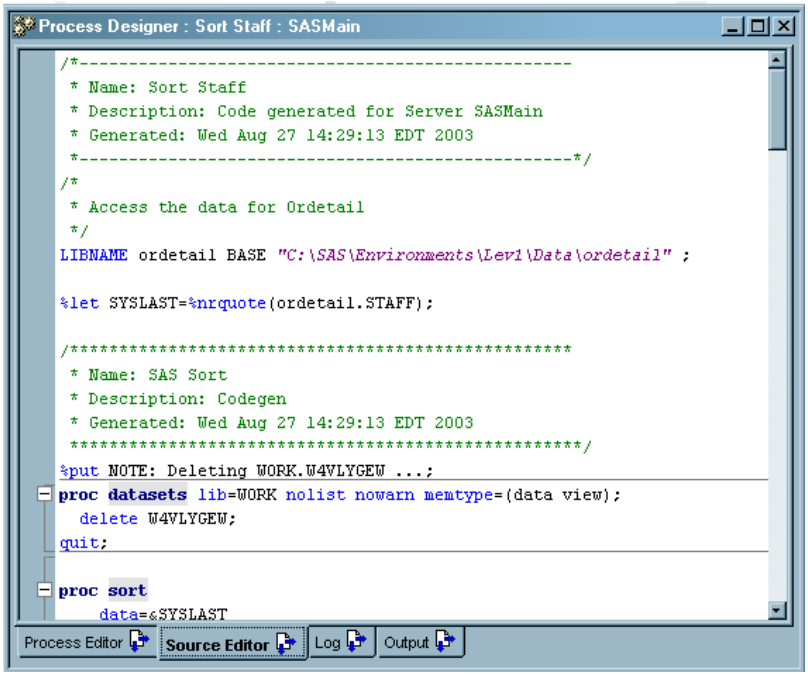
Process Editor Tab

Use the **`Process Editor`** tab to add and maintain a process flow diagram for the selected job. For a summary of how you can use the Process Editor to create a process flow diagram for a job, see "Main Tasks for Creating Process Flows" on page 96.

Source Editor Tab

Use the **`Source Editor`** tab to view or modify SAS code for the selected job. For example, if the Sort Staff job was displayed on the **`Process Editor`** tab, and you selected the **`Source Editor`** tab, code for the entire job would be generated and displayed. The following display shows some of the code that would be generated for the Sort Staff job.

Display 3.7 Source Editor Tab



Log Tab

Use the **Log** tab to view the SAS log that is returned from the code that was submitted for execution. The **Log** tab can help you identify the cause of problems with the selected job or transformation. For a summary of how you can use the **Log** tab, see “Check the Log” on page 101. See also “Using SAS Logs to Analyze Process Flows” on page 189.

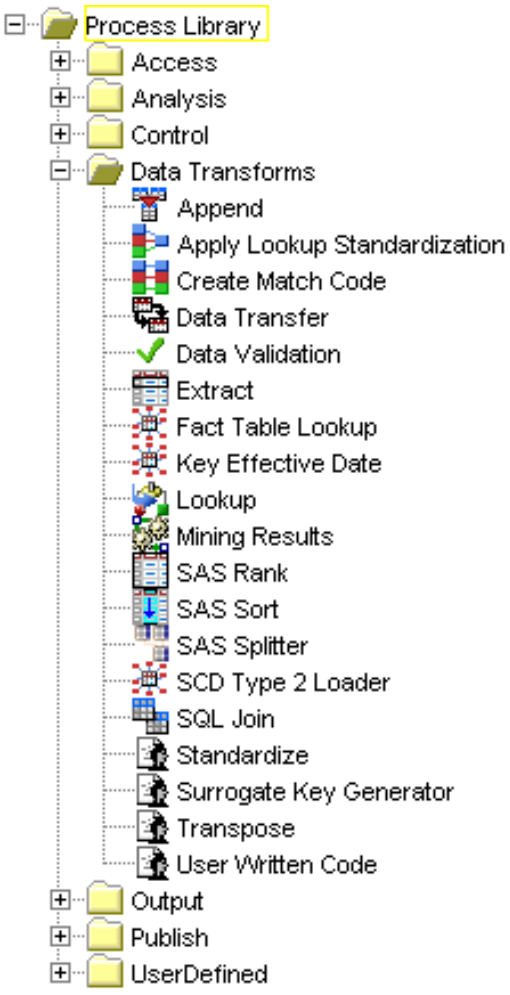
Output Tab

Use the **Output** tab to view any printed output from a SAS program. For example, SAS Data Integration Studio jobs that produce reports can specify that the reports are sent to the **Output** tab. For an example of a job that sends output to the **Output** tab, see “Example: Using a Generated Transformation in a Job” on page 174.

Process Library

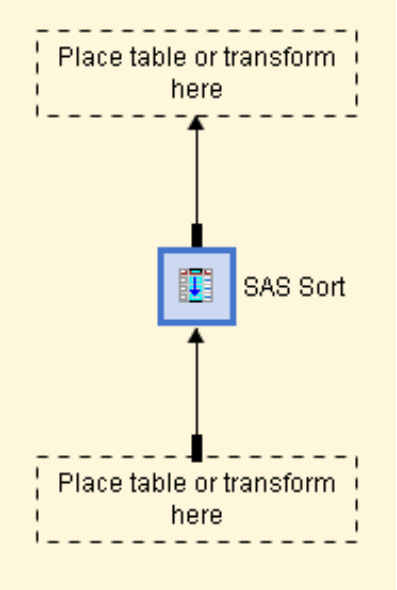
The **`Process Library`** tree is one of the tabs in the tree view on the SAS Data Integration Studio desktop. If you select this tab, it displays a collection of transformation templates. As shown in the following display, the templates are organized into folders, such as **`Access`**, **`Analysis`**, **`Data Transforms`**, and **`Publish`**.

Display 3.8 Process Library Tree



A transformation template is a process flow diagram that includes drop zones for metadata that the user must supply to make the transformation complete. A template typically consists of a transformation object and one or more drop zones for sources, targets, or both, as shown in the following display.

Display 3.9 SAS Sort Transformation Template



There are several kinds of drop zones:

- Dashed line boxes. Before a template is populated with the minimum sources and targets, drop zones are indicated by dashed line boxes, as shown in the previous display.
- Lines between objects in a process flow diagram. After a template is populated with the minimum sources and targets, drop zones are indicated by lines between objects in the process flow diagram, as shown in Display 3.6 on page 20.
- Transformation objects themselves. Transformations that can take multiple inputs or outputs have drop zones on the transformation itself.

The SAS Sort template that is shown in the previous display could be used to create the diagram that is shown in Display 3.6 on page 20. For a summary of how you can use the Process Editor and the Process Library tree to create a process flow diagram for a job, see “Creating, Running, and Verifying Jobs” on page 99.

Java Transformations and Generated Transformations

The Process Library tree contains two different kinds of transformations: Java plug-in transformations and generated transformations (formerly called SAS code transformations).

Java plug-in transformations are created with the Java programming language. Examples include most of the default transformations in the **`Analysis`** folder, such as SAS Sort and SAS Splitter.

Generated transformations are created with the Transformation Generator wizard. Examples include the default transformations in the **`Output`** folder and the **`Publish`** folder. With the exception of the User-Written Code transformation, which is a Java

plug-in, all other templates in the Process Library that have the user-written icon are generated transformations.

Generated transformations are unique in two ways:

- When you right-click a generated transformation in the Process Library tree, the pop-up menu has two unique options: **`Edit Source`** and **`Transformation Export`**.
- You can easily add your own generated transformations to the Process Library tree, where you can drag and drop them into the process flow diagram for any job.

For details about the Transformation Generator wizard, see "Transformation Generator Wizard" on page 34.

Additional Information About the Process Library Transformations

For a description of the standard transformations in the Process Library, see Appendix 1, "Standard Transformations in the Process Library," on page 217.

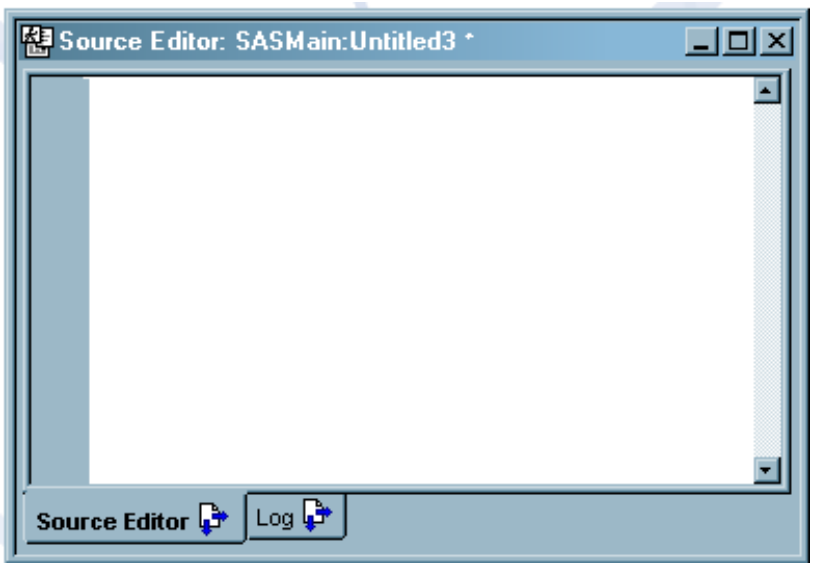
To display examples that illustrate how the Process Library templates can be used in a SAS Data Integration Studio job, follow these steps:

1. From the SAS Data Integration Studio menu bar, select **Help ▶ Contents**. The online Help window displays.
2. In the left pane of the Help window, select **Examples ▶ Process Library Examples**.

Source Editor Window

SAS Data Integration Studio also provides a Source Editor window that you can use as a general-purpose SAS code editor. Display 3.10 on page 25 shows an example of this window.

Display 3.10 Source Editor Window



To display the Source Editor window, from the SAS Data Integration Studio desktop, select **Tools ▶ Source Editor**.

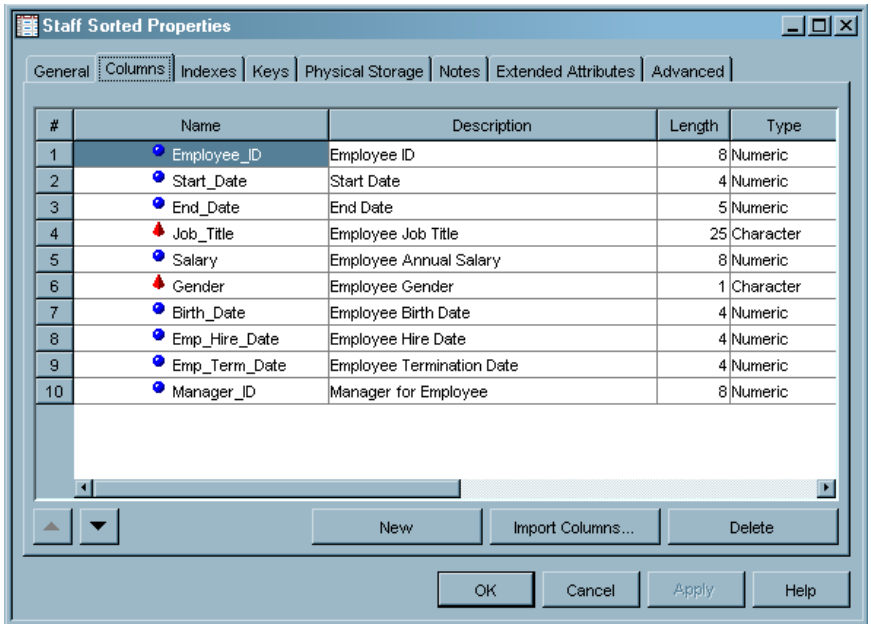
To submit code from the Source Editor, from the SAS Data Integration Studio desktop, select **Editor ▶ Submit**.

To display Help for this window, press the F1 key.

Table or External File Properties Window

Use the properties window for a table or an external file to view or update the metadata for its columns and other attributes. The following display shows a typical window.

Display 3.11 Table Properties Window

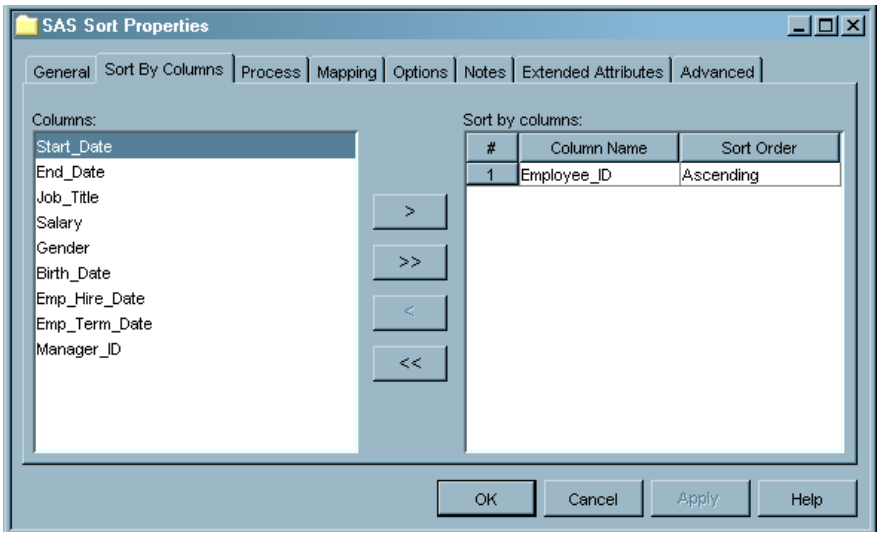


The window shown in the previous display contains the metadata for the Staff Sorted table from “Process Flows and Jobs” on page 6. For a summary of how to use this window, see “Viewing Metadata for Tables and External Files” on page 112.

Transformation Properties Window

Use a transformation properties window to view or update the metadata for a process in a job. The metadata for a transformation specifies how SAS Data Integration Studio will generate code for the corresponding process. The window for each kind of transformation has one or more tabs that are unique to the corresponding process. The following display shows a typical window.

Display 3.12 Transformation Properties Window



The window shown in the previous display contains the metadata for the SAS Sort transformation from "Process Flows and Jobs" on page 6. Note that the rows in the output table for this transformation will be sorted by employee ID. For a summary of how to use transformation property windows, see "Updating Metadata for Transformations" on page 108 and "Viewing Metadata for Transformations" on page 113.

View Data Window

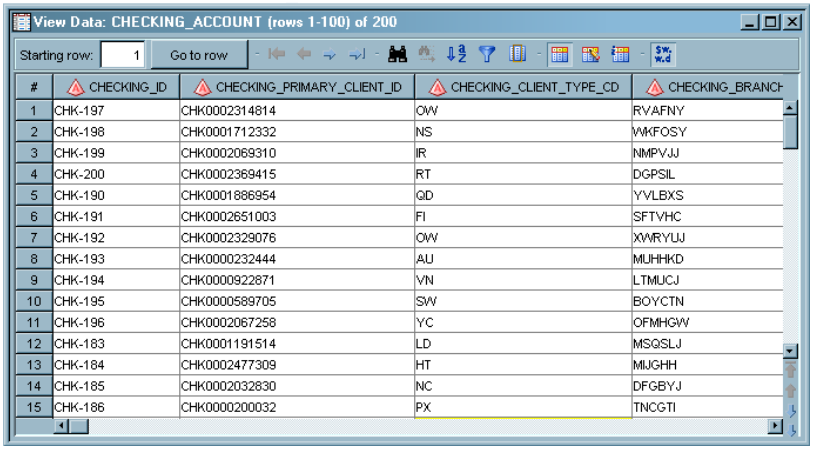
In the tree views on the desktop, use the View Data window to display data in a SAS table or view, in an external file, or in a DBMS table or view that is part of a SAS library for DBMS data stores. The table, view, or external file must be registered in a current metadata repository and must exist in physical storage.

In the Process Designer window for a job, use the View Data window to display data in the temporary output tables for a transformation. In order for the temporary output tables to exist in physical storage, the transformation must be one of the transformations that create temporary output tables, and the transformation must have been executed at least once

The View Data window typically uses the metadata for a data store to format the data for display. Accordingly, the View Data window can be used to verify that the metadata for a data store is appropriate for use in the intended job. If the window does not correctly display the data in the selected data store, then you might have to update the corresponding metadata before you use it in a job.

The following display shows a typical View Data window.

Display 3.13 View Data Window



#	CHECKING_ID	CHECKING_PRIMARY_CLIENT_ID	CHECKING_CLIENT_TYPE_CD	CHECKING_BRANCH
1	CHK-197	CHK0002314814	OW	RVAFNY
2	CHK-198	CHK0001712332	NS	WKFOSY
3	CHK-199	CHK0002069310	IR	NMPVJJ
4	CHK-200	CHK0002369415	RT	DGPSIL
5	CHK-190	CHK0001886954	QD	YVLBXS
6	CHK-191	CHK0002651003	FI	SFTVHC
7	CHK-192	CHK0002329076	OW	XWRYUJ
8	CHK-193	CHK0000232444	AU	MUHHKD
9	CHK-194	CHK0000922871	VN	LTMUCJ
10	CHK-195	CHK0000589705	SW	BOYCTN
11	CHK-196	CHK0002067258	YC	OFMHGW
12	CHK-183	CHK0001191514	LD	MSQSLJ
13	CHK-184	CHK0002477309	HT	MIJGHH
14	CHK-185	CHK0002032830	NC	DFGBYJ
15	CHK-186	CHK0000200032	PX	TNCGTI

The title bar in the View Data window displays the name of the object that is being viewed, the range of rows that are being viewed, and the total number of rows. The View Data window can display up to 100 rows of data at a time. To go to the next 100 rows of data, click the double down-arrow in the lower-right corner of the window.

If a column has a description, the description displays in the column header in the View Data window. Otherwise, the physical name of the column displays in the column header. A round icon  to the left of the name indicates that the column is numeric, and a pyramid-shaped icon  to the left of the name indicates that the column contains character data.

To customize the data view displayed in the View Data window, right-click on a column name, row number, or table cell. Then, select an appropriate option from the pop-up menu. To display Help for the View Data window, press F1. See also “Viewing Data in Tables, External Files, or Temporary Output Tables” on page 110.

Overview of the Main Wizards

Several wizards are available from the shortcut bar or from the tools menu on the SAS Data Integration Studio desktop. The next table describes the main wizards.

Table 3.2 SAS Data Integration Studio Wizards

Wizard	Description
Calculated Members	Enables you to add, edit, and delete the calculated members associated with the cubes that are registered to a current metadata repository. For more information, see the Help for this wizard.
Data Surveyors (optional)	If installed, data surveyors access the data and metadata in enterprise applications, such as the applications from vendors such as PeopleSoft, SAP R/3, Siebel, and Oracle. For more information, see the Help for these wizards.
Export Job to File	Enables you to export a selected job from the Inventory tree, Custom tree, or Project tree to an XML file. One XML file is created for each job, and you can export multiple jobs at the same time. For more information, see the Help for this wizard.
Import Cobol Copybook	Use the Import COBOL Copybook feature to create a COBOL format file from the COBOL copybook file. Then use the Fixed-Width External File wizard to copy column metadata from the COBOL format file. For more information, see the Help for this wizard.
Import Cube	After you have exported the metadata for a SAS OLAP cube, you can import this metadata and register it to a current repository. For more information, see the Help for this wizard.
Job Import and Merge	Imports SAS Data Integration Studio jobs that were previously exported in XML format. A job specifies a set of source tables, target tables, and processes. You can export a SAS Data Integration Studio job to an XML file, then use SAS Data Integration Studio to import the job into the same metadata repository or into a different repository. As long as the two repositories specify the same libraries, tables, and database schemas, the imported job can be successfully executed in the new repository. For more information, see the Help for this wizard.

Wizard	Description
Metadata Export	Exports the default metadata repository that is specified in your metadata profile. You can export metadata in Common Warehouse Metamodel (CWM) format or in formats that are supported by the optional Meta Integration Model Bridges (MIMBs) from Meta Integration Technology, Inc. See "Importing and Exporting Metadata" on page 98.
Metadata Import	Imports table metadata in CWM format or in formats that are supported by the optional MIMBs from Meta Integration Technology, Inc. You have the option of comparing the imported metadata to existing metadata. You can view any changes in the Differences window and choose which changes to apply. See "Importing and Exporting Metadata" on page 98.
Process Designer	Displays the New Job wizard. The New Job wizard enables you to specify one or more tables as the targets (outputs) of a job. This wizard can also be used to create an empty job, into which you can drag and drop tables and transformation templates. See "New Job Wizard" on page 32.
Source Designers	Generate metadata for one or more selected tables, based on the physical structure of the table(s). See "Example: Using a Source Designer to Register SAS Tables" on page 120.
Target Designers	Two target designer wizards are provided with SAS Data Integration Studio. The Cube Designer adds and maintains a cube, a data store that supports Online Analytical Processing (OLAP). See "Working with OLAP Cubes" on page 116. The Target Table Designer generates metadata for a single table that does not yet exist in physical storage, such as a table that will be created when a SAS Data Integration Studio job is executed for the first time. Other target designer wizards can be licensed for your site. See "Example: Using the Target Table Designer to Register SAS Tables" on page 140.
Transformation Exporter	Exports a generated transformation that is registered in your metadata repository. For more information, see "Exporting a Generated Transformation" on page 87.

Wizard	Description
Transformation Generator	Enables you to create a generated transformation and make it available in the Process Library tree. See "Transformation Generator Wizard" on page 34.
Transformation Importer	Imports one or more generated transformations that have been exported as XML files. These transformations will be registered in your default metadata repository. For more information, see "Importing a Generated Transformation" on page 87.

The Stored Process wizard is available when an administrator right-clicks a job in a tree view and selects **Stored Process ▶ New** from the pop-up menu. For details about stored processes in SAS Data Integration Studio, see "Converting Jobs into Stored Processes" on page 69.

The following wizards are available from the New Object wizard selection window. This window is displayed by selecting **File ▶ New Object** from the SAS Data Integration Studio desktop. Some of these wizards are also available from the properties windows for some objects, as described in the following table.

Table 3.3 Wizards That Are Accessible from New Object Wizard Selection Window

Wizard	Description
Cube Designer	Enables you to add and maintain a cube, a data store that supports OLAP. See "Working with OLAP Cubes" on page 116.
New Document	Creates a document that you can associate with one or more objects in a metadata repository. For more information, see the Help for this wizard.
External Files	Enables you to register external files in a metadata repository. Separate wizards are provided for delimited files, fixed-width files, and for user-written access to external files. For more information, see the Help for these wizards. See also "Example: Using a Source Designer to Register an External File" on page 126.
New Group	Adds a user-defined group to the Custom tree on the SAS Data Integration Studio desktop. For more information, see the Help for this wizard.

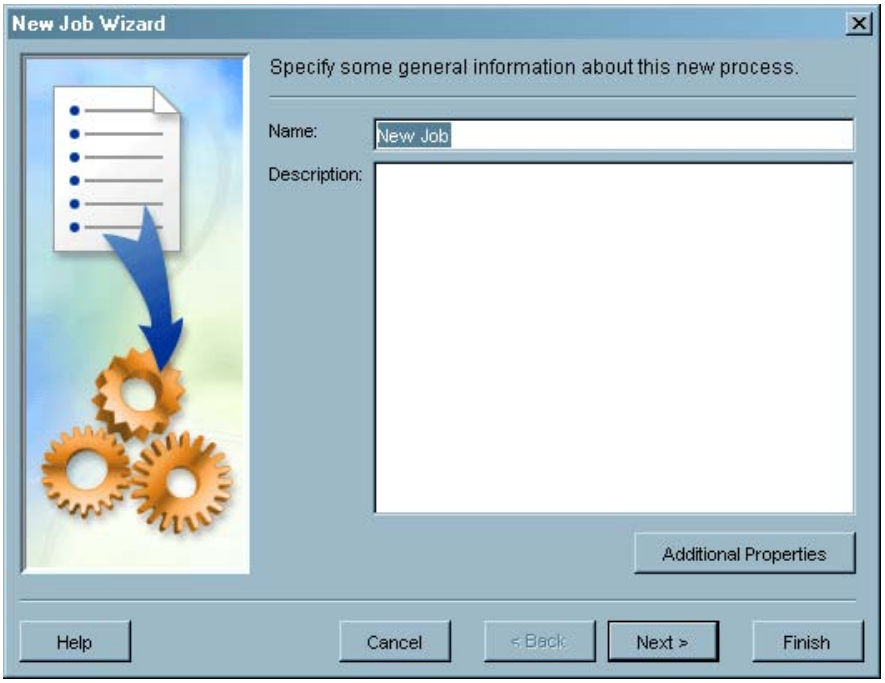
Wizard	Description
New Library	Registers a SAS library in a metadata repository. See “Registering Libraries” on page 59.
New Note	Creates a note that you can associate with one or more objects in a metadata repository. For more information, see the Help for this wizard.

New Job Wizard

Use the New Job wizard to select one or more tables as the targets (outputs) of a job. This wizard can also be used to create an empty job into which you can drag and drop tables and transformation templates.

One way to display the New Job wizard is to select **Tools ▶ Process Designer** from the menu bar on the SAS Data Integration Studio desktop. The first window in the wizard is shown in the following display.

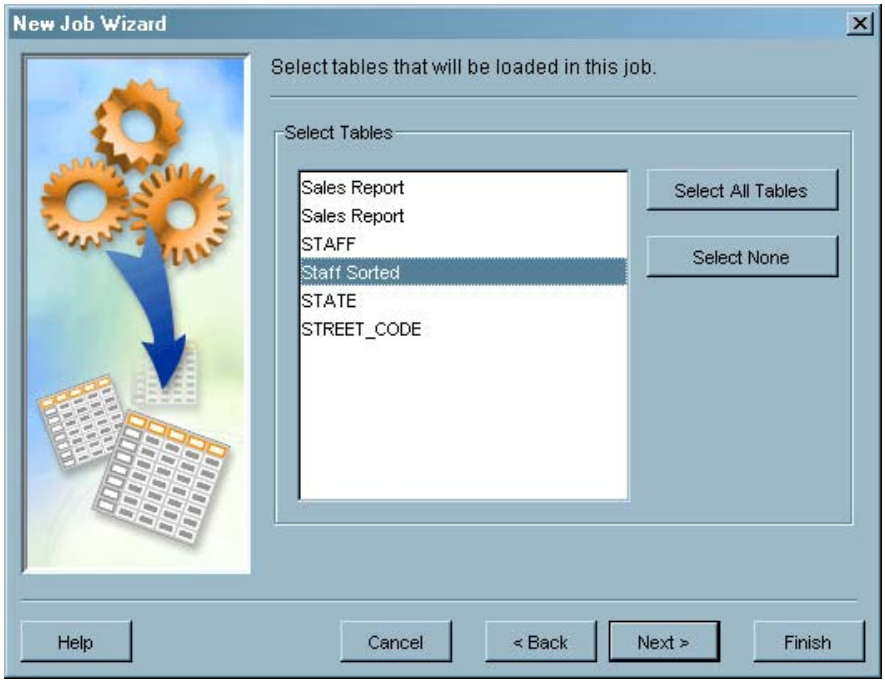
Display 3.14 New Job Wizard



The first window enables you to enter a name and description for the new job.

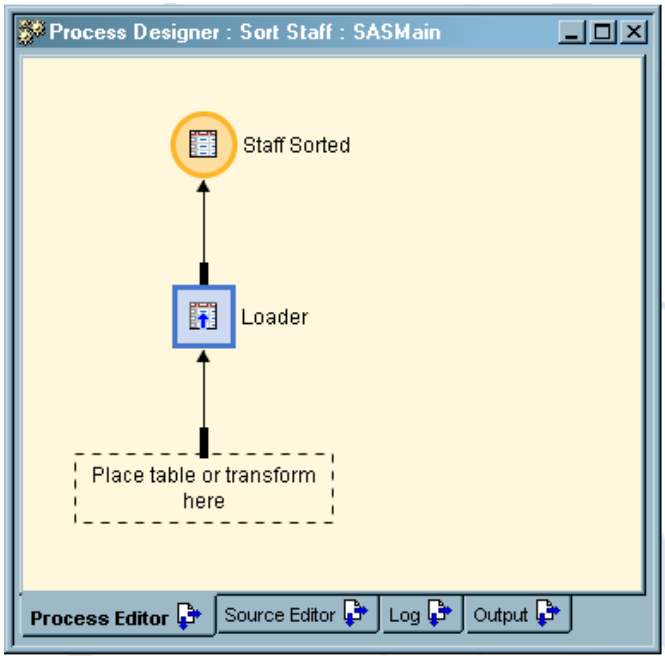
The second window of the wizard enables you to select one or more tables as the targets (outputs) of a job, as shown in the following display.

Display 3.15 New Job Wizard, Second Window



The wizard uses the selected table(s) to generate a transformation template—a process flow diagram that includes drop zones for metadata that the user must supply. For example, if the Staff Sorted table is selected as the target, the wizard would generate the transformation template that is shown in the following display.

Display 3.16 Transformation Template for Sort Staff Job



To update a process flow diagram, drag and drop tables from the Inventory tree or from another tree in the tree view. Drag and drop transformation templates from the Process Library tree.

Alternatively, in the second window of the New Job wizard, you can select no targets and click **`Finish`** after entering a name for the job. The wizard will open an empty job in the Process Designer window. After you have an empty job, you can create a process flow diagram by dragging and dropping tables and transformations into the Process Designer window. This is the approach that is described in "Create and Populate the New Job" on page 100.

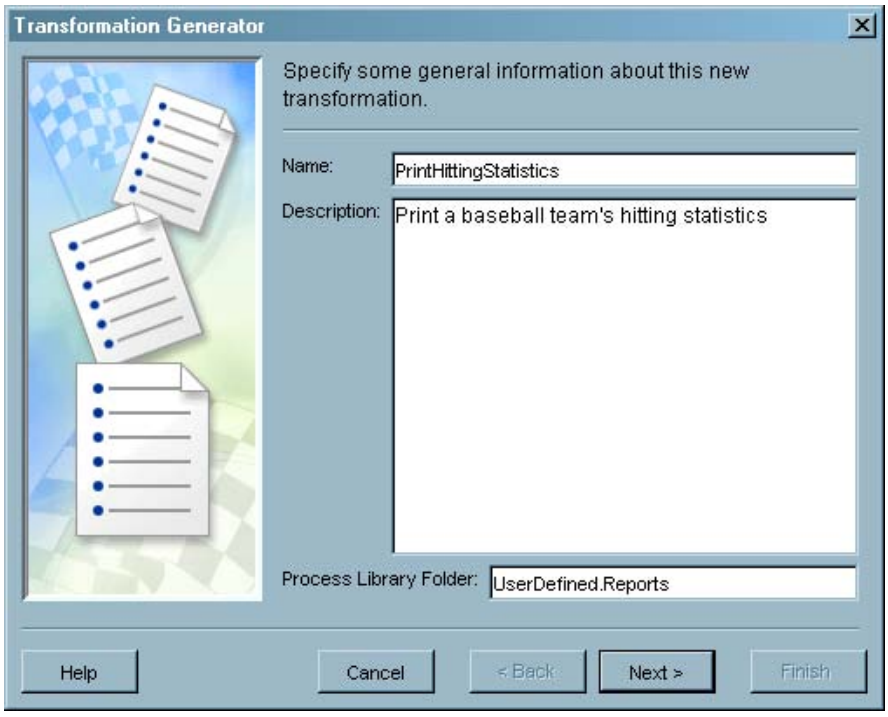
Transformation Generator Wizard

One of the easiest ways to customize SAS Data Integration Studio is to create your own generated transformations. Unlike Java-based plug-ins that require software development, generated transformations are created with a wizard.

The Transformation Generator wizard guides you through the steps of specifying SAS code for a generated transformation and saving the transformation to the current metadata repository. After the transformation is saved, it is displayed in the Process Library tree, where it is available for use in any job.

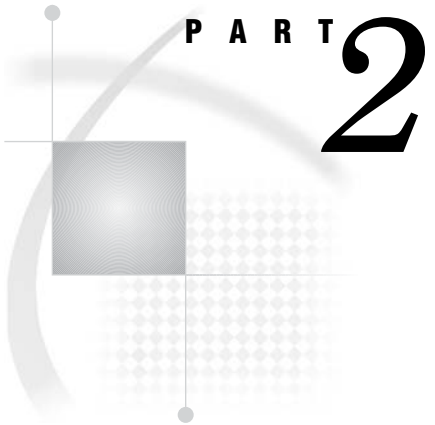
To display the Transformation Generator wizard, go to the SAS Data Integration Studio desktop, then select **Tools ▶ Transformation Generator** from the menu bar. The general information window of the wizard is displayed, as shown in the following display.

Display 3.17 General Information Window, Transformation Generator Wizard



The general information window enables you to enter a name and description for the new transformation. It also enables you to specify the folder where the new transformation will appear in the Process Library tree.

For details about using the Transformation Generator wizard, see “Maintaining Generated Transformations” on page 75.



PART 2

Planning, Installation, and Setup

CHAPTER

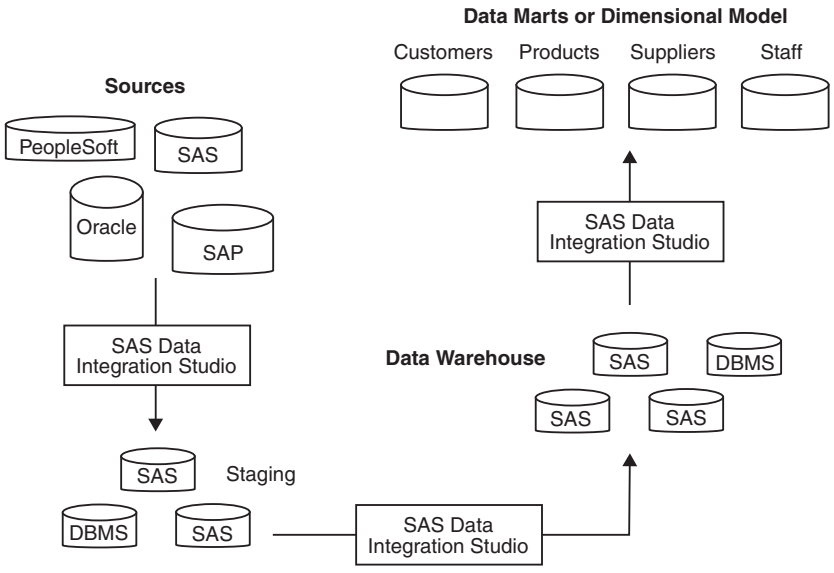
4

Designing a Data Warehouse

Overview of Warehouse Design

The following figure shows how SAS Data Integration Studio is used to flow data into and out of a central data warehouse.

Figure 4.1 Best Practice Data Warehouse



In this model, SAS Data Integration Studio jobs are used to perform the following tasks:

1 Extract enterprise data into a staging area.

2 Cleanse and validate data and load a central data warehouse.
3 Populate a data mart or dimensional model that provides collections of data from across the enterprise.

Each step of the enterprise data model is implemented by multiple jobs in SAS Data Integration Studio. Each job in each step can be scheduled to run at the time or event that best fits your business needs and network performance requirements.

Data Warehousing with SAS Data Integration Studio

Developing an Enterprise Model

SAS Data Integration Studio helps you build dimensional data from across your enterprise in three steps:

- Extract source data into a staging area (see "Step 1: Extract and Denormalize Source Data" on page 40).
- Cleanse extracted data and populate a central data warehouse (see "Step 2: Cleanse, Validate, and Load Data" on page 40).
- Create dimensional data that reflects important business needs (see "Step 3: Create Data Marts or Dimensional Data" on page 41).

The three-step enterprise model represents best practices for large enterprises. Smaller models can be developed from the enterprise model. For example, you can easily create one job in SAS Data Integration Studio that extracts, transforms, and loads data for a specific purpose.

Step 1: Extract and Denormalize Source Data

The extraction step consists of a series of SAS Data Integration Studio jobs that capture data from across your enterprise for storage in a staging area. SAS data access capabilities in the jobs enable you to extract data without changing your existing systems.

The extraction jobs denormalize enterprise data for central storage. Normalized data (many tables, few connections) is efficient for data collection. Denormalized data (few tables, more connections) is more efficient for a central data warehouse, where efficiency is needed for the population of data marts.

Step 2: Cleanse, Validate, and Load Data

After loading the staging area, a second set of SAS Data Integration Studio jobs cleanse the data in the staging area, validate the data prior to loading, and load the data into the data warehouse.

Data quality jobs remove redundancies, deal with missing data, and standardize inconsistent data. They transform data as needed so that the data fits the data model. For more information about available data cleansing capabilities, see the *SAS Data Quality Server: Reference*.

Data validation ensures that the data meets established standards of integrity. Tests show that the data is fully denormalized and cleansed, and that primary keys, user keys, and foreign keys are correctly assigned.

When the data in the staging area is valid, SAS Data Integration Studio jobs load that data into the central data warehouse.

Step 3: Create Data Marts or Dimensional Data

After the data has been loaded into the data warehouse, SAS Data Integration Studio jobs extract data from the warehouse into smaller data marts, OLAP structures, or star schemas that are dedicated to specific business dimensions, such as products, customers, suppliers, financials, and employees. From these smaller structures, additional SAS Data Integration Studio jobs generate, format, and publish reports throughout the enterprise.

Planning a Data Warehouse

The following steps outline one way of implementing a data warehouse.

1 Determine your initial needs:
   - a Generate a list of business questions that you would like to answer.
   - b Specify data collections (data marts or dimensional data) that will provide answers to your business questions.
   - c Determine how and when you would like to receive information. Information can be delivered based on events, such as supply shortages, on time, such as monthly reports, or simply on demand.

2 Map the data in your enterprise:
   - □ Locate existing storage locations for data that can be used to populate your data collections.
   - □ Determine storage format, data columns, and operating environments.

3 Create a data model for your central data warehouse:
   - □ Combine selected enterprise data sources into a denormalized database that is optimized for efficient data extraction and ad hoc queries. SAS Data Integration Studio resolves issues surrounding the extraction and combination of source data.
   - □ Consider a generalized collection of data that might extend beyond your initial scope to account for unanticipated business requirements.

4 Estimate and order hardware and software:
   - □ Include storage, servers, backup systems, and disaster recovery.
   - □ Include the staging area, the central data warehouse, and the data marts or dimensional data model.

5 Based on the data model, develop a plan for extracting data from enterprise sources into a staging area. Then specify a series of SAS Data Integration Studio jobs that put the extraction plan into action:
   - □ Consider the frequency of data collection based on business needs.
   - □ Consider the times of data extraction based on system performance requirements and data entry times.
   - □ Note that all data needs to be cleansed and validated in the staging area to avoid corruption of the data warehouse.
   - □ Consider validation steps in the extraction jobs to ensure accuracy.

6 Plan and specify SAS Data Integration Studio jobs for data cleansing in the staging area:
   - SAS Data Integration Studio contains all of the data cleansing capabilities of the SAS Data Quality Server software.
   - Column combination and creation are readily available through the data quality functions that are available in the SAS Data Integration Studio Expression Builder.

7 Plan and specify SAS Data Integration Studio jobs for data validation and load:
   - Ensure that the extracted data meets the data mode of the data warehouse before the data is loaded into the data warehouse.
   - Load data into the data warehouse at a time that is compatible with the extraction jobs that populate the data marts.

8 Plan and specify SAS Data Integration Studio jobs that populate data marts or a dimensional model out of the central data warehouse.

9 Plan and specify SAS Data Integration Studio jobs that generate reports out of the data marts or dimensional model. These jobs and all SAS Data Integration Studio jobs can be scheduled to run at specified times.

10 Install and test the hardware and software that was ordered previously.

11 Develop and test the backup and disaster recovery procedures.

12 Develop and individually test the SAS Data Integration Studio jobs that were previously specified.

13 Perform an initial load and examine the contents of the data warehouse to test the extract, cleanse, verify, and load jobs.

14 Perform an initial extraction from the data warehouse to the data marts or dimensional model. Then examine the smaller data stores to test that set of jobs.

15 Generate and publish an initial set of reports to test that set of SAS Data Integration Studio jobs.

Planning Security for a Data Warehouse

You should develop a security plan for controlling access to libraries, tables, and other resources that are associated with a data warehouse. The phases in the security planning process are as follows:

- Define your security goals.
- Make some preliminary decisions about your security architecture.
- Determine which user accounts you must create with your authentication providers and which user identities and logins you must establish in the metadata.
- Determine how you will organize your users into groups.
- Determine which users need which permissions to which resources, and develop a strategy for establishing those access controls.

For details about developing a security plan, see the security planning chapter in the *SAS Intelligence Platform: Security Administration Guide*.

CHAPTER 5

Example Data Warehouse

Overview of Orion Star Sports & Outdoors

Orion Star Sports & Outdoors is a fictitious international retail company that sells sports and outdoor products. The headquarters is based in the United States, and retail stores are situated in several other countries including Belgium, Holland, Germany, the United Kingdom, Denmark, France, Italy, Spain, and Australia. Products are sold through physical retail stores, as well as through mail-order catalogs and on the

Internet. Customers who sign up as members of the Orion Star Club organization can receive favorable special offers; therefore, most customers enroll in the Orion Star Club.

Note: The sample data for Orion Star Sports & Outdoors is for illustration only. The reader is not expected to use sample data to create the data warehouse that is described in the manual. △

Asking the Right Questions

Possible High-Level Questions

Suppose that the executives at Orion Star Sports & Outdoors want to be proactive in regard to their products, customers, delivery, staff, suppliers, and overall profitability. They might begin by developing a list of questions that needed to be answered, such as the following:

Product Sales Trends

- What products are available in the company inventory?
- What products are selling?
- What are the time and place dependencies of product sales?
- Who is making the sales?

Slow-Moving Products

- Which products are not selling?
- Are these slow sales time or place dependent?
- Which products do not contribute at least 0.05% to the revenue for a given country/year?
- Can any of these products be discontinued?

Profitability

- What is the profitability of products, product groups, product categories, and product line?
- How is the profitability related to the amount of product sold?

Discounting

- Do discounts increase sales?
- Does discounting yield greater profitability?

After reviewing their list of questions, Orion Star executives might select a few questions for a pilot project. For example, the executives might choose the following two initial questions:

- Which salesperson is making the most sales?
- What are the time and place dependencies of product sales?

The executives would then direct the data warehousing team to answer the selected questions. The examples used in this manual are derived from the selected questions.

Which Salesperson Is Making the Most Sales?

Identifying Relevant Information

To answer the question, "Which salesperson is making the most sales?", the data warehousing team decided to design a report that lists total sales by employee. The following display shows an example of such a report.

Display 5.1 Total Sales by Employee Report

Name	Total Revenue	Emp ID	Job Title	Company	Dept	Section	Group
Alban Kingston	$5,030.00	120170	Sales Rep II	Orion France	Sales	Sales	Racket Sports
Andre Noel	$4,535.00	120411	Sales Rep i	Orion France	Sales	Sales	Winter Sports
Giulia Buono	$6,230.00	120599	Sales Rep II	Orion Italy	Sales	Sales	Children Sports

The next step is to identify how such a report could be created.

Identifying Sources

The data warehouse team examined existing tables to determine if they could be used to the create the report shown in the previous display. They identified several tables that could be used. These tables are described in the following sections.

Source for Staff Information

The STAFF table contains information about employees, such as name, ID, department, supervisor, and salary, as shown in the following display.

Display 5.2 The STAFF Table

View Data: STAFF

#	Employee ID	Start Date	End Date	Employee Job Title	Employee Annual Salar
1	120101	01JUL1999	31DEC9999	Director	$163,040
2	120102	01JUN1985	31DEC9999	Sales Manager	$108,255
3	120103	01JAN1970	31DEC9999	Sales Manager	$87,975
4	120104	01JAN1977	31DEC9999	Administration Manager	$46,230
5	120105	01MAY1995	31DEC9999	Secretary I	$27,110
6	120106	01JAN1970	31DEC9999	Office Assistant II	$26,960
7	120107	01FEB1970	31DEC9999	Office Assistant III	$30,475
8	120108	01AUG2002	31DEC9999	Warehouse Assistant II	$27,660
9	120109	01OCT2002	31DEC9999	Warehouse Assistant I	$26,495
10	120110	01NOV1975	31DEC9999	Warehouse Assistant III	$28,615
11	120111	01NOV1970	31DEC9999	Security Guard II	$26,895
12	120112	01JUL1986	31DEC9999	Security Guard I	$26,550
13	120113	01JAN1970	31DEC9999	Security Guard II	$26,870
14	120114	01JAN1970	31DEC9999	Security Manager	$31,285

Source for Organization Information

The following ORGANIZATION table identifies the organization to which an employee belongs.

Display 5.3 The ORGANIZATION Table

View Data: ORGANIZATION

#	Organization Name	Country Abbreviation	Organization Level Number	Start Date
1	Lu	Australia	1	01JUL1999
2	hou	Australia	1	01JUN1985
3	Dawes	Australia	1	01JAN1970
4	n Billington	Australia	1	01JAN1977
5	vey	Australia	1	01MAY199
6	lornsey	Australia	1	01JAN1970
7	Sheedy	Australia	1	01FEB1970
8	Gromek	Australia	1	01AUG2002
9	le Baker	Australia	1	01OCT2002
10	Entwisle	Australia	1	01NOV1975
11	Spillane	Australia	1	01NOV1970
12	attback	Australia	1	01JUL1986
13	rsey	Australia	1	01JAN1970
14	tte Buddery	Australia	1	01JAN1970

Source for Order Information

The following ORDERS table contains information about orders placed with salespersons, including date, salesperson ID, type of order, and customer.

Display 5.4 The ORDERS Table

View Data: ORDERS

#	Order ID	Order Type	Employee ID	Customer ID	Date Order was placed by Custo
1	1230000033	Internet Sale	99999999	8818	01JAN1998
2	1230000204	Internet Sale	99999999	47793	01JAN1998
3	1230000268	Internet Sale	99999999	71727	01JAN1998
4	1230000487	Internet Sale	99999999	74503	01JAN1998
5	1230000494	Internet Sale	99999999	8610	01JAN1998
6	1230000689	Internet Sale	99999999	19278	01JAN1998
7	1230000871	Internet Sale	99999999	28861	01JAN1998
8	1230001178	Internet Sale	99999999	57972	01JAN1998
9	1230001237	Internet Sale	99999999	62492	01JAN1998
10	1230001311	Internet Sale	99999999	78913	01JAN1998
11	1230001374	Internet Sale	99999999	11129	01JAN1998
12	1230001460	Internet Sale	99999999	25928	01JAN1998
13	1230001472	Internet Sale	99999999	32447	01JAN1998
14	1230002011	Internet Sale	99999999	9065	01JAN1998

Source for Order Item Information

The following ORDER_ITEM table contains information about orders placed with the company, and includes product ID, amount ordered, price of items, and other data.

Display 5.5 The ORDER_ITEM Table

View Data: ORDER_ITEM

#	Order Item Number	Product ID	Quantity Ordered	Total Retail Price for This Product
1	1	220101400065	3	$28.50
2	1	220100100228	2	$113.40
3	2	220101100031	2	$41.00
4	1	240100200004	1	$35.20
5	1	240200100007	1	$24.70
6	1	240200100224	1	$136.10
7	1	230100100012	2	$358.60
8	1	230100500068	1	$1.70
9	1	240400200093	1	$155.80
10	2	240400200106	1	$39.00
11	1	220200100166	2	$285.80
12	2	220200100224	1	$144.90
13	1	240400100015	2	$186.40
14	2	240400300035	1	$19.10
15	1	210200600055	1	$85.50

Source for Customer Information

The following CUSTOMER table contains information about the customers who are placing orders with the company. Information includes name, address, birthdate, and other data.

Display 5.6 The CUSTOMER Table

View Data: CUSTOMER

#	Customer Country	Customer Gender	Personal ID	Customer Name
1	rance	Male		Albert Collet
2	pain	Female		Mercedes Martínez
3	aly	Male		Pier Egidio Boeris
4	nited States	Male		James Kvarniq
5	nited States	Female		Sandrina Stephano
6	elgium	Male		Rent Van Lint
7	pain	Female		Julián Escorihuela Monserrate
8	inland	Male		Aki Ivonen
9	ermany	Female		Cornelia Krahl
10	nited States	Female		Karen Ballinger
11	ermany	Female		Elke Wallstab
12	nited States	Male		David Black
13	ermany	Male		Markus Sepke
14	rance	Male		Albert Eulert
15	aly	Female		Claudio Cambiaggi

In reviewing the previous tables, the data warehousing team identified the following issues:

- The salesperson and salesperson ID must be correlated to determine sales.
- The sales totals for each order must be correlated with the correct salesperson.
- The sales for each salesperson must be totaled.
- Some information does not exist in current source tables. New columns and tables must be created.

The next step is to specify the new tables that must be created in order to produce the desired reports.

Identifying Targets

To simplify the SAS Data Integration Studio job that will be used to create the desired report, the team decided to combine certain columns from existing tables into a smaller number of new tables:

- A new table will be created that joins the CUSTOMER, ORDERS, and ORDER_ITEMS tables: the ORDER_FACT table.
- A new table will be created that joins the STAFF and ORGANIZATION tables: the ORGANIZATION_DIM table.
- In order to answer the question of who made the most sales, the two new tables will be combined to create a third new table on which the report will be based. The third new table will be called Total_Sales_by_Employee.

By combining tables, the warehouse team can easily answer the specified question, as well as create a diverse range of reports to answer other business questions. Details about each new table are provided in the following sections.

Target That Combines Order Information

The ORDER_FACT table is created by joining the CUSTOMER, ORDERS, and ORDER_ITEMS tables. The new table will include all order data, including salesperson ID, customer, price, and quantity.

Target That Combines Organization Information

The ORGANIZATION_DIM table is created by joining the STAFF and ORGANIZATION tables. The new table will include all employee information including name, ID, salary, department, and managers.

Target That Lists Total Sales by Employee

The Total_Sales_by_Employee table is created by joining the ORDER_FACT table and ORGANIZATION_DIM table. It will be used to produce a report similar to the draft report shown in Display 5.1 on page 45. “Example: Using the Target Table Designer to Register SAS Tables” on page 140 describes how to specify metadata for the Total_Sales_by_Employee table. “Example: Creating a Job That Joins Two Tables and Generates a Report” on page 150 describes the job that uses the Total_Sales_by_Employee table to produce the report.

Creating the Report

For an example of the report can be created in SAS Data Integration Studio, see “Example: Creating a Job That Joins Two Tables and Generates a Report” on page 150.

What Are the Time and Place Dependencies of Product Sales?

Identifying Relevant Information

To answer the question, "What are the time and place dependencies of product sales?", the data warehousing team decided to design a report that reports sales across a time dimension and a geographic dimension. The following display shows an example of such a report.

Display 5.7 Time and Place Dependencies for Sales Report

Sum of Quantity			Year	Month Number				
			2001	2002			2002 Total *	Grand Total *
Country	State	Product Line		October	November	December		
Belgium		Sports	2861	140	188	432	2929	1003
Belgium Total *			5735	299	431	910	6531	2131
Germany		Sports	13448	819	913	2427	14296	6822
Germany Total *			38985	2159	2520	6300	41696	20120
Italy		Sports	4130	457	452	1243	7129	2166
Italy Total *			15868	1057	1243	3036	19720	7784
United Kingdom		Sports	12824	1088	1131	2275	15384	6314
United Kingdom Total *			33545	2078	2621	5285	38215	16869
United States	Colorado	Sports	366	12	16	43	252	155
	Colorado Total *		861	40	46	134	793	391
	Florida	Sports	2208	123	150	470	2415	1125
	Florida Total *		5050	314	339	977	5828	2588
	Illinois	Sports	1385	89	72	250	1490	725
	Illinois Total *		3157	202	196	610	3790	1743
	Michigan	Sports	1070	78	90	236	1308	616
	Michigan Total *		2706	170	213	519	3344	1517
United States Total *			73701	4424	5043	13766	85043	38162
Grand Total *			223036	13894	16489	39159	255362	1137518

The next step is to identify how such a report could be created.

Identifying Sources

Further questioning of the executive team revealed that it would be helpful to track sales across a customer dimension and an internal organization dimension as well as across the dimensions of time and geography. Questions that require multiple dimensions to be analyzed together can often be answered with online analytical processing (OLAP). Accordingly, the data warehousing team concluded that the question, "What are the time and place dependencies of product sales?", could be answered most efficiently with OLAP.

The data warehouse team examined existing tables to determine whether they could be used as inputs to an OLAP data store that would produce reports similar to the report shown in Display 5.7 on page 49. They identified several tables that could be used. These tables are described in the following sections.

Sources Related to Customers

The following tables can contribute to the customer dimension of the OLAP data store:

- CUSTOMER table
- CUSTOMER_TYPE table

Sources Related to Geography

The following tables can contribute to the geographic dimension of the OLAP data store:

- CONTINENT table
- COUNTRY table
- STATE table
- COUNTY table
- CITY table
- STREET_CODE table

Sources Related to Organization

The following tables can contribute to the organization dimension of the OLAP data store:

- STAFF table
- ORGANIZATION table

Sources Related to Time

The following tables can contribute to the time dimension of the OLAP data store:

- CONTINENT table
- COUNTRY table
- STATE table
- COUNTY table
- CITY table
- STREET_CODE table

While the previous tables contain the appropriate information, it is not in the correct format for OLAP. To support OLAP, several new data stores most be created, as described in the following section.

Identifying Targets

In order to support the OLAP reports such as the one shown in Display 5.7 on page 49, the data warehousing team specified the following new data stores:

- A SAS cube that will support OLAP reporting.
- A set of new tables that will form the central fact table and dimension tables for a star schema. Each new table will be created by joining two or more source tables that are related to a particular dimension, such as customers, geography, organization, and time.

The target tables are described in the following sections.

Target to Support OLAP

A SAS cube named Star will be created to support OLAP. This cube will support reports similar to Display 5.7 on page 49.

Target to Provide Input for the Cube

In this example, the ORDER_FACT table that is described in "Target That Combines Order Information" on page 48 is the central fact table in a star schema. Its dimension tables are described in the following sections.

Target That Combines Customer Information

The CUSTOMER_DIM table will be created by joining the tables described in "Sources Related to Customers" on page 49. In this example, CUSTOMER_DIM is one dimension of a star schema.

Target That Combines Geographic Information

The GEOGRAPHY _DIM table will be created by joining the tables described in "Sources Related to Geography" on page 50. In this example, GEOGRAPHY_DIM is one dimension in a star schema.

Target That Combines Organization Information

This dimension table is the same as "Target That Combines Organization Information" on page 48. In this example, ORGANIZATION_DIM is one dimension in a star schema.

Target That Combines Time Information

The TIME _DIM table will be created by joining the tables that are described in "Sources Related to Time" on page 50. In this example, TIME_DIM is one dimension in a star schema.

Building the Cube

For information about building cubes, see "Working with OLAP Cubes" on page 116.

The Next Step

After the questions, desired outputs, sources, and targets have been specified, administrators can begin setting up the servers, libraries, and other resources that SAS Data Integration Studio requires.

CHAPTER

6

Main Tasks for Administrators

Main Tasks for Installation and Setup

Overview of Installation and Setup

Administrators must complete several installation and setup tasks before users can begin work in SAS Data Integration Studio. This chapter describes the main installation and setup tasks, and it identifies documentation that describes these tasks in detail.

1. Review your project plans. For an overview of warehouse project plans, see "Planning a Data Warehouse" on page 41 and "Planning Security for a Data Warehouse" on page 42.
2. Use the SAS Software Installation wizard to install SAS Data Integration Studio and related software. For more information, see the *SAS Intelligence Platform: Installation Guide*. See also "Installing Software" on page 55.

3 Use the SAS Software Configuration wizard to specify the initial metadata for servers and users and to generate a set of configuration instructions that are customized for the current installation. For more information, see the configuration chapter in the *SAS Intelligence Platform: Administration Guide*. See also "Registering Servers" on page 56 and "Registering User Identities" on page 58.

4 Perform post-configuration tasks, such as setting up change management, registering libraries, and configuring SAS/ACCESS products. For more information, see the data administration chapters and the SAS Data Integration Studio chapter in the *SAS Intelligence Platform: Administration Guide*. See also "Creating Metadata Repositories" on page 55, "Registering Libraries" on page 59, and "Supporting Multi-Tier (N-Tier) Environments" on page 64.

Installing Software

For details about installing SAS software, see the *SAS Intelligence Platform: Installation Guide*. Your data warehouse project plan should identify the SAS software that is required for your site. For example, to answer the business questions that are described in Chapter 5, "Example Data Warehouse," on page 43, the software that is listed in the following table must be installed.

Table 6.1 Software Required to Create the Example Data Warehouse

Software	Required in Order to Perform These Tasks
SAS Management Console	Administer SAS software.
SAS Data Integration Studio	Consolidate and manage enterprise data from a variety of source systems, applications, and technologies.
SAS Metadata Server	Read and write metadata in a SAS Metadata Repository.
SAS Workspace Server	Access data and execute SAS code.
SAS OLAP Server	Create cubes and process queries against cubes.

Note: The data sources and targets for the example warehouse are assumed to be local, and all are assumed to be in Base SAS format or in comma-delimited format. △

Additional software would be required to access remote data or data that is under the control of SAS/SHARE, a SAS Scalable Performance Data (SPD) Server, a data base management system (DBMS), or an enterprise application. For more information about accessing remote data, see "Supporting Multi-Tier (N-Tier) Environments" on page 64. For more information about data management in general, see the data administration chapters in *SAS Intelligence Platform: Administration Guide*.

Creating Metadata Repositories

SAS Data Integration Studio enables you to create metadata objects that define sources, targets, and the transformations that connect them. These objects are saved to one or more metadata repositories. After a metadata server has been installed and started, one of the first tasks that an administrator must do is define one or more metadata repositories that are associated with the server.

Your data warehouse project plan should identify the metadata repositories that are required for your data warehouse. Typically, your metadata repositories will be under change management. Change management enables multiple SAS Data Integration

Studio users to work with the same metadata repository at the same time—without overwriting each other's changes.

For the example data warehouse, the following metadata repositories must be created:

- A foundation repository where all metadata about the example warehouse will be stored. This repository will be under change-management control. The repository will be named Foundation.
- A set of project repositories, one for each SAS Data Integration Studio user. Each project repository depends on (inherits metadata from) the foundation repository. Each project repository enables a user to check metadata out of the foundation repository. After changes are made to checked-out objects, or new metadata objects are added, the new or updated metadata is checked into the foundation repository. For the data warehouse example, each project repository will have a name such as Project: etlUser1.

For details about setting up change-managed repositories for SAS Data Integration Studio, administrators should see the change management section in the SAS Data Integration Studio chapter in the *SAS Intelligence Platform: Administration Guide*. In general, an administrator uses SAS Management Console to define a change-managed repository (such as a foundation repository) and one or more project repositories that depend on the change-managed repository. The administrator designates a SAS Data Integration Studio user as the owner of each project repository. Administrators with the appropriate privilege can update a change-managed repository directly, without having to work with a project repository.

Registering Servers

Server Metadata Required for the Example

The SAS Configuration Wizard in the SAS Software Navigator enables you to run a script that will automatically add metadata for some servers. Use SAS Management Console to add metadata that is not provided by the scripts in the SAS Software Navigator.

The following table summarizes how the servers for the example data warehouse would be made available to SAS Data Integration Studio users.

Table 6.2 Main Servers for the Example Data Warehouse

Software	Required in Order to Perform These Tasks	Where Metadata Is Specified
SAS Metadata Server	Read and write metadata in a SAS Metadata Repository.	Specified in the metadata profiles for administrators and users. Administrators should see "Creating a Metadata Profile (for Administrators)" on page 58. Users should see "Creating a Metadata Profile (for Users)" on page 94.
SAS Workspace Server	Access data and execute SAS code.	Can be specified as one component of the default SAS application server for SAS Data Integration Studio. See "Default SAS Application Server" on page 57.
SAS OLAP Server	Create cubes and process queries against cubes.	Can be specified as one component of the default SAS application server for SAS Data Integration Studio.

For details about entering metadata for the SAS Data Quality Servers or job scheduling servers from Platform Computing, see the appropriate sections in the SAS Data Integration Studio chapter in the *SAS Intelligence Platform: Administration Guide*.

For details about entering metadata for a SAS/SHARE server, a SAS Scalable Performance Data (SPD) Server, a DBMS, or an enterprise application, see the multi-tier environment section in SAS Data Integration Studio chapter and the Connecting to Common Data Sources chapter in the *SAS Intelligence Platform: Administration Guide*.

Default SAS Application Server

SAS Data Integration Studio enables users to select a default SAS application server. The default SAS application server enables SAS Data Integration Studio to execute SAS code, to access data, and to perform other tasks that require a SAS server—without having to specify a server each time.

When you select a default SAS application server, you are actually selecting a metadata object that can provide access to several servers, libraries, schemas, directories, and other resources. Typically, a metadata administrator defines the metadata for a SAS application server and tells users which object to select as the default in SAS Data Integration Studio.

For the example data warehouse, assume the metadata object for the default SAS application server is named SASMain. To support the example data warehouse, SASMain must include the following components:

- a SAS Workspace Server component
- a SAS OLAP Server component

To enter metadata for SAS application servers, follow the instructions that are provided by the SAS Configuration Wizard that is associated with the SAS Software Navigator. See also the configuring your SAS servers chapter in the *SAS Intelligence Platform: Administration Guide*. For more information about how the default SAS application server affects data access, see "Supporting Multi-Tier (N-Tier) Environments" on page 64.

Registering User Identities

In SAS Data Integration Studio, the metadata for users and groups is used to support change management, connections to a remote computer with SAS/CONNECT, and connections to a DBMS with SAS/ACCESS software.

Also, SAS Data Integration Studio users can select the metadata for a user or group and associate it with the metadata for a table, a job, or any other kind of object that can be displayed in the Inventory tree. To the metadata for a job, for example, you could add the metadata for the person who needs to be contacted if the job fails.

Your data warehouse project plan should identify the users and groups that are required for your data warehouse. For the example data warehouse, metadata for the following persons and groups must be added to the foundation repository:

- an administrator with the generic name Metadata Admin
- several SAS Data Integration Studio users with generic names such as etlUser1 and etlUser2
- a group for SAS Data Integration Studio users named ETL User Group

The metadata for each person or group specifies certain privileges. For example, the metadata for Metadata Admin specifies administrative privileges, such as the privilege to write metadata directly to a foundation repository or a custom repository without having to use a project repository. The metadata for ETL User Group specifies privileges for users who work under change management, and etlUser1, etlUser2, and other users are members of that group.

The SAS Configuration Wizard in the SAS Software Navigator enables you to run a script that will automatically add metadata for some users and groups. Use SAS Management Console to add metadata that is not provided by the scripts in the SAS Software Navigator. For details about entering metadata for users and administrators in a change-management context, see the change management section in the SAS Data Integration Studio chapter in the *SAS Intelligence Platform: Administration Guide*.

Creating a Metadata Profile (for Administrators)

After you start SAS Management Console, a window displays that has various options for maintaining a *metadata profile*. A metadata profile is a client-side definition of where the metadata server is located. The definition includes a machine name, a port number, and one or more metadata repositories. In addition, the metadata profile can contain log on information and instructions for connecting to the metadata server automatically. You cannot do any work in SAS Management Console, in SAS Data Integration Studio, or in related applications until you create and open the appropriate metadata profile.

For the example data warehouse, the following metadata profiles must be created:

- at least one metadata profile for an administrator
- a metadata profile for each SAS Data Integration Studio user. Users will create their own metadata profiles, as described in “Creating a Metadata Profile (for Users)” on page 94.

SAS Data Integration Studio users typically work under change-management control. However, some tasks in SAS Data Integration Studio, such as deploying a job for scheduling or generating a stored process for a job, cannot be done under change-management control. In contrast to most users, a SAS Data Integration Studio administrator has the following characteristics:

- a metadata identity that grants the privilege to write metadata directly to a foundation repository or to a custom repository without having to use a project repository

- a metadata profile that specifies the login and password from the administrator's metadata identity. The profile also specifies a foundation repository or a custom repository as the default repository.

For details about entering metadata for users and administrators in a change-management context, see the change management section in the SAS Data Integration Studio chapter in the *SAS Intelligence Platform: Administration Guide*.

Registering Libraries

Overview of Libraries

In SAS software, a library is a collection of one or more files that SAS recognizes and that SAS references and stores as a unit. SAS Data Integration Studio uses a combination of server metadata and library metadata to access the sources and targets that are referenced in SAS Data Integration Studio jobs. Accordingly, one of the first tasks for an administrator might be to specify metadata for the libraries that contain data stores or other resources.

Both SAS Management Console and SAS Data Integration Studio enable you to enter metadata for libraries. A typical approach would be for administrators to use SAS Management Console to add metadata for an initial set of libraries. SAS Data Integration Studio users would then use source designer wizards or target designer wizards to add metadata about specific tables in a library. Later, administrators and/or users could add metadata for other libraries as needed.

Entering metadata for a library does not, in itself, provide access to tables in the library. You must also specify metadata for all tables that you want to access in the library, as described in "Registering Sources and Targets" on page 97.

Which Libraries Are Needed?

Administrators should ask questions such as these to determine which libraries are needed for a given data warehouse:

- In what format are the source tables and target tables? Are they SAS files, Microsoft Excel files, DBMS tables, flat files, enterprise application files, or files in which values are separated with commas or other characters?
- If the tables are in SAS format, do the tables use column formats that are defined in a SAS format library?
- If the tables are in SAS format, will SAS/SHARE software be used to provide concurrent update access to the tables?
- If the tables are not in SAS format, how do you plan to access these tables? With a database library (SAS/ACCESS software for relational databases)? With an ODBC library (SAS/ACCESS for ODBC)? With the external file interface? With an enterprise application library (such as a library that uses SAS/ACCESS to R/3)?

Answers to questions such as these determine the type of library metadata that you need to enter.

Enter Metadata for a Library

The New Library wizard in SAS Management Console and SAS Data Integration Studio enables you to enter metadata about many different kinds of libraries. For details about entering metadata for different kinds of libraries, administrators should see the managing libraries chapter in the *SAS Management Console: User's Guide*.

The following steps summarize how to use SAS Data Integration Studio to enter metadata about a library. These steps are appropriate for an administrator who does not have to use the change-management facility. The steps for a user would be similar, except that the user would have to check in the metadata for the new library as a last step.

1. Start SAS Data Integration Studio as described in "Starting SAS Data Integration Studio" on page 93.
2. Open the metadata profile that specifies the repository where metadata for the new library should be stored. The steps for opening a metadata profile are described in "Opening a Metadata Profile" on page 95.
3. In SAS Data Integration Studio, click the **Inventory** tab to display the Inventory tree.
4. In the Inventory tree, expand the folders until the Libraries folder is displayed.
5. Select the **Libraries** folder, then select **File ▶ New** from the menu bar. The New Library wizard displays.
6. In the New Library wizard, expand the folders to view the folder for the type of library for which you want to enter metadata. (The wizard includes folders for **Database Libraries**, **Enterprise Application Libraries**, and **SAS Libraries**, for example.)
7. Expand the folder for the type of library for which you want to enter metadata, such as **SAS Libraries**.
8. Select the type of library for which you want to enter metadata, such as **SAS Base Engine Library** and click **Next**.
9. Enter metadata as prompted by the wizard.

After the metadata for a library has been entered and saved, it is available for use in SAS Data Integration Studio. For example, most source designer wizards and target designer wizards will prompt you to select the library that contains or will contain a given source table or target table.

Preassigned Libraries

It is possible to assign a SAS library to a server so that the library is assigned whenever the server is started. Such a library is said to be preassigned. Preassigned libraries are used whenever you want a SAS library to be available in the current session without explicitly assigning the library during the session.

For example, suppose that you wanted to use the View Data feature to display a table that contains custom SAS formats. The SAS library that contains the formats can be preassigned to the SAS application server that is used to access the table.

Some of the tasks that are associated with preassigning a SAS library must be done outside of SAS Data Integration Studio or SAS Management Console. For details, see the assigning libraries chapter in the *SAS Intelligence Platform: Administration Guide*.

The properties window for a library includes a **Library is Preassigned** check box. To display this check box, follow these steps:

1. From the SAS Data Integration Studio desktop, in the Inventory tree, open the Libraries folder and select the library that you want to view or update.
2. Select **File ▶ Properties** from the menu bar. The properties window for the library displays.
3. Select the **Options** tab, then click the **Advanced Options** button. The Advanced Options window is displayed.
4. Select the **Pre-Assign** tab. The **Library is Preassigned** check box is on that tab.

Note: Selecting the **`Library is Preassigned`** check box does not preassign the library. The check box indicates that the library has been preassigned, using the methods that are described in the multi-tier environments section in the administering SAS Data Integration Studio chapter in the *SAS Intelligence Platform: Administration Guide*. △

Libraries for the Example Warehouse

For the example data warehouse, assume that most data sources and targets are in Base SAS format, and that some of these tables use custom column formats that are stored in a SAS library. Accordingly, metadata for the following Base SAS libraries must added to the foundation repository:

- one or more Base SAS libraries for data sources
- one or more Base SAS libraries for data targets

The general steps for entering library metadata are described in "Enter Metadata for a Library" on page 59. You do not need to enter metadata for a library that contains SAS formats, but this library must be properly set up. See "Libraries for Custom SAS Formats" on page 62.

Assume that some of the source data for the example data warehouse is in comma-delimited files, and that the external file interface will be used to access these files. See "Supporting HTTP or FTP Access to External Files" on page 72.

Base SAS Libraries

To access tables in Base SAS format, metadata for the appropriate SAS libraries must be defined and saved to a metadata repository. To access the tables, SAS Data Integration Studio will use the default SAS application server or the server that is specified in the metadata for the library. The general steps for entering library metadata are described in "Enter Metadata for a Library" on page 59.

SAS/SHARE Libraries

A SAS/SHARE server enables multiple users to access a library concurrently. To access tables that are under the control of a SAS/SHARE server, metadata for the SAS/SHARE server and a SAS/SHARE library must be defined and saved to a metadata repository.

SAS SPD Server Libraries

SAS Scalable Performance Data (SPD) Server is a high-performance, multi-user, parallel-processing data server with a comprehensive security infrastructure, backup and restore utilities, and sophisticated administrative and tuning options. SAS SPD Server stores data in a special format that facilitates parallel processing. You can use SAS SPD Server to access tables in SAS SPD Server format, using a special SAS library that is designed for this purpose.

To use SAS SPD Server to access tables in SAS SPD Server format, metadata for SAS SPD Server and a SAS SPD Server library must be defined and saved to a metadata repository.

SAS SPD Engine Libraries

SAS Scalable Performance Data (SPD) Engine is included with Base SAS. It is a single-user data storage solution that shares the high performance, parallel processing, and parallel I/O capabilities of SAS SPD Server for managing large data volumes, but

without the additional complexity of a full server. SAS SPD Engine can read and write data stores in SPD Server format.

To use SAS SPD Engine to access tables in SAS SPD Server format, metadata for a SAS SPD Engine library must be defined and saved to a metadata repository. To access the tables, SAS Data Integration Studio will use the default SAS application server or the server that is specified in the metadata for the library.

Libraries for Custom SAS Formats

A format is an instruction that SAS uses to write data values. You use formats to control the written appearance of data values, or, in some cases, to group data values together for analysis. Some SAS tables use custom column formats that are stored in a SAS library.

Note: You do not need to enter metadata for a library that contains custom SAS formats. However, if a table uses custom formats that are stored in a SAS library, the library of formats must be available to the SAS application server that is used to display data in the table or to execute code for the table. △

For details about setting up a SAS format library, see the user-defined formats section in the connecting to common data sources chapter in the *SAS Intelligence Platform: Administration Guide*.

DBMS Libraries

To access tables in a database management system (DBMS) such as Oracle or DB2, metadata for the DBMS server and the appropriate DBMS library must be defined and saved to a metadata repository. See also the following sections about Open Database Connectivity (ODBC) libraries and OLE DB libraries.

ODBC Libraries

ODBC is an application programming interface (API). ODBC provides a standard interface for SQL databases. An application that uses the ODBC interface can connect to any database that has an ODBC driver, such as Microsoft Access, Microsoft Excel, Borland dBase, or IBM DB2. Use ODBC libraries when an interface for a specific DBMS is not available and the DBMS complies with ODBC.

To use ODBC to access tables, the following requirements must be met:

- The appropriate ODBC driver must be available on the computer where the database resides.
- Metadata for the ODBC database server must be available in a current metadata repository.

For example, assume that you want to access tables in a Microsoft Access database and the database resides on a computer with the Microsoft Windows XP Professional operating system. You could use the ODBC Data Source Administrator administrative tool to define Microsoft Access as a system data source on that computer. You might create a system data source called msdb, for example. (A system data source can be more useful than a user data source because it is available to all users and to NT services on that computer.)

After the ODBC data source has been defined, an administrator could use SAS Management Console to enter metadata for the ODBC database server. The ODBC database server is the computer where the ODBC-compliant database resides. The metadata for the ODBC database server includes the network address for that computer, as well as the relevant ODBC driver (such as the msdb data source) on that

computer. For details about entering metadata for servers, see the Help for the Server Manager plug-in to SAS Management Console.

OLE Libraries

OLE DB is a Microsoft API for access to different data sources. An OLE library uses the SAS/ACCESS interface for OLE DB providers. Use OLE libraries when an interface for a specific DBMS is not available and the DBMS complies with OLE DB.

Generic Libraries

When you display the New Library wizard from the SAS Data Integration Studio desktop, the first page of the wizard enables you to select the type of library that you want to create. If you cannot find an exact match for the kind of data that you want to access, you can select a **`Generic`** library.

A Generic library enables you to manually specify a SAS engine and the options that are associated with that engine. Because it is general by design, the Generic library offers few hints as to what options should be specified for a particular engine. Accordingly, the Generic library might be most useful to experienced SAS users. For details about the options for a particular engine, see the SAS documentation for that engine.

Note: SAS has several library templates for specific data formats. The specific library templates will often give better results than the generic template, which has not been optimized for particular data formats. Use the templates for a specific format whenever possible. △

Microsoft Excel and Microsoft Access Files

See “Supporting HTTP or FTP Access to External Files” on page 72.

XML Files

To provide access to one or more tables that are defined in an XML file, you could create a Generic library and specify options that are appropriate for the XML LIBNAME engine and the XML file. After the Generic library (with the XML options) is registered in a metadata repository, SAS Data Integration Studio users can use the Generic source designer to generate metadata for the tables that are defined in the XML file.

Libraries for Enterprise Applications

Optional data surveyor wizards can be installed in SAS Data Integration Studio that provide access to the metadata in enterprise applications such as PeopleSoft and SAP R/3. See the documentation for these wizards for details about any servers and libraries that must be defined to support the wizards.

Additional Information about Libraries

The Help for SAS Data Integration Studio contains additional information about libraries. To display the relevant Help topics, follow these steps:

1. From the SAS Data Integration Studio menu bar, select **Help ▶ Contents**. The Help window displays.
2. In the left pane of the Help window, select **Prerequisites ▶ Specifying Metadata for Libraries**.

Supporting Multi-Tier (N-Tier) Environments

Administrators set up servers, libraries and other resources so that SAS Data Integration Studio can access data and execute jobs across multiple tiers in the SAS Intelligence Platform.

Accessing Data in the Context of a Job

When code is generated for a job, it is generated in the current context. The context includes the default SAS application server when the code was generated, the credentials of the person who generated the code, and other information. The context of a job affects the way that data is accessed when the job is executed.

In the context of a job, local data is data that is addressable by the SAS Application Server when code was generated for the job. Remote data is data that is not addressable by the SAS Application Server when code was generated for the job.

For example, the following data would be considered local in the context of a job:

- data that can be accessed as if it were on the same computer(s) as the SAS Workspace Server component(s) of the default SAS application server
- data which is accessed with a SAS/ACCESS engine (used by the default SAS application server).

The following data would be considered remote in a SAS Data Integration Studio job:

- data that cannot be accessed as if it were on the same computer(s) as the SAS Workspace Server component(s) of the default SAS application server
- data that exists in a different operating environment from the SAS Workspace Server component(s) of the default SAS application server (such as MVS data that is accessed by servers running under Microsoft Windows)

Note: Avoid or minimize remote data access in the context of a SAS Data Integration Studio job. △

Remote data has to be moved because it is not addressable by the relevant components in the default SAS application server at the time that the code was generated. SAS Data Integration Studio uses SAS/CONNECT and the UPLOAD and DOWNLOAD procedures to move data. Accordingly, it can take longer to access remote data than local data, especially for large data sets. It is especially important to keep an understanding of where the data is located when using advanced techniques such as parallel processing because the UPLOAD and DOWNLOAD procedures would run in each iteration of the parallel process.

For information about accessing remote data in the context of a job, administrators should see “Multi-Tier Environments” in the SAS Data Integration Studio chapter in the *SAS Intelligence Platform: Administration Guide*. Administrators should also see “Deploying a Job for Execution on a Remote Host” on page 68.

Interactive Access to Data

When SAS Data Integration Studio is used to access information interactively, the server that is used to access the resource must be able to resolve the physical path to the resource. The path can be a local path or a remote path, but the relevant server must be able to resolve the path. The relevant server is the default SAS application server, a server that has been selected, or a server that is specified in the metadata for the resource.

For example, in the source designers for external files, the Server tab in the Advanced File Location Settings window enables you to specify the SAS application

server that is used to access the external file. This server must be able to resolve the physical path that you specify for the external file.

As another example, suppose that you use the View Data option to view the contents of a table in the Inventory tree. To display the contents of the table, the default SAS application server or a SAS application server that is specified in the library metadata for the table must be able to resolve the path to the table.

For the relevant server to resolve the path to a table in a SAS library, one of the following conditions must be met:

- The metadata for the library does not include an assignment to a SAS application server, and the default SAS application server can resolve the physical path that is specified for this library.
- The metadata for the library includes an assignment to a SAS application server that contains a SAS Workspace Server component, and the SAS Workspace Server is accessible in the current session.
- The metadata for the library includes an assignment to a SAS application server, and SAS/CONNECT is installed on both the SAS application server and the machine where the data resides.

For more information about configuring SAS/CONNECT to access data on a machine that is remote to the default SAS application server, administrators should see "Multi-Tier Environments" in the SAS Data Integration Studio chapter in the *SAS Intelligence Platform: Administration Guide*.

Note: If you select a library that is assigned to an inactive server, you receive a Cannot connect to workspace server error. Check to be sure that the server assigned to the library is running and is the active server. △

Execute a Job on a Remote Host

See "Deploying a Job for Execution on a Remote Host" on page 68.

Deploying a Job for Scheduling

Preparation

After you have verified that a SAS Data Integration Studio job runs successfully, you might want to schedule it to run in batch mode at a specified date and time. Scheduling a job is a two-stage process as follows:

1. An administrator deploys the job for scheduling. Code is generated for the job and the generated code is saved to a file. Metadata about the deployed job is saved to the current metadata repository.
2. The job is scheduled using the Schedule Manager plug-in to SAS Management Console or another scheduler.

If possible, test a job and verify its output before deploying the job for scheduling.

A job cannot be deployed for scheduling unless it is checked in. If you attempt to deploy a job that has been checked out, deployment fails and an error message displays. The error message indicates that the job is locked by a user.

You cannot deploy a job for scheduling unless your metadata profile enables you to connect to SAS Data Integration Studio without change-management control. For

details about administrative profiles, see “Creating a Metadata Profile (for Administrators)” on page 58. It is assumed that the prerequisites for job scheduling have been met, as described in the SAS Data Integration Studio chapter in the *SAS Intelligence Platform: Administration Guide*.

Deploy a Job for Scheduling

Assume that you have tested a SAS Data Integration Studio job and verified its output and you are ready to deploy the job for scheduling in SAS Management Console.

1. Start SAS Data Integration Studio. A window displays that has various options for maintaining a metadata profile.
2. Select a profile that will enable you to connect to SAS Data Integration Studio without change-management control.
3. In the tree view, select the **Inventory** tab.
4. In the Inventory tree, expand the **Jobs** group.
5. Select one or more jobs that you want to deploy, right-click to display the pop-up menu, and select **Deploy for Scheduling**. The Deploy for Scheduling window displays.

   If you selected only one job, the file name for that job appears in the **Directory Path** field in the Deploy for Scheduling window. If you selected more than one job, no file names appear in the **Directory Path** field in the Deploy for Scheduling window. However, a separate file will be created for each job that is selected. Each deployed job file will be named after the corresponding job.
6. In the **SAS server** field, accept the default server or select the server that will be used to generate and store code for the selected job.
7. The next step is to select the job deployment directory. One or more job deployment directories were defined for the selected server when the metadata for that server was created. Click the **Select** button. The Deployment Directories window displays.
8. In the Deployment Directories window, select a directory where the generated code for the selected job will be stored, then click **OK**. You are returned to the Deploy for Scheduling window. The directory that you selected is specified in the **Directory name** field.

   If you selected one job, in the **File name** field you can edit the default name of the file that will contain the generated code for the selected job. The name must be unique in the context of the directory name that is specified above.
9. When you are ready to deploy the job or jobs, click **OK**. Code is generated for the selected job or jobs and is saved to the directory that is specified in the **Directory name** field. Metadata about the deployed jobs is saved to the current metadata repository. A status window displays. It states whether the deployment was successful or not.

   The icon next to the jobs that you selected changes to indicate that the jobs are now available for scheduling in SAS Management Console. A small clock is added to the job icon.

Next Tasks

After a job has been deployed from SAS Data Integration Studio, you can schedule the job by using the Schedule Manager plug-in to SAS Management Console or another scheduler. For details about the Schedule Manager plug-in, see its Help or see the *SAS Management Console: User’s Guide*. See also the chapter about SAS scheduling in the *SAS Intelligence Platform: Administration Guide*.

If you deploy a job for scheduling and then update the job, you must redeploy the job so that the latest version of the job is scheduled. For details, see redeploying jobs for scheduling topic in the Help.

Note: The code that is generated for a job contains the credentials of the person who created the job. If a person's credentials have changed, and a deployed job contains outdated user credentials, the deployed job will fail to execute. The remedy is to redeploy the job with the appropriate credentials. △

Additional Information About Job Scheduling

The Help for SAS Data Integration Studio contains additional information about job scheduling. To display the relevant Help topics, follow these steps:

1. From the SAS Data Integration Studio menu bar, select **Help ▶ Contents**. The Help window displays.
2. In the left pane of the Help window, select **Task Overviews ▶ SAS Data Integration Studio Task Reference ▶ Maintaining Stored Processes**.

Deploying a Job for Execution on a Remote Host

Preparation

Administrators can use the Deploy for Scheduling window to deploy a job for execution on a host that is remote from the default SAS application server. Code is generated for the job and the generated code is saved to a file. Metadata about the deployed job is saved to the current metadata repository. After a job has been deployed to the remote host, it can be executed by any convenient means.

Remember that the generated code is local to the server that is selected in the Deploy for Scheduling window. For example, suppose that the default SAS application server for SAS Data Integration Studio is named SASMain, but you want a job to execute on another SAS application server that is named DEZ_App Server. You would select DEZ_App Server in the Deploy for Scheduling window, and the code that is generated for the job would be local to DEZ_App Server.

If possible, test a job and verify its output before deploying the job.

A job cannot be deployed unless it is checked in. If you attempt to deploy a job that has been checked out, deployment fails and an error message displays. The error message indicates that the job is locked by a user.

You cannot deploy a job unless your metadata profile enables you to connect to SAS Data Integration Studio without change-management control. For details about administrative profiles, see "Creating a Metadata Profile (for Administrators)" on page 58.

A SAS Workspace Server and a SAS Data Step Batch Server must be configured on the remote host. For information about this configuration, administrators should see "Multi-Tier Environments" in the SAS Data Integration Studio chapter in the *SAS Intelligence Platform: Administration Guide*.

Task Summary

1. Start SAS Data Integration Studio. A window displays that has various options for maintaining a metadata profile.
2. Select a profile that will enable you to connect to SAS Data Integration Studio without change-management control.
3. In the tree view, select the **Inventory** tab.
4. In the Inventory tree, expand the **Jobs** group.
5. Select one or more jobs that you want to deploy, right-click to display the pop-up menu, and select **Deploy for Scheduling**. The Deploy for Scheduling window displays.

   If you selected only one job, the file name for that job appears in the **Directory Path** field in the Deploy for Scheduling window. If you selected more than one job, no file names appear in the **Directory Path** field of the Deploy for Scheduling window. However, a separate file will be created for each job that is selected. Each deployed job file will be named after the corresponding job.
6. In the **SAS server** field, select the server that is to used to execute the selected job. In this example, you would not accept the default but would select the remote server, DEZ_App Server.

7 The next step is to select the job deployment directory. One or more job deployment directories were defined for the selected server when the metadata for that server was created. Click the **Select** button. The Deployment Directories window displays.

8 In the Deployment Directories window, select a directory where the generated code for the selected job will be stored, then click **OK**. You are returned to the Deploy for Scheduling window. The directory that you selected is specified in the **Directory name** field.

   If you selected one job, in the **File name** field you can edit the default name of the file that will contain the generated code for the selected job. The name must be unique in the context of the directory name that is specified above.

9 When you are ready to deploy the job or jobs, click **OK**. Code is generated for the selected job or jobs and is saved to the directory that is specified in the **Directory name** field. Metadata about the deployed jobs is saved to the current metadata repository. A status window displays. It states whether the deployment was successful or not.

   The icon next to the jobs that you selected changes to indicate that the jobs have been deployed to the remote server.

Next Tasks

After a job has been deployed to the remote host, it can be executed by any convenient means.

Converting Jobs into Stored Processes

About Stored Processes

SAS Data Integration Studio enables you to create jobs that perform various data integration tasks. You can execute jobs immediately, or you can save them to a file so that a stored process server can execute them later.

A stored process is a SAS program that is stored on a server and can be executed as required by requesting applications. You can use stored processes for Web reporting, analytics, building Web applications, delivering result packages to clients or to the mid-tier, and publishing results to channels or repositories. Stored processes can also access any SAS data source or external file and create new data sets, files, or other data targets supported by SAS.

Administrators can generate one or more stored processes for a job that is selected in the Inventory tree or the Custom tree on the SAS Data Integration Studio desktop. Code is generated for the stored process and the generated code is saved to a file. Metadata about the stored process is saved in the current metadata repository.

After the metadata has been saved to a repository, other applications can connect to the repository and access the stored processes that are defined there—if they have appropriate privilege. For example, a stored process that was generated for a SAS Data Integration Studio job could be executed from the following applications:

- SAS Add-In for Microsoft Office, a Component Object Model (COM) add-in that extends Microsoft Office by enabling you to dynamically execute stored processes and embed the results in Microsoft Word documents and Microsoft Excel spreadsheets.

- SAS Enterprise Guide, an integrated solution for authoring, editing, and testing stored processes.
- SAS Information Map Studio, an application that can be used to implement information map data sources. Stored processes can use the full power of SAS procedures and the DATA step to generate or update the data in an information map.
- SAS Information Delivery Portal, a portal that provides integrated Web access to SAS reports, stored processes, information maps, and channels.
- Stored Process Service, a Java API that enables you to execute stored processes from a Java program.
- SAS Stored Process Web applications, Java Web applications that can execute stored processes and return results to a Web browser.
- SAS BI Web Services, a Web service interface to SAS stored processes.

Prerequisites for Stored Processes

SAS Data Integration Studio enables administrators to generate a stored process for a job. Code is generated for the stored process, and the generated code is saved to a file. Metadata about the stored process is saved in the current metadata repository. The stored process feature in SAS Data Integration Studio requires the following components:

- A SAS Stored Process Server. For details about how this server is installed and configured, see the *SAS Intelligence Platform: Administration Guide*.
- A source repository for the stored processes that are generated from SAS Data Integration Studio jobs.

A source repository is a location, such as a directory, that contains stored processes. Users who will execute stored processes must have the appropriate access rights to the source repository, as defined by the operating system. The stored process feature in SAS Data Integration Studio is typically used in conjunction with the Stored Process Manager plug-in to SAS Management Console. The Stored Process Manager plug-in can be installed with SAS Foundation Services.

To use the stored process feature to your best advantage, you should be familiar with stored process parameters, input streams, and result types. For a detailed discussion of stored processes, see the SAS stored processes section in the *SAS Integration Technologies: Developer's Guide*.

Preparation

It is assumed that the prerequisites for stored processes have been met. You must have write access to the source repository for the stored processes that are generated from SAS Data Integration Studio jobs.

You cannot generate a stored process for a job unless your metadata profile enables you to connect to SAS Data Integration Studio without change-management control. For details about administrative profiles, see "Creating a Metadata Profile (for Administrators)" on page 58.

Generate a Stored Process for a Job

Follow these steps to generate a stored process for a job:

1. Start SAS Data Integration Studio. A window displays that has various options for maintaining a metadata profile.

2 Select a profile that enables you to connect to SAS Data Integration Studio without change-management control.

3 In the Inventory tree or in the Custom tree on the SAS Data Integration Studio desktop, right-click the job for which you want to generate a stored process. Then select **Stored Process ▶ New** from the pop-up menu. The New Stored Process wizard displays.

   A tree of folders displays, which you can use to organize the metadata for different kinds of stored processes. The folders are viewed and maintained in the New Stored Process wizard and in the Stored Process Manager plug-in to SAS Management Console. You can add a new folder by selecting **`Add Folder`**.

4 Select the folder where the metadata for the new stored process should be displayed. When you are finished, click **`Next`**.

5 Enter a descriptive name for the stored process metadata. You could use a variation of the job name.

6 Enter other information as needed. For Help fields in this window, select the Help button. When finished, click **`Next`**.

7 Specify a SAS server, a source repository, a source file name, any input stream, and any output type (result type) for the new stored process. When finished, click **`Next`**.

8 Specify any parameters for the stored process. When finished, click **`Finish`**.

A stored process is generated for the current job and saved to the source repository. Metadata about the stored process is saved to the current metadata repository. The icon next to the job you selected changes to indicate that the job has a stored process.

Next Tasks

After the metadata for a stored process has been saved to a repository, other applications can connect to the repository and use the metadata to execute the stored process—if they have appropriate privilege. You can also view or update the properties for the stored process, as described in Viewing or Updating Stored Process Metadata.

Additional Information About Stored Processes

The Help for SAS Data Integration Studio contains additional information about stored processes. To display the relevant Help topics, follow these steps:

1 From the SAS Data Integration Studio menu bar, select **Help ▶ Contents**. The Help window displays.

2 In the left pane of the Help window, select **Task Overviews ▶ SAS Data Integration Studio Task Reference ▶ Maintaining Stored Processes**.

Metadata Administration

For details about maintaining the SAS Metadata Server, promoting and replicating metadata, and similar tasks, see the metadata administration chapters in the *SAS Intelligence Platform: Administration Guide*.

Supporting HTTP or FTP Access to External Files

An external file is a file that is maintained by the machine operating environment or by a software product other than SAS. A flat file with comma-separated values is one example.

SAS Data Integration Studio provides the following three source designer wizards that enable you to create metadata objects for external files:

- Delimited External File wizard for external files in which data values are separated with a delimiter character. Enables you to specify multiple delimiters, nonstandard delimiters, missing values, and multi-line records.
- Fixed Width External File wizard for external files in which data values appear in columns that are a specified number of characters wide. Enables you to specify non-contiguous data.
- User Written External File wizard for complex external files that require user-written SAS code to access their data.

The Delimited External File wizard and the Fixed Width External File wizard prompt you to specify the physical path to an external file. By default, a SAS application server is used to access the file. However, you can access the file with an HTTP server, HTTPS server, or FTP server if the metadata for that server is available in a current metadata repository. For details about defining metadata for an HTTP server, HTTPS server, or an FTP server, administrators should see the SAS Data Integration Studio chapter in the *SAS Intelligence Platform: Administration Guide*.

Supporting SAS Data Quality

As described in "Working With SAS Data Quality Software" on page 104, SAS Data Integration Studio has several features that can help you to improve the quality of your data. Except for the Data Validation transformation, these features require your site to license SAS Data Quality Server software and to complete some configuration tasks. For more information, see the SAS Data Quality Server section in the SAS Data Integration Studio chapter in the *SAS Intelligence Platform: Administration Guide*. See also the *SAS Data Quality Server: Reference*, which is available in the online SAS Help and Documentation for Base SAS and in the SAS OnlineDoc CD-ROM.

Supporting Metadata Import and Export

SAS Data Integration Studio and SAS Management Console include wizards that enable you to import metadata from—and to export metadata to—other applications that support the Common Warehouse Metamodel (CWM) format. For example, it is possible to import a data model for a set of sources or targets using the Metadata Importer wizard. If the model to be imported is not in CWM format, you must install optional bridge software from Meta Integration Technology, Inc. For details, see "Importing and Exporting Metadata" on page 98.

Supporting Case and Special Characters in Table and Column Names

Overview of Case and Special Characters

SAS Data Integration Studio cannot access tables or columns with case-sensitive names or with special characters in the names unless the appropriate options have been specified. For the example data warehouse, assume that all tables are in SAS format, and that all names for tables and columns follow the standard rules for SAS names.

Case and Special Characters in SAS Table and Column Names

By default, the names for SAS tables and columns must follow the standard rules for SAS names. However, SAS Data Integration Studio will support case-sensitive names for tables and columns, as well as special characters in column names, if the appropriate options are specified in the metadata for the SAS table.

SAS Data Integration Studio users can set name options in the metadata for individual tables. For a description of this task, see "Setting Name Options for Individual Tables" on page 109.

As an alternative to setting name options in the metadata for individual tables, you can set default name options for all table metadata that is entered with a source designer or a target designer in SAS Data Integration Studio. For details, see "Setting Default Name Options for Tables and Columns" on page 74.

Case and Special Characters in DBMS Table and Column Names

SAS Data Integration Studio cannot access a DBMS table with case-sensitive names or with special characters in names unless the appropriate name options are specified in the metadata for the database library that is used to access the table, and in the metadata for the table itself.

One approach would be for administrators to specify name options in the metadata for the database library, as described in this section. Administrators could then let SAS Data Integration Studio users know which DBMS name options to specify in the metadata for tables in that library. SAS Data Integration Studio users can set name options in the metadata for DBMS tables. For a description of this task, see "Setting Name Options for Individual Tables" on page 109.

As an alternative to setting name options in the metadata for individual tables, you can set default name options for all table metadata that is entered with a source designer or a target designer in SAS Data Integration Studio. For details, see "Setting Default Name Options for Tables and Columns" on page 74.

Enabling DBMS Name Options for a New Database Library

The following steps describe how to enable name options when you enter the metadata for a new database library. These steps are appropriate for an administrator who does not have to use the change-management facility. The steps for a user would be similar, except that the user would have to check in the metadata for the new library as a last step.

1 Follow the steps in "Enter Metadata for a Library" on page 59. In the New Library wizard, select the appropriate kind of database library and click **Next**.

2 Enter a name for the library and click **Next**.
3 Enter a SAS LIBNAME for the library, then click **Advanced Options**. The Advanced Options window displays.
4 In the Advanced Options window, click the **Output** tab.
5 To preserve DBMS column names, select **Yes** in the **Preserve column names as in the DBMS** field.
6 Click the **Input/Output** tab.
7 To preserve DBMS table names, select **Yes** in the **Preserve DBMS table names** field.
8 Click **OK** and enter the metadata as prompted by the wizard.

Enabling DBMS Name Options for an Existing Database Library

The following steps describe one way to update the existing metadata for a database library in order to specify name options. These steps are appropriate for an administrator who does not have to use the change-management facility. The steps for a user would be similar, except that the user would have to check out the library, update the metadata as described in the following steps, then check in the metadata for the library as a last step.

1 Start SAS Data Integration Studio as described in "Starting SAS Data Integration Studio" on page 93.
2 Open the metadata profile that specifies the repository where metadata for the library is stored. The steps for opening a metadata profile are described in "Opening a Metadata Profile" on page 95.
3 In SAS Data Integration Studio, click the **Inventory** tab to display the Inventory tree.
4 In the Inventory tree, expand the folders until the Libraries folder is displayed.
5 Select the **Libraries** folder, then select the library whose metadata must be updated.
6 Select **File ▶ Properties** from the menu bar. The properties window for the library displays.
7 In the properties window, click the **Options** tab.
8 On the **Options** tab, click **Advanced Options**. The Advanced Options window displays.
9 In the Advanced Options window, click the **Output** tab.
10 To preserve DBMS column names, select **Yes** in the **Preserve column names as in the DBMS** field.
11 Click the **Input/Output** tab.
12 To preserve DBMS table names, select **Yes** in the **Preserve DBMS table names** field.
13 Click **OK** twice to save your changes.

Setting Default Name Options for Tables and Columns

You can set default name options for all table metadata that is entered with a source designer wizard or a target designer wizard in SAS Data Integration Studio. These defaults apply to tables in SAS format or in DBMS format.

Note: For details about these defaults as they relate to SAS tables, see "Case and Special Characters in SAS Table and Column Names" on page 73. △

Defaults for table and column names can make it easier for users to enter the correct metadata for tables. Administrators still have to set name options on database libraries, and users should at least verify that the appropriate name options are selected for a given table.

The following steps describe how to set default name options for all table metadata that is entered with a source designer wizard or a target designer wizard in SAS Data Integration Studio.

1. Start SAS Data Integration Studio.
2. Open the metadata profile that specifies the repository where metadata for the tables is stored.
3. On the SAS Data Integration Studio desktop, select **Tools ▶ Options** from the menu bar. The Options window is displayed.
4. In the Options window, select the **General** tab.
5. On the **General** tab, select **Enable case-sensitive DBMS object names** to have source designers and target designers support case-sensitive table and column names by default.
6. On the **General** tab, select **Enable special characters within DBMS object names** to have source designers and target designers support special characters in table and column names by default.
7. Click **OK** to save any changes.

Maintaining Generated Transformations

Overview of Generated Transformations

One of the easiest ways to customize SAS Data Integration Studio is to write your own generated transformations. The Transformation Generator wizard guides you through the steps of specifying SAS code for the transformation and saving the transformation in a current metadata repository. After the transformation is saved and checked in, it is displayed in the Process Library tree, where it is available for use in any job.

As you use the wizard, you can define options that help users to enter the correct values in the property window for your generated transformation. For example, you could define an option that only accepts integers within a certain range. This option appears on the **Options** tab of the property window for the transformation. When users open the property window for your transformation and go to the **Options** tab, they would have to enter an integer value in the correct range in the field for this option.

Example: Creating a Generated Transformation

Preparation

For this example, assume that a SAS data set named TigersHitting2002 contains batting statistics for a baseball team. The following display shows the content and structure of this data set.

Display 6.1 Contents of Data Set TigersHitting2002

View Data: TIGERSHITTING2002 (rows 1-5) of 5

Starting row: 1 Go to row

#	Name	G	AB	R	H	HR	TB	RBI	BB	SO
1	Smithy Jones	158	548	90	179	26	294	100	107	89
2	Gary Troy	135	492	82	151	25	252	84	72	53
3	Rafael Fernando	154	636	95	175	8	246	47	43	114
4	Andy Hitfield	154	560	91	148	35	287	94	83	135
5	Vinny Toredo	143	543	56	126	12	189	61	22	69

The goal is to create a transformation template that takes a SAS data set as input and produces a report. The report displays a user-defined title, displays a user-defined set of columns, and calculates the sum of the values in one column of the table. The following display shows the kind of output that is desired.

Display 6.2 Tigers Hitting Statistics 2002

```
            Tigers Hitting Statistics 2002                          5
                                          12:44 Tuesday, September 27, 2005

 Obs    Name                 G     AB      HR    RBI

  1     Smithy Jones       158    548      26    100
  2     Gary Troy          135    492      25     84
  3     Rafael Fernando    154    636       8     47
  4     Andy Hitfield      154    560      35     94
  5     Vinny Toredo       143    543      12     61
                                          ===
                                          106
```

Assume the following about the current example:

- The main metadata repository is under change-management control. In this example, however, assume that an administrator is creating the new template, so the template would be added to directly to the Process Library tree, without having to be checked in. For details about change management, see “Working with Change Management” on page 113.
- You have selected a default SAS application server for SAS Data Integration Studio, as described in “Selecting a Default SAS Application Server” on page 96.

Start SAS Data Integration Studio and Open the Appropriate Metadata Profile

Follow these steps to begin work in SAS Data Integration Studio:

1 Start SAS Data Integration Studio.

2 Open the appropriate metadata profile. For the current example, the metadata profile is for an administrator who has the appropriate level of privilege to directly update metadata in the main metadata repository, without having to work through a project repository.

The next task is to display the Transformation Generator wizard.

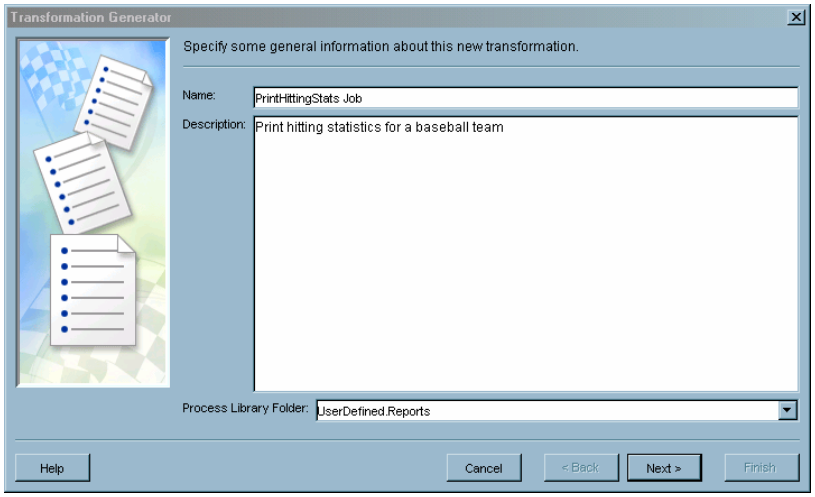
Display the Transformation Generator Wizard

Follow these steps to display the wizard that will guide you through the process of creating a user-defined, generated transformation.

1 On the SAS Data Integration Studio desktop, select **Tools ▶ Transformation Generator** from the menu bar. The first window of the wizard displays.

2 Enter a name and a description for the new transformation template, as shown in the following display.

Display 6.3 Transformation Generator Window



3 Specify the folder in the Process Library tree in which you want to store the new transformation. You do this by specifying a relative path from the Process Library folder to the directory that will hold the transformation. If the path contains two or more directory levels, separate directory level names with a period. For example, UserDefined.Reports.

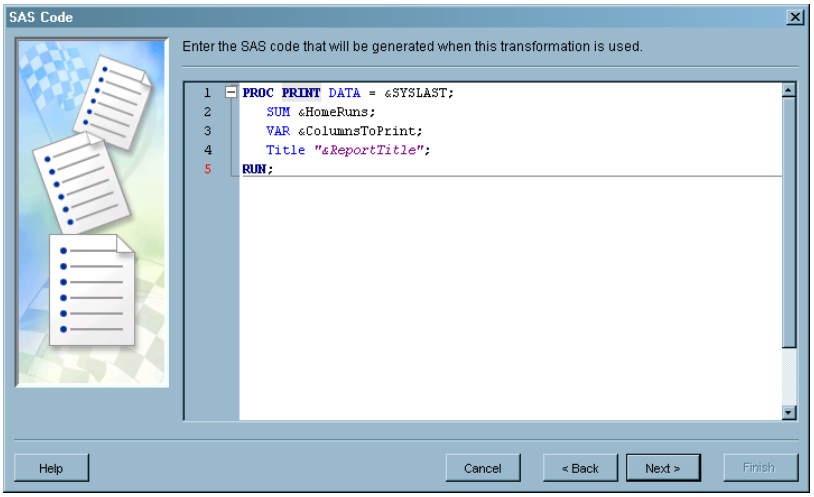
4 Click **`Next`**.

The next task is to specify SAS code for this transformation.

Specify SAS Code for the Transformation

In the SAS Code window, enter SAS code for the transformation. The following display shows the code that could be entered for the current example.

Display 6.4 Sample Transformation Code



A number of macro variables appear in the code. The variable &SYSLAST is a system variable that refers to the last data set created. The &SYSLAST variable enables a transformation in a process flow diagram to use the output from the previous transformation.

The other variables that are shown in the previous display, such as &ColumnsToPrint, are user-defined variables. Any user-defined variables must be defined in the Create Option window that is displayed later in the wizard.

After you have finished writing your SAS code, click **Next**. The Options window displays.

Note: The user of the transformation supplies values for these user-defined variables when the transformation is included in a job. In addition, the % character is used to escape single quotation marks, double quotation marks, and the %, (, and) characters. The practice of escaping these characters prevents problems with mismatched quotation marks or parentheses. You need to use the %unquote() function when you use a macro variable that contains SAS code. △

Create User-Defined Variables

In the Create Option window, define any user-defined variables that you used in the SAS Code window. The following table shows the values that you would enter for the user-defined variables that are shown in Display 6.4 on page 78.

Table 6.3 User-Defined Variables from the Create Option Window

Option Name	Macro Variable	Description	Type
Home runs	HomeRuns	Home runs hit	String
Columns to print	ColumnsToPrint	Name of the columns to print	Column
Report title	ReportTitle	Title of the report	String

The Create Option window enables you to specify 10 types of variables:

- String (Default)
- Boolean True/False
- Boolean Yes/No
- Column
- File
- Float
- Integer
- Library
- Lookup
- Table

The variables that you define in the Create Option window are used in the transformation template that you are creating. For example, string variables appear on the **Options** tab in the properties window for the PrintHittingStatistics template. Users display the **Options** tab in the window and enter values for each option.

The column variables appear on the **Column Options** tab in the properties window for the transformation template that you are creating. For example, users display the **Column Options** tab in the properties window for the PrintHittingStatistics template and select the columns that correspond to the ColumnsToPrint variable of the Column type.

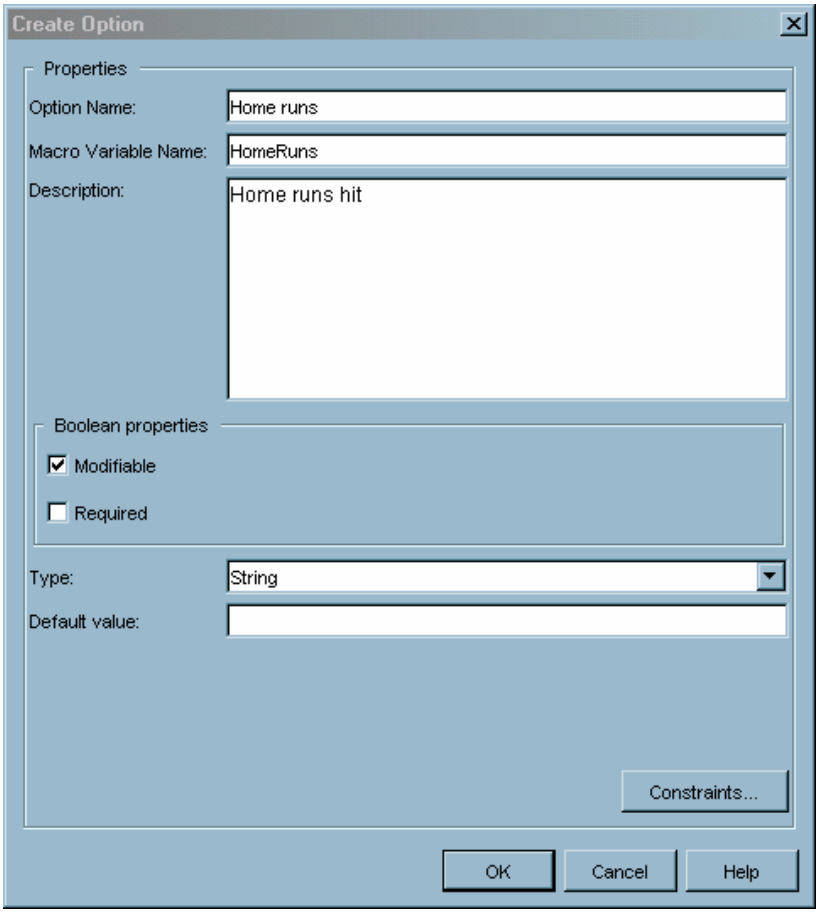
For details about these option types and the constraints that can be set for most types, see the Help topic in the Create/Edit Option window, which is one of the windows in the Transformation Generator wizard. (You display this window by clicking the **New** button in the Options window of the Transformation Generator wizard.)

To define the **Home runs** option, follow these steps:

1. Click **New**. A new row displays in the options table.
2. Enter *Home runs* in the **Option Name** field.
3. Enter *HomeRuns* in the **Macro Variable Name** field.
4. Enter a description of the variable in the **Description** field.
5. Select the **Modifiable** option in the **Boolean properties** check box
6. Keep the default value of **String** in the **Type** field. You must connect **Home runs** option to the column for home runs in the source table. Because you are accepting the default value of *String* in the **Type** field, users of the transformation can enter the character-based name of the column when they configure the transformation in a job.
7. Click **OK** to save the settings. The first row of the options list displays in the Options window.

8 The following display shows the Create Option window for the **Home runs** option.

Display 6.5 Create Option Window for the Home Runs Option



Click the **New** button to return to the Create Option window. To define the **Columns to print**, follow these steps:

To define the **Columns to print** option, follow these steps:

1 Click **New**. A new row displays in the options table.

2 Enter *ColumnsToPrint* in the **Option Name** field.

3 Enter *Names of the columns to print* in the **Macro Variable Name** field.

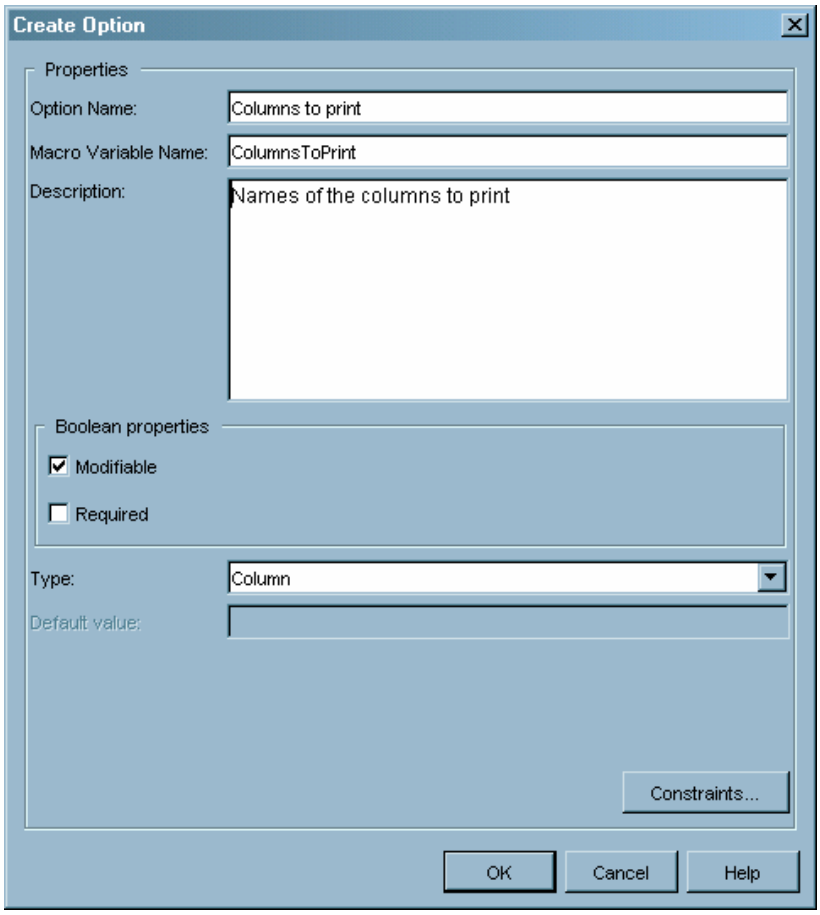
4 Enter a description of the variable in the **Description** field.

5 Select the **Modifiable** check box in the **Boolean properties** group box.

6 Keep the default value of **Column** in the **Type** field.

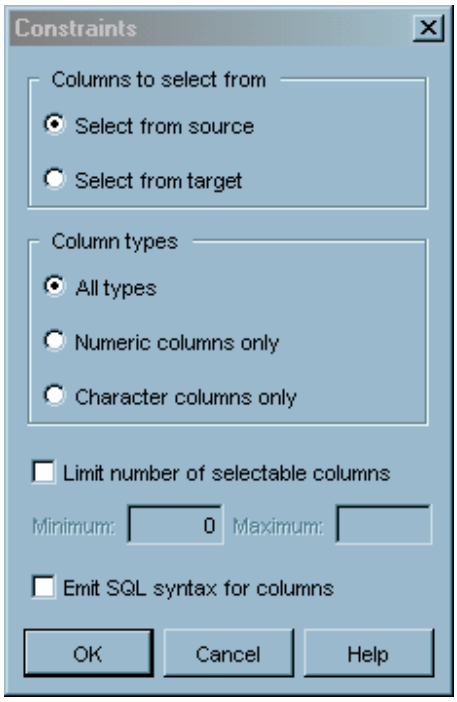
7 The following display shows the Create Option window **`Columns to print`** option.

Display 6.6 Create Option Window for the Columns To Print Option



8 Click **Constraints** to set constraints for the ColumnsToPrint option. The following display shows the Constraints window for the **Columns to print** option.

Display 6.7 Constraints Window



To define the constraints on the **Columns to print** option, follow these steps:

1 Select the **Select from source** check box in the **Boolean properties** group box.
2 Select the **All types** check box in the **Column types** group box.
3 Click **OK** to return to the Create Option window.

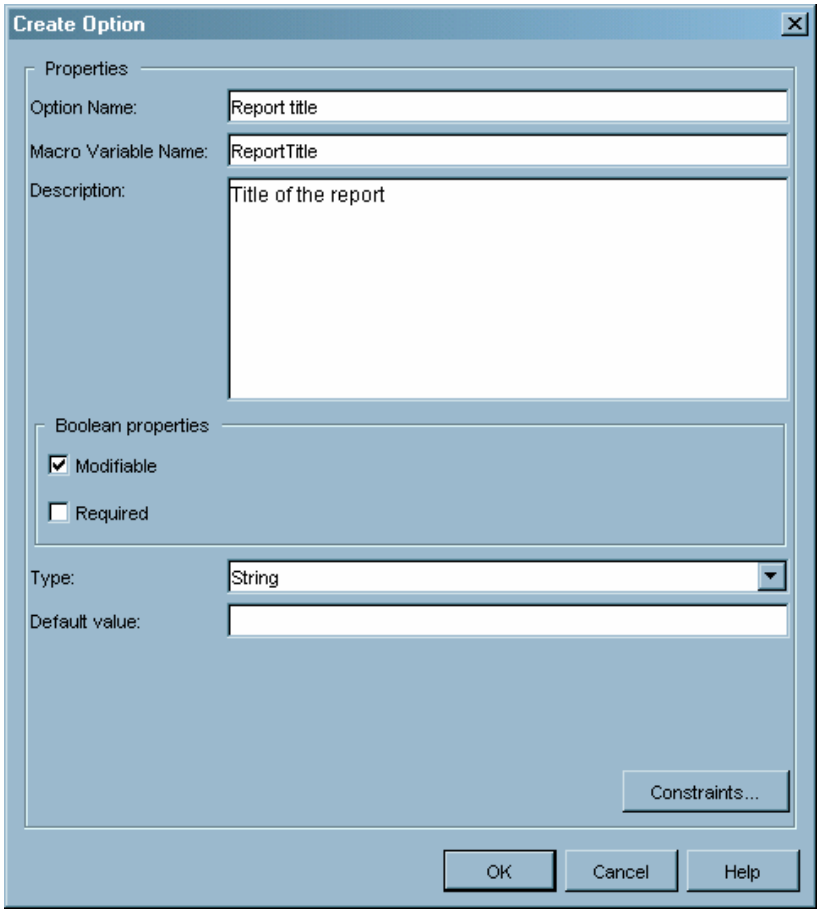
The **Columns to print** option specifies which columns are printed when the transformation is used in a job. Click **OK** to save the settings. A second row of the options list is displayed in the Options window. Click **New** to return to the Create Option window so that you can define the **Report title** option.

To define the **Report title** option, follow these steps:

1 Click **New**. A new row displays in the options table.
2 Enter *Report title* in the **Option Name** field.
3 Enter *ReportTitle* in the **Macro Variable Name** field.
4 Enter a description of the variable in the **Description** field.
5 Select the **Modifiable** option in the **Boolean properties** check box
6 Keep the default value of **String** in the **Type** field.

The following display shows the Create Option window for the **Report title** option.

Display 6.8 Create Option Window for the Report Title Option

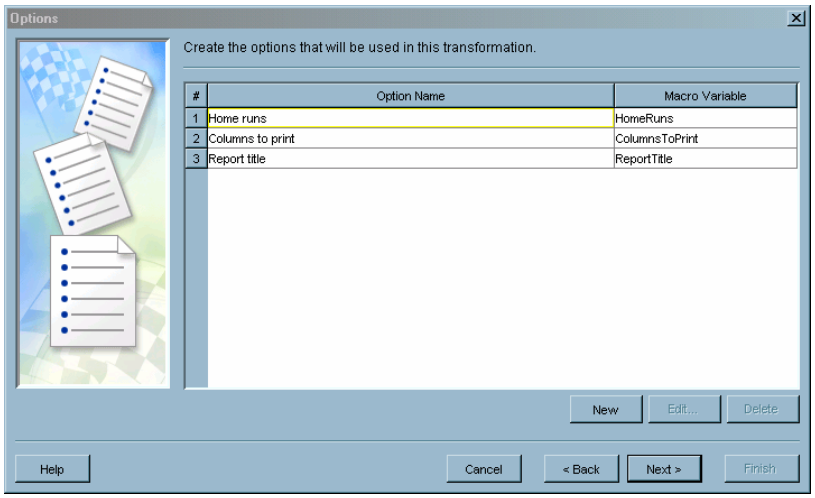


The **Report title** option enables users to enter a title on the **Options** tab in the transformation properties window when the transformation is used in a job. Click the **OK** button to save the settings. A third row of the options list is displayed in the Option window.

The **Report title** option creates the final user-defined variables in your transformation. Click **OK** to exit the Create Option window and return to the Options window. Click **Next** to access the Transformation Properties window.

The following display shows the completed Options window.

Display 6.9 Completed Options Window

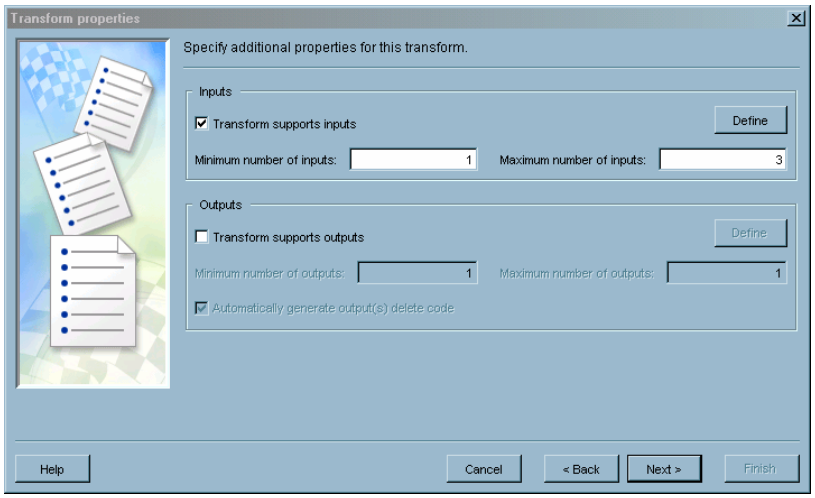


Click **`Next`** to access the Transform Properties window.

Specify the Remaining Options for the Transformation

Use the Transform Properties window to specify the remaining options for your generated transformation. The Transform Properties window for the transformation in this example resembles the following display.

Display 6.10 Transform Properties Window



To define the properties for the transformation, complete these steps:

1. Select the **`Transform supports inputs`** check box.
2. Enter *1* in the **`Minimum number of inputs`** field.
3. Enter *3* in the **`Maximum number of inputs`** field.

These values add a single input drop zone to the transformation when it is used in a job. The number of drop zones displayed in the Process Designer window is equal to the

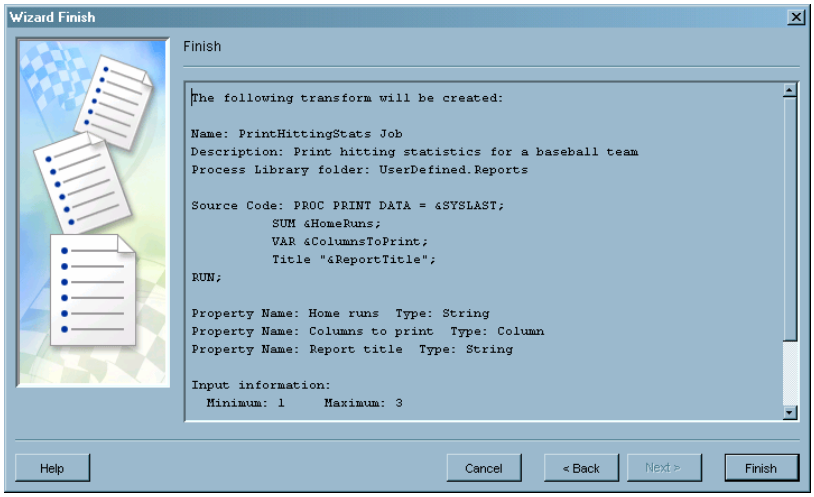
value in the **Minimum number of inputs** field. Therefore, only one drop zone is displayed. If you later update the transformation to increase this minimum number of inputs value, any jobs that have been submitted and saved will use the original value. The increased minimum number of inputs will be enforced only for subsequent jobs. This means that you can increase the minimum number of inputs without breaking existing jobs.

The increased maximum number of inputs is used to allow you to drop additional inputs into the drop zone. (In this example, you can have up to three inputs because you set the maximum to three.) The same rules apply to outputs.

The report generated by this transformation will be sent to the **Output** panel in the Process Designer window. Therefore, you do not need to add an output drop zone to the transformation by using the controls in the **Outputs** group box.

When you are finished defining properties for your transformation, click **Next** to access the Wizard Finish window. The Wizard Finish window for the transformation in this example resembles the following display.

Display 6.11 Wizard Finish Window

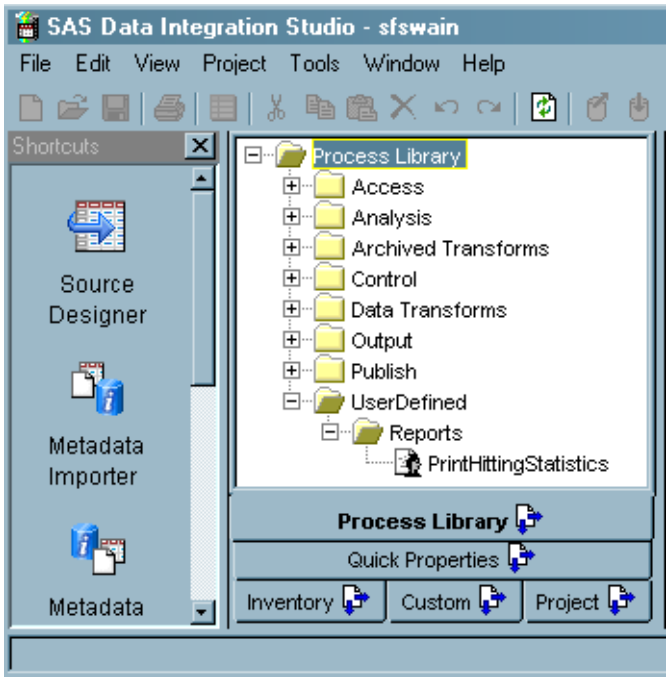


Use the Wizard Finish window to review the metadata that you have entered. When you are satisfied, click **Finish**. The transformation is created and saved in your metadata repository.

Save the Transformation

For this example, assume that the generated transformation was saved to the current metadata repository. The transformation is now visible in the Process Library tree, as specified in Display 3.17 on page 34. The following display illustrates the updated Process Library tree.

Display 6.12 Process Library Tree



The new template, PrintHittingStatistics, can now be used to create a job, as described in “Example: Using a Generated Transformation in a Job” on page 174.

Check In Your Transformation

If you are working under change management and you have added a new generated transformation, you must check it in order to make the transformation available to other SAS Data Integration Studio users who work with the same change-managed repository. Unlike other new objects that are added under change management, a new generated transformation does not appear in the Project tree. Nevertheless, you can still check in the transformation.

After you have created a new generated transformation, as described above, follow these steps:

1. From the SAS Data Integration Studio desktop, select the **`Project`** tab in the tree view.
2. In the Project tree, select the repository icon.
3. On the SAS Data Integration Studio desktop, select **Project ► Check In** from the menu bar. All objects in the project repository are checked into the change-managed repository.

The generated transformation will now be available to other SAS Data Integration Studio users who have access to the change-managed repository. The checked-in transformation will appear in the Process Library tree, in the Process Library folder.

Note: After a transformation is checked in, only an administrator can update it so that future changes do not adversely affect other users. △

Document Any Usage Details for the Transformation

The person who creates a generated transformation should document how it can be used to get the desired result. SAS Data Integration Studio users would need to know the following:

- Any requirements for inputs and outputs.

  For example, the columns in the source table for the PrintHittingStatistics template are assumed to be similar to the columns that are shown in Display 6.1 on page 76.

- Where the template sends its output: to a table, to the **Output** tab in the Process Designer window, or elsewhere.

- How to specify any values that are required by the template.

  For example, to produce the report that is shown in Display 6.2 on page 76, a title must be specified, a set of columns must be selected from the source, and the sum of the values in the HR column must be calculated.

Using a Generated Transformation in a Job

See "Example: Using a Generated Transformation in a Job" on page 174.

Importing and Exporting Generated Transformations

You can export a generated transformation to an XML file. The export feature enables you to create a generated transformation and make it available to SAS Data Integration Studio users who are using different metadata repositories.

The following sections explain how to export a transformation that is currently registered in your metadata repository, and how to import a transformation from an XML file and register it in the current metadata repository.

Exporting a Generated Transformation

Follow these steps to export a generated transformation that is registered in your metadata repository:

1. Select the generated transformation in the Process Library tree.
2. On the SAS Data Integration Studio desktop, select **Tools ▶ Transformation Export** from the menu bar. The Export Transform window displays.
3. Enter a filename for the XML file in the **File name** text box.
4. Click **OK** to export the transformation.

Importing a Generated Transformation

Follow these steps to import a generated transformation that has been saved in an XML file:

1. On the SAS Data Integration Studio desktop, select **Tools ▶ Transformation Import** from the menu bar. The Transformation Importer window displays. Build a list of XML files to import by following these steps one or more times.
2. Click **Add**. An Import Transform window appears.

3 Browse for and select an XML file that represents a transformation, and click **OK**.

4 Click **OK** in the Transform Importer window to import the transformations.

Additional Information About Generated Transformations

The Help for SAS Data Integration Studio provides additional information about generated transformations.

To display Help topics about installation and setup, follow these steps:

1 From the SAS Data Integration Studio menu bar, select **Help ▶ Contents**. The Help window displays.

2 In the left pane of the Help window, select **Prerequisites**.

To display Help topics about all administrative tasks, follow these steps:

1 From the SAS Data Integration Studio menu bar, select **Help ▶ Contents**. The Help window displays.

2 In the left pane of the Help window, select **Task Overviews ▶ SAS Data Integration Studio Task Reference ▶ User-Written Components and SAS Data Integration Studio ▶ Maintaining Generated Transformations**.

Additional Information About Administrative Tasks

The Help for SAS Data Integration Studio provides additional information about administrative tasks.

To display Help topics about installation and setup, follow these steps

1 From the SAS Data Integration Studio menu bar, select **Help ▶ Contents**. The Help window displays.

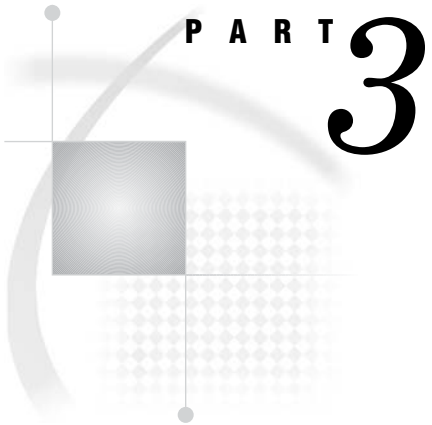
2 In the left pane of the Help window, select **Prerequisites**.

To display Help topics about all administrative tasks, follow these steps:

1 From the SAS Data Integration Studio menu bar, select **Help ▶ Contents**. The Help window displays.

2 In the left pane of the Help window, select **Task Overviews ▶ SAS Data Integration Studio Task Reference**.

3 See the section for administrative tasks.



PART 3

Creating Process Flows

CHAPTER

7

Main Tasks for Users

Preliminary Tasks for Users

Overview

After administrators complete the tasks that are described in Chapter 6, "Main Tasks for Administrators," on page 53, you must perform several tasks before you can begin work in SAS Data Integration Studio.

Starting SAS Data Integration Studio

Start SAS Data Integration Studio as you would any other SAS application on a given platform. For example, under Microsoft Windows, you can select **Start ▸ Programs ▸ SAS ▸ SAS Data Integration Studio**. You can also start the application from a command line. Navigate to the SAS Data Integration Studio installation directory and issue the

```
etlstudio.exe
```

command.

If you do not specify any options, SAS Data Integration Studio uses the parameters specified in the **etlstudio.ini** file. The following sections contain information on options you can specify on the command line or add to the **etlstudio.ini** file.

Specifying Java Options

To specify Java options when you start SAS Data Integration Studio, use the **--javaopts** option and enclose the Java options in single quotation marks. For example, the following command starts SAS Data Integration Studio on Windows and contains Java options that specify the locale as Japanese:

```
etlstudio ---javaopts '-Duser.language=ja -Duser.country=JP'
```

Specifying the Plug-in Location

By default, SAS Data Integration Studio looks for plug-ins in a **plugins** directory under the directory in which the application was installed. If you are starting SAS Data Integration Studio from another location, you must specify the location of the plug-in directory by using the

```
--pluginsDir
```

option. The syntax of the option is

```
etlstudio --pluginsdir <plugin path>
```

Specifying the Error Log Location

SAS Data Integration Studio writes error information to a file named **errorlog.txt** in the working directory. Because each SAS Data Integration Studio session overwrites this log, you might want to specify a different name or location for the log file. Use the following option to change the error logging location:

```
etlstudio --logfile '<filepath/filename>'
```

Specifying Message Logging

You can specify the server status messages that are encountered in a SAS Data Integration Studio session by using the

```
--MessageLevel level_value
```

option. Valid values for *level_value* include the following:

ALL	all messages are logged
CONFIG	static configuration messages are logged
FINE	basic tracing information is logged
FINER	more detailed tracing information is logged
FINEST	highly detailed tracing information is logged. Specify this option to debug problems with SAS server connections.
INFO	informational messages are logged
OFF	no messages are logged
SEVERE	messages indicating a severe failure are logged
WARNING	messages indicating a potential problem are logged

Creating a Metadata Profile (for Users)

A metadata profile is a client-side definition of where the metadata server is located. The definition includes a machine name, a port number, and one or more metadata repositories. In addition, the metadata profile can contain log on information and instructions for connecting to the metadata server automatically.

You must open a metadata profile before you can do any work in SAS Data Integration Studio. This section describes how a user could create a metadata profile for the example data warehouse.

Preparation

After an administrator has created one or more metadata repositories, the administrator provides the SAS Data Integration Studio user with the following information:

- the network name of the metadata server
- the port number used by the metadata server
- a login ID and password for that server
- the name of the repository that should be selected as the default metadata repository for this profile

Default Metadata Repository

When you create a metadata profile, you specify a default metadata repository for that profile. Typically, the administrator who creates metadata repositories simply tells SAS Data Integration Studio users which repository to select as the default. As a user, however, you might want to be aware of the effect that the default repository has on your work in SAS Data Integration Studio. The effect depends on whether you are working with change-managed metadata repositories.

If you are working with change-managed repositories, the default metadata repository must be a project repository that you own. You will use the project repository to check metadata out of and into the repository that is under change management. For the example data warehouse, the main metadata repository (Foundation) is under change-management control. Each user will use his own project repository to check metadata out of and into the foundation repository.

If you are not working with change-managed repositories, you can update objects in any metadata repository that is visible in the tree view on the SAS Data Integration Studio desktop, but you can add new objects to the default metadata repository only. If you try to add an object to a repository other than the default repository, the new object will be added to the default repository.

Task Summary

SAS Data Integration Studio users follow these steps to create a metadata profile:

1. Start SAS Data Integration Studio. The Open a Metadata Profile window displays.
2. Select **`Create a new metadata profile`**. The Metadata Profile wizard displays.
3. Click **`Next`**. In the general information window, enter a name for the profile. For the example data warehouse, the name could be **`etlUser1 Profile`**.
4. Click **`Next`**. In the Connection Information window, enter a machine address, port, user name, and password that will enable you to connect to the appropriate SAS Metadata Server.
5. Click **`Next`**. The wizard attempts to connect to the metadata server. If the connection is successful, the Select Repositories window displays.
6. In the Select Repositories window, select the appropriate repository as the default metadata repository for this profile. For the example data warehouse, the default repository for a user would be a project repository that would be used to check metadata out of and into the foundation repository.
7. Click **`Finish`** to exit the metadata profile wizard. You are returned to the Open a Metadata Profile window.

Opening a Metadata Profile

After a metadata profile has been created, you can open the profile in SAS Data Integration Studio. You must open a metadata profile in order to do any work in SAS Data Integration Studio.

Follow these steps to open a metadata profile:

1. Start SAS Data Integration Studio. The Open a Metadata Profile window displays.
2. Select **`Open an existing metadata profile`**. The selected profile is opened in SAS Data Integration Studio.

Another way to open a metadata profile is to start SAS Data Integration Studio, then select **File ▶ Open a Metadata Profile** from the menu bar.

If you are working with change-managed metadata repositories, see "Working with Change Management" on page 113. Assume that the main metadata repository for the example data warehouse is under change-management control.

If you are not working with change-managed metadata repositories, the following statements apply:

- You can update objects in any metadata repository for which you have write authority in the tree view on the SAS Data Integration Studio desktop.
- You can add only new objects to the default metadata repository.
- If you try to add an object to a repository other than the default repository, the new object is added to the default repository.

Selecting a Default SAS Application Server

One of the first tasks that most users will perform in SAS Data Integration Studio is to select a default SAS application server. A default SAS application server lets you access data, execute SAS code, and perform other tasks that require a SAS server but without having to specify a server each time. Typically, a metadata administrator defines this metadata object and then tells the SAS Data Integration Studio user which object to select as the default SAS application server.

For the example data warehouse, assume the metadata object for the default SAS application server is called SASMain. For details about SASMain, see "Default SAS Application Server" on page 57.

Follow these steps to select a default SAS application server:

1. From the SAS Data Integration Studio menu bar, select **File ▸ Options** to display the Options window.
2. Select the **SAS Server** tab.
3. On the **SAS Server** tab, select the desired server from the Server drop-down list. The name of the selected server appears in the **Server** field.
4. Click **Test Connection** to test the connection to the SAS Workspace Server(s) that are specified in the metadata for the server. If the connection is successful, go to the next step. If the connection is not successful, contact the metadata administrator who defined the server metadata for additional help.
5. After you have verified the connection to the default SAS application server, click **OK** to save any changes. The server that specified in the **Server** field is now the default SAS application server.

Main Tasks for Creating Process Flows

This section lists the main tasks for creating process flows in SAS Data Integration Studio. It is assumed that project leaders have identified the information that is required to answer a specific business question, and that they have identified a process flow that will load a target data store with the desired information, as described in Chapter 5, "Example Data Warehouse," on page 43. It is also assumed that installation and setup tasks have been completed as described in Chapter 6, "Main Tasks for Administrators," on page 53.

1. Start SAS Data Integration Studio. For details, see "Starting SAS Data Integration Studio" on page 93.
2. Create the appropriate metadata profile if one does not already exist. For details, see "Creating a Metadata Profile (for Users)" on page 94.
3. Open the appropriate metadata profile. For details, see "Opening a Metadata Profile" on page 95.

4 Add metadata for the job's inputs (data sources). For details, see "Registering Sources and Targets" on page 97.

5 Add metadata for the job's outputs (data targets). For details, see "Registering Sources and Targets" on page 97.

6 Create a new job and a process flow that will read the appropriate sources, perform the required transformations, and load the target data store with the desired information. See "Creating, Running, and Verifying Jobs" on page 99.

7 Run the job. See "Run the Job" on page 101.

Registering Sources and Targets

Overview

After you have completed the tasks that are described in "Preliminary Tasks for Users" on page 93, you are ready to specify metadata for sources and targets in SAS Data Integration Studio jobs.

A source is an input to an operation, and a target is an output of an operation. A data store can be a source, a target, or both, depending on its role in a process flow. Accordingly, there is no difference in the metadata for a source and a target. The methods in the following table can be used to enter metadata for both sources and targets in SAS Data Integration Studio jobs.

Table 7.1 Methods for Specifying Metadata for Data Stores

Data Store	Method for Specifying Metadata
A set of tables that are defined in a data model.	Import the data model in CWM format or in a format for which you have the appropriate Meta Integration Model Bridge. See "Importing and Exporting Metadata" on page 98.
One or more SAS tables that exist in physical storage.	SAS source designer. See "Example: Using a Source Designer to Register SAS Tables" on page 120.
One or more DBMS tables that exist in physical storage.	DBMS source designer. See the DBMS examples in the "Source Designer Examples" topic in the online help for SAS Data Integration Studio.
One or more Microsoft Excel (spreadsheet) tables that exist in physical storage.	Microsoft Excel source designer. See "Source Designer Example: Generate Metadata for a Microsoft Excel Table" in the Help for SAS Data Integration Studio.
One or more tables that exist in physical storage and that can be accessed with an Open Database Connectivity (ODBC) driver.	ODBC source designer. See "Source Designer Example: Generate Metadata for an ODBC Table" in the Help for SAS Data Integration Studio.

Data Store	Method for Specifying Metadata
One or more tables that exist in physical storage and that can be accessed with an OLE DB driver, such as a table that is stored in an OLE DB Oracle database.	OLE DB source designer. See "Source Designer Example: Generate Metadata for an OLE DB Table" in the Help for SAS Data Integration Studio.
A comma-delimited file or a similar external file that exists in physical storage.	External File source designer. See "Example: Using a Source Designer to Register an External File" on page 126.
A single table that does not exist in physical storage, such as a table that is created when a SAS Data Integration Studio job is executed for the first time.	Target Table Designer. See "Example: Using the Target Table Designer to Register SAS Tables" on page 140.
Generate metadata for a table when a specific source designer for that kind of table is not available. An example might be one or more tables that are defined in an XML file.	Generic source designer. See "Source Designer Example: Generate Metadata for an XML Table" in the Help for SAS Data Integration Studio.
Add and maintain a cube.	Cube Designer. See "Working with OLAP Cubes" on page 116.

Registering DBMS Tables with Keys

Tables in a database management system often have primary keys, unique keys, and foreign keys.

A primary key is one or more columns that are used to uniquely identify a row in a table. A table can have only one primary key. The column(s) in a primary key cannot contain null values.

A unique key is also one or more columns that can be used to uniquely identify a row in a table. A table can have one or more unique keys. Unlike a primary key, a unique key can contain null values.

A foreign key is one or more columns that are associated with a primary key or unique key in another table. A table might have one or more foreign keys. A foreign key is dependent upon its associated primary or unique key. In other words, a foreign key cannot exist without a primary or unique key.

Note: When specifying metadata for a DBMS table with foreign keys, if you want to preserve the foreign key, you must specify metadata for all of the tables that are referenced by the foreign keys. △

For example, suppose that Table 1 had foreign keys that referenced primary keys in Table 2 and Table 3. To preserve the foreign keys in Table 1, you could use the Metadata Importer wizard or a source designer wizard to import metadata for Tables 1, 2, and 3.

Importing and Exporting Metadata

Introduction

SAS Data Integration Studio is a SAS Open Metadata Architecture application. It can share metadata repositories with other SAS Open Metadata Architecture applications,

such as SAS Management Console, SAS Enterprise Miner, SAS Information Delivery Portal, SAS OLAP Administrator, and the metadata LIBNAME engine.

SAS Data Integration Studio also enables you to do the following tasks:

- import table metadata from applications that do not support the SAS Open Metadata Architecture
  - the metadata must be in Common Warehouse Metamodel (CWM) format or in a format that is supported by the optional Meta Integration Model Bridges (MIMBs) from Meta Integration Technology, Inc.
  - you can perform change analysis on the imported metadata
- export the default metadata repository
- export and import SAS Data Integration Studio jobs
- export and import cubes

Importing Metadata with Change Analysis

Suppose that you wanted to import a data model for a set of new tables, but you are not certain that you want to register all tables in the default metadata repository. Accordingly, you will choose the **`Compare import metadata to repository`** option in the Metadata Import wizard so that you can view the new tables before you register them.

The **`Compare import metadata to repository`** option specifies that metadata in the selected file will be imported and compared to existing metadata. Differences in tables, columns, indexes, and keys are detected. Under change-management, imported metadata is compared to checked-in metadata that is associated with the library or DBMS server that you selected in the wizard. Without change management, imported metadata is compared to the metadata in the default repository that is associated with the selected library or DBMS server. Differences will be stored in a comparison result library. You can view the changes in the Differences window.

Additional Information

The Help for SAS Data Integration Studio provides additional information about exporting and importing metadata. To display the relevant Help topics, do the following:

1. From the SAS Data Integration Studio menu bar, select **Help ▸ Contents**. The Help window displays.
2. In the left pane of the Help window, select **Task Overviews ▸ SAS Data Integration Studio Task Reference ▸ Understanding Metadata Import and Export**.

Working With Jobs

Creating, Running, and Verifying Jobs

Overview

A job is a metadata object that specifies processes that create output. SAS Data Integration Studio uses each job to generate or retrieve SAS code that reads sources

and creates targets in physical storage. After you have entered metadata for sources and targets, you are ready to create jobs.

Prerequisites

It is easier to create a job if metadata for the tables in the job are created first. For details about these tasks, see Chapter 8, "Registering Data Sources," on page 119 and Chapter 9, "Registering Data Targets," on page 139.

Check Out Existing Metadata That Must Be Updated

As you work on the current job, will you update the metadata for any registered tables? If so, you must check out the metadata for that table.

For example, suppose that you want to create a job in which data is extracted from a registered table and then is written to a report. In this case, the metadata for the table is not changed, so you would not have to check out the metadata. However, suppose that you want to create a job that require you to add three new columns to the metadata for a registered table. In that case, the metadata for the table is changed, so you would need to check out the metadata.

The next task is to create and populate the job.

Create and Populate the New Job

Follow these steps to create and populate a new job. To populate a job, you will create a complete process flow diagram, from sources, through transformations, to targets.

1. On the SAS Data Integration Studio desktop, select **Tools ▶ Process Designer** from the menu bar. The New Job wizard displays. (You can use this wizard to create an empty job, into which you can drag and drop tables and transformations. That is the approach that is described here.)
2. Enter a name for the job and click **`Finish`**.
3. An empty job will open in the Process Designer window.
4. Add metadata for sources, targets, and transformations as needed. The goal is to create a complete process flow diagram, from sources, through transformations, to targets. Drag and drop transformation templates from the Process Library tree. Drag and drop tables from the Inventory tree or from another tree in the tree view. If you try to drop an object in a zone where it is invalid, an error message will be written to the Status bar at the bottom of the SAS Data Integration Studio desktop.

As you add sources, targets, and transformations to a process flow diagram, SAS Data Integration Studio automatically maps source columns to target columns. Depending on the nature of the job, you might or might not need to update the automatic column mappings or the other default metadata in a job.

The next task is to view or update the job, as needed.

Update the Job as Needed

The following steps describe a general approach for updating the default metadata in a job. The specific updates will vary according to the sources, targets, and transformations in a job and the purpose of the job. The examples in Chapter 10, "Example Process Flows," on page 149 describe scenarios in which a few, specific updates are needed to the automatic column mappings and the other default metadata in a job.

1. In the Process Designer window, select the first source in the flow, then select **File ▶ Properties** from the menu bar. A properties window displays.

2 Click the **Columns** tab to confirm that the needed columns are present. Add, delete, or replace columns as necessary. Repeat these steps for each source and target in the job, as needed. For details about updating column metadata, click the **Help** button on the **Columns** tab.

3 In the Process Designer window, select the first transformation in the flow, then select **File ▶ Properties** from the menu bar. A properties window displays.

4 Update the transformation as necessary to achieve the purpose of the job. Be sure to display the **Mapping** tab for the transformation to be sure that data flows correctly through the transformation. As needed, repeat these steps for each transformation in the job, working in a source-to-target direction. For details about updating mapping metadata, click the **Help** button on the **Mapping** tab.

When all metadata in the job is correct, the next task is to run the job.

Run the Job

After the metadata for a job is complete, you must submit the job for execution in order to create targets on the file system. With the job displayed in the Process Designer window, select **Process ▶ Submit** from the menu bar. SAS Data Integration Studio generates code for the job and submits the code to a SAS application server. The server executes the code. A pop-up window is displayed to indicate that the job is running.

The next task is to check the log.

Check the Log

If a job executes and a pop-up error message appears, or if you simply want to look at the log for the completed job, follow these steps:

1 Click the **Log** tab in the Process Designer window.

2 In the **Log** tab, scroll through the SAS log information that was generated during the execution of the job. (The code that was executed for the job is available in the **Source Code** tab of the Process Designer window. The source code is continuously updated as you make changes to the job, and it is checked and updated as necessary when you submit the job.)

3 If you find errors in the source code for a step, select the corresponding transformation in the process flow diagram, then select **File ▶ Properties** from the menu bar. A properties window displays.

4 Correct the metadata and resubmit the job until there are no more errors.

5 After the job runs without error, save the job. Select **File ▶ Save** from the menu bar.

For more information about evaluating logs, see "Using SAS Logs to Analyze Process Flows" on page 189.

The next task is to verify that the job created the correct output.

Verify Job Outputs

After the job runs without error and has been saved, you should confirm that the targets contain the data you need, in the format that best communicates the purpose of the targets.

1 To view the data for a target in the job process flow diagram, select the desired target, then select **View ▶ View Data** from the menu bar. The data in the target is displayed. Confirm that the correct data is displayed and that the data is correctly formatted for the purpose of the target.

For more information about viewing data, see “Viewing Data in Tables, External Files, or Temporary Output Tables” on page 110. If a target needs to be improved, change the properties of that target or the transformations that feed data to that target. If the outputs are correct, and you are working in a change-managed repository, you can check in the job.

Check In the Metadata

Follow these steps to check in all metadata objects in the Project tree:

1. In the Project tree, select the repository icon.
2. On the SAS Data Integration Studio desktop, select **Project ▶ Check In Repository** from the menu bar. All of the objects in the Project repository are checked into the change-managed repository.

For more information abour change management, see “Working with Change Management” on page 113.

Customizing or Replacing Code Generated for Jobs

See Appendix 2, “Customizing or Replacing Generated Code in SAS Data Integration Studio,” on page 223.

Deploying a Job for Scheduling

Administrators can deploy a job for scheduling so that the job can run in batch mode at a specified date and time. For more information, see “Deploying a Job for Scheduling” on page 65.

Enabling Parallel Execution of Process Flows

When SAS Data Integration Studio generates code for a job, it can add macros that enable parts of the job to be executed in parallel. You can enable these macros by doing the following:

- selecting YES in the Enable parallel processing macros option on the **Options** tab in the properties window for a job
- including a Loop transformation in a job

When you enable the parallel processing option for a job, macros are generated at the top of the job code with comments to enable you to create your own transformations or code to take advantage of parallel processing. When you include a Loop transformation in a job, the transformation generates the necessary macros to take advantage of sequential execution, symmetric multiprocessing (SMP) execution, or execution on a grid computing network.

No special software or metadata is required to enable parallel processing on SMP servers. If grid options have been enabled for a job, but the grid software has not been configured and licensed, SAS Data Integration Studio does not generate grid-enabled code for the job. It generates code that is appropriate for symmetric multiprocessing (SMP) on the SAS application server.

The Help for SAS Data Integration Studio provides additional information about parallel processing. To display the relevant Help topics, do the following:

1. From the SAS Data Integration Studio menu bar, select **Help ▶ Contents**. The Help window displays.

2 In the left pane of the Help window, select **Task Overviews ▶ SAS Data Integration Studio Task Reference ▶ Maintaining Jobs ▶ Parallel Processing Options**.

Generating a Stored Process for a Job

Administrators can generate one or more stored processes for a job that is selected in the Inventory tree or the Custom tree on the SAS Data Integration Studio desktop. For more information, see "Converting Jobs into Stored Processes" on page 69.

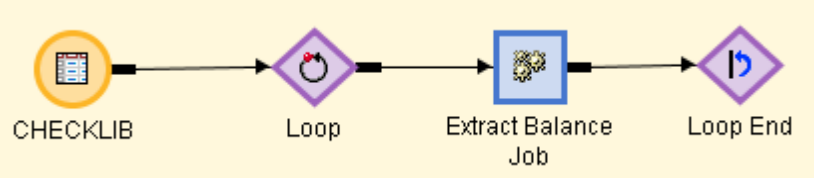
Improving the Performance of Jobs

See Chapter 11, "Optimizing Process Flows," on page 181.

Maintaining Iterative Jobs

An iterative job is a job with a control loop in which one or more processes are executed multiple times. For example, the following display shows the process flow for an iterative job.

Display 7.1 Process Flow for an Iterative Job



The process flow specifies that the inner Extract Balance job will be executed multiple times, as specified by the loop transformations and the CHECKLIB control table. The inner job is also called a parameterized job because it specifies its inputs and outputs as parameters.

The Help for SAS Data Integration Studio provides additional information about iterative jobs. To display the relevant Help topics, do the following:

1 From the SAS Data Integration Studio menu bar, select **Help ▶ Contents**. The Help window displays.

2 In the left pane of the Help window, select **Task Overviews ▶ SAS Data Integration Studio Task Reference ▶ Maintaining Jobs ▶ Maintaining Iterative Jobs**.

Monitoring the Status of Jobs

About Job Status Monitoring

When you execute a job in SAS Data Integration Studio, a return code for each transformation in the job is captured in a macro variable. The return code for the job is set according to the least successful transformation in the job. SAS Data Integration Studio enables you to associate a return code condition, such as Successful, with an

action, such as Send Email or Send Event. In this way, users can specify how a return code is handled for the job or transformation.

For example, you can associate a return code with an action that performs these tasks:

- aborts the job or transformation
- calls a user-defined SAS macro
- sends a status message to a person, a file, or an event broker that then passes the status code to another application

You can also use status codes to capture job statistics, such as the number of records before and after the append of the last table loaded in the job. To capture statistics about a job, you would associate a return code with the Send Job Status action.

The **`Status Handling`** tab, which is included in the property windows for jobs and for some transformations, is used to associate a return code condition with an action. However, the property windows for most transformations do not have a **`Status Handling`** tab. To return the status of a transformation that does not have a **`Status Handling`** tab, you can use a Return Code Check transformation to insert status-handling logic at a desired point in the process flow for a job. The Return Code Check transformation can be inserted between existing transformations and removed later without affecting the mappings in the original process flow.

Note: The Lookup transformation has its own way of handling exceptions. For details, see the Help for the **`Errors`** tab in the property window for the Lookup transformation. △

The Help for SAS Data Integration Studio provides additional information about monitoring job status. To display the relevant Help topics, do the following:

1. From the SAS Data Integration Studio menu bar, select **Help ▶ Contents**. The Help window displays.
2. In the left pane of the Help window, select **Task Overviews ▶ SAS Data Integration Studio Task Reference ▶ Maintaining Jobs ▶ Maintaining Status Code Handling**.

Displaying Job Status

After you have submitted one or more jobs for execution, display the Job Status Manager window to view the name, status, starting time, ending time, and application server that will be used for all jobs that are submitted in the current session. From the SAS Data Integration Studio desktop, select **Tools ▶ Job Status Manager** to display the Job Status Manager window.

Using the New Job Wizard

See “New Job Wizard” on page 32.

Working With SAS Data Quality Software

SAS Data Integration Studio has a number of features that can help you to improve the quality of your data. Except for the Data Validation transformation, these features require your site to license SAS Data Quality Server software and to complete some configuration tasks. For more information about setup, administrators should see “Supporting SAS Data Quality” on page 72.

Create Match Code and Apply Lookup Standardization Transformations

The Process Library tree includes two transformation templates that require SAS Data Quality Server software: Create Match Code and Apply Lookup Standardization. These transformations enable you to increase the value of your data through data analysis and data cleansing.

To use these transformations, the SAS Data Quality Server software must be installed, a SAS application server must be configured to access a Quality Knowledge Base, and the Quality Knowledge Base must contain the locales that you need to reference in your SAS Data Integration Studio jobs. When the prerequisites have been met, you can drag and drop these transformations into your process flow diagrams.

SAS Data Quality Functions in the Expression Builder Window

SAS Data Integration Studio provides an Expression Builder window in the properties window of some transformations. (For a description of this window, see "Expression Builder Window" on page 16.) If SAS Data Quality Server software is available to you, the Expression Builder window includes a wide range of data quality functions. One way to see the data quality functions is to open the properties window of the SQL Join transformation and select the **Where** tab. For detailed information about the data quality functions, see the *SAS Data Quality Server: Reference*, which is available in the online SAS Help and Documentation for Base SAS and in SAS OnlineDoc.

Data Validation Transformation

When incorporated into SAS Data Integration Studio jobs, the Data Validation transformation enables you to detect error conditions and specify responses to those errors. Error conditions include blank or missing values, duplicate values, and invalid values. The actions that you can take in response to erroneous values include stopping the job, changing the value, or writing the row to an error table instead of to the target. The Data Validation transformation does not require SAS Data Quality Server software.

Updating Metadata

Updating Metadata for Jobs

Overview

Use the property window for a job to update its basic metadata. For example, you can specify code that should be run before or after the job. For a description of the job properties window, see "Job Properties Window" on page 17.

Update Metadata for a Job

Assume that the metadata for the job is currently checked into a change-managed repository.

1. On the SAS Data Integration Studio desktop, select the **Inventory** tab.
2. In the Inventory tree, open the **Jobs** folder.
3. Select the desired job, then select **Project ▶ Check Out**. The metadata for the job is checked out. A check mark is displayed next to the job in the Inventory tree. An icon indicating a checked-out job appears in the Project tree.
4. In the Project tree, select the metadata for the job, then select **File ▶ Properties** from the menu bar. The properties window for the job displays.
5. Use the tabs in this window to update the metadata for the job. Each tab has its own Help button.
6. When you are finished updating the metadata, save the changes to the job by clicking the **OK** button.
7. In the Project tree, select the repository icon. From the menu bar on the SAS Data Integration Studio desktop, select **Project ▶ Check In Repository**.

Updating Metadata for Tables or External Files

Overview

Use the table properties window to update the metadata for a table or external file that is registered in a current metadata repository. For a description of the table properties window, see "Table or External File Properties Window" on page 26.

Update Metadata for a Table or External File in a Tree View

The following steps describe one way to update the metadata for a table or external file in a tree view.

1. On the SAS Data Integration Studio desktop, select the **Inventory** tab.
2. In the Inventory tree, open the **Tables** folder or the **External Files** folder.
3. Select the table, then select **Project ▶ Check Out**. The metadata for the table or file is checked out. A check mark is displayed next to the table or file in the Inventory tree. An icon indicating a checked-out table or file appears in the Project tree.
4. In the Project tree, select the table or file, and select **File ▶ Properties** from the menu bar. The properties window for the table or file is displayed.

   Note: You must display the table or file from the Project tree in order to update metadata. Displaying the table or file from the Inventory tree enables browsing only.
5. Use the tabs in this window to make changes to the metadata for the table or file. Each tab has its own **Help** button.
6. When you are finished updating the metadata, save the changes to the table or file by clicking the **OK** button.
7. In the Project tree, select the repository icon. From the menu bar on the SAS Data Integration Studio desktop, select **Project ▶ Check In Repository**.

Update Metadata for a Table or External File in a Process Flow

The following steps describe one way to update the metadata for a table or external file in the process flow for a job. When working under change management, you must check out both the job and the table or external file.

1. On the SAS Data Integration Studio desktop, select the **Inventory** tab.
2. In the Inventory tree, expand the **Jobs** folder and the **Tables** folder or the **External Files** folder.
3. Select the job and the table or external file, then select **Project ▶ Check Out**. The metadata for the job and the table or file is checked out. A check mark is displayed next to these objects in the Inventory tree. Icons indicating a checked-out job and a checked-out table or file appear in the Project tree.
4. In the Project tree, select the desired job, then select **View ▶ View Job** from the menu bar. The process flow diagram for the job displays on the **Process Editor** tab in the Process Designer window.
5. Select the metadata for the desired table or external file, then select **File ▶ Properties** from the menu bar. The properties window displays the metadata for the table or file.

   Note: You must display the table or file from the Project tree in order to update metadata. Displaying the table or file from the Inventory tree enables browsing only.
6. Use the tabs in the properties window to make changes to the metadata for the table or file. Each tab has its own **Help** button.
7. When you are finished updating the metadata, save the changes to the table or file by clicking the **OK** button.
8. Close the Process Designer window and save your changes to the job.
9. In the Project tree, select the repository icon. From the menu bar on the SAS Data Integration Studio desktop, select **Project ▶ Check In Repository**.

Using a Physical Table to Update Table Metadata

The Update Table Metadata feature compares the columns in a physical table to the columns that are defined in the metadata for that table. If column metadata does not match the columns in the physical table, the metadata is updated to match the physical table.

For existing tables, the Update Table Metadata feature adds new columns, removes deleted columns, and records changes to all column attributes. When you select and run this feature against one or more tables simultaneously, the update log lists which tables have been successfully updated and which have failed.

When you use the Update Table Metadata option on a physical table in DBMS format and the DBMS table has more than one schema, the Update Table Metadata option selects the first schema. The Update Table Metadata feature uses the following resources: the current metadata server and the SAS application server to read the physical table the current metadata server to update the metadata to match the physical table.

You are prompted to provide a user name and password for the metadata server if this was not already saved with the current metadata profile. If you have not provided a user name and password for the SAS application server during the current session, you are prompted to provide them. A warning message displays if the SAS Workspace Server component of the SAS application server is older than SAS 9.1.3, Service Pack 3.

Follow these steps to use the Update Table Metadata feature:

1. If you are working under change management, check out one or more tables that contain the metadata that you want to update.

2 From the Project tree, select one or more tables for which you want to update the metadata and then select **Update Table Metadata** from the Tools menu.

You might be prompted to supply a user name and password for the relevant servers When the update is finished, you can choose to view the resulting SAS log.

Updating Metadata for Transformations

Overview

A transformation is a metadata object that specifies how to extract data, transform data, or load data into data stores. A number of standard transformations are provided in the Process Library for SAS Data Integration Studio. For a description of the standard transformations, see Appendix 1, "Standard Transformations in the Process Library," on page 217.

There are two main reasons to update the metadata for a transformation:

- to update the default metadata for a transformation when it is first added to a job
- to change the behavior of a transformation in a process flow

Use the property window for a transformation to update its metadata. For a description of the transformation properties window, see "Transformation Properties Window" on page 27.

Update the Default Metadata for a Transformation

When you drag a transformation template from the Process Library and drop it into the process flow for a job, the default metadata for the transformation is mostly blank. You must update the default metadata according to your goals for that transformation in the process flow.

The general steps for updating the default metadata for a transformation are described in "Update the Job as Needed" on page 100.

Examples of how to update the default metadata for transformation are provided in Chapter 10, "Example Process Flows," on page 149. To see more examples of how to update the default metadata for transformations, follow these steps to display the relevant Help topics:

1 From the SAS Data Integration Studio menu bar, select **Help ▶ Contents**. The online Help window displays.

2 In the left pane of the Help window, select **Examples ▶ Process Library Examples**.

Other Updates to the Metadata for a Transformation

After you have made the initial updates to the default metadata for a transformation, you might have to update the transformation again later. For example, you might need to change the options on the **Options** tab for the transformation. The following steps describe one way to update the metadata for a transformation. Assume that the metadata for the job that contains the transformation is currently checked into a change-managed repository.

1 On the SAS Data Integration Studio desktop, select the **Inventory** tab.

2 In the Inventory tree, expand the **Jobs** folder.

3 Select the job with the transformation that you want to update, then select **Project ▶ Check Out**. The metadata for the job is checked out. A check mark is

displayed next to the job in the Inventory tree. An icon indicating a checked-out job appears in the Project tree.

4 In the Project tree, select the desired job, then select **View ▶ View Job** from the menu bar. The process flow diagram for the job displays on the **Process Editor** tab in the Process Designer window.

5 Select the metadata for the transformation that you want to update, then select **File ▶ Properties** from the menu bar. The properties window displays the metadata for the transformation.

   Note: You must display the transformation from the Project tree in order to update metadata. Displaying the transformation from the Inventory tree enables browsing only.

6 Use the tabs in the properties window to make changes to the metadata for the transformation. Each tab has its own **Help** button.

7 When you are finished updating the metadata, save the changes to the table or file by clicking the **OK** button.

8 Close the Process Designer window and save your changes to the job.

9 In the Project tree, select the repository icon. From the menu bar on the SAS Data Integration Studio desktop, select **Project ▶ Check In Repository**.

Setting Name Options for Individual Tables

Overview

SAS Data Integration Studio cannot access tables or columns with case-sensitive names or with special characters in the names unless the appropriate options have been specified in the metadata for the table.

Prerequisites

For tables in DBMS format, it is assumed that the appropriate name options have already been set for the database library that is used to access the table, as described in “Supporting Case and Special Characters in Table and Column Names” on page 73. Name options do not need to be set on the library that is used to access a table in SAS format.

Task Summary

The following steps describe one way to enable name options for a table whose metadata has been saved to a metadata repository. It is assumed that the metadata repository is under change management.

1 On the SAS Data Integration Studio desktop, select the **Inventory** tab.

2 In the Inventory tree, open the **Tables** folder or the **External Files** folder.

3 Select the table that you want to update, then select **Project ▶ Check Out**. The metadata for the table is checked out. A check mark is displayed next to the table in the Inventory tree. An icon indicating a checked-out table appears in the Project tree.

4 In the Project tree, select the metadata for the table, then select **File ▶ Properties** from the menu bar. The properties window for the table is displayed.

5 In the properties window, click the **Physical Storage** tab.

6 On the **Physical Storage** tab, select **Enable case-sensitive DBMS object names** to support case-sensitive table and column names. Select **Enable special characters within DBMS object names** to support special characters in table and column names. For details about these options as they relate to SAS tables, see "Case and Special Characters in SAS Table and Column Names" on page 73.

7 Click **OK** to save your changes.

8 In the Project tree, select the repository icon. From the menu bar on the SAS Data Integration Studio desktop, select **Project ▶ Check In Repository**.

Viewing Data in Tables, External Files, or Temporary Output Tables

Overview

Use the View Data window to display the data for a table or external file that is registered in a current metadata repository. You can also use the View Data window to display the data for a transformation's temporary output table. For a description of the View Data window, see "View Data Window" on page 27. When working under change management, you do not have to check out a table or file to view its data.

View Data for a Table or External File in a Tree View

The following steps describe one way to display the data for a table or external file from the tree view. The table or file must be registered in a current metadata repository and must exist in physical storage.

1 On the SAS Data Integration Studio desktop, select the **Inventory** tab.

2 In the Inventory tree, open the **Tables** folder or the **External Files** folder.

3 Select the metadata for the desired table or file, then select **View ▶ View Data** from the menu bar. The View Data window displays the data in the table or file. Verify that the content, format, and column headings are what you expected.

View Data for a Table or External File in a Process Flow

After the metadata for a table or external file has been added to the process flow in a job, you might want to verify that the corresponding physical table or file contains the data that you were expecting.

Note: The metadata for a target table might not point to a physical table until after the job has run for the first time. Before the first run, new target tables might exist as metadata only. △

The following steps describe one way to view the data for a table or external file in the process flow for a job:

1 On the SAS Data Integration Studio desktop, select the **Inventory** tab.

2 In the Inventory tree, expand the **Jobs** folder.

3 Select the desired job, then select **View ▶ View Job** from the menu bar. The process flow diagram for the job displays on the **Process Editor** tab in the Process Designer window.

4 Select the metadata for the desired table or external file, then select **View ▶ View Data** from the menu bar. The View Data window displays the data in the table or file. Verify that the content, format, and column headings are what you expected.

View Data in a Transformation's Temporary Output Table

If a target table in a process flow does not contain the data that you were expecting, you might want to verify that the inputs to that target table are correct. The most immediate input to the target table is the temporary output table of the preceding transformation in the process flow. Use the View Data window to display data in the temporary output tables for a transformation in a job, if the transformation creates temporary output tables.

Note: To view the temporary output tables for a transformation, you have to execute the job and view the temporary tables in the same session of the Process Designer window. The temporary output tables for transformations are deleted when you close the Process Designer window. △

The following steps describe one way to display data in the temporary output tables for a transformation in a job:

1 On the SAS Data Integration Studio desktop, select the **`Inventory`** tab.

2 In the Inventory tree, expand the **`Jobs`** folder.

3 Select the desired job, then select **View ▶ View Job** from the menu bar. The process flow diagram for the job displays on the **`Process Editor`** tab in the Process Designer window.

4 Execute the job. (The transformation must have been executed at least once in order for the temporary output tables to exist in physical storage.) Select **Tools ▶ Submit** from the menu bar.

5 Select the metadata for the desired transformation, then select **View ▶ View Data** from the menu bar. The View Data window displays the data in the temporary output table for a transformation. Verify that the content, format, and column headings are what you expected.

See also "Analyzing Transformation Output Tables" on page 192.

Viewing Metadata

Viewing Metadata for Jobs

Overview

Use the property window for a job to view its basic metadata. For example, you can find out if user-written code has been specified for the entire job, or if any code is supposed to run before or after the job. For a description of the job properties window, see "Job Properties Window" on page 17.

View Metadata for a Job

Assume that the metadata for the job is currently checked into a change-managed repository.

1. On the SAS Data Integration Studio desktop, select the **Inventory** tab.
2. In the Inventory tree, expand the **Jobs** folder.
3. Select the desired job, then select **File ▶ Properties** from the menu bar. A properties window for the job is displayed.
4. Use the tabs in this window to view the metadata for the jobs. Each tab has its own **Help** button.

Viewing Metadata for Tables and External Files

Overview

Use the properties window to display the metadata for a table or external file that is registered in a current metadata repository. For a description of the table properties window, see "Table or External File Properties Window" on page 26. When working under change management, you do not have to check out a table or file to view its metadata.

View Metadata for a Table or External File in a Tree View

The following steps describe one way to view the metadata for a table or external file in the tree view:

1. On the SAS Data Integration Studio desktop, select the **Inventory** tab.
2. In the Inventory tree, open the **Tables** folder or the **External Files** folder.
3. Select the metadata for the desired table or file, then select **File ▶ Properties** from the menu bar. The properties window displays the metadata for the table or file. Each tab has its own **Help** button.

View the Metadata for a Table in a Process Flow

The following steps describe one way to view the metadata for a table or external file in the process flow for a job:

1. On the SAS Data Integration Studio desktop, select the **Inventory** tab.
2. In the Inventory tree, expand the **Jobs** folder.
3. Select the desired job, then select **View ▶ View Job** from the menu bar. The process flow diagram for the job displays on the **Process Editor** tab in the Process Designer window.
4. Select the metadata for the desired table or external file, then select **File ▶ Properties** from the menu bar. The properties window displays the metadata for the table or file. Each tab has its own **Help** button.

Viewing Metadata for Transformations

Overview

Use the property window for a transformation to view the metadata for a process in a job. For example, you can find out what options have been set on the **Options** tab or if user-written code has been specified for the transformation. For a description of the transformation properties window, see "Transformation Properties Window" on page 27. For a description of the standard transformations in the Process Library, see Appendix 1, "Standard Transformations in the Process Library," on page 217.

View Metadata for a Transformation

Assume that the metadata for the job that contains the transformation is currently checked into a change-managed repository.

1. On the SAS Data Integration Studio desktop, select the **Inventory** tab.
2. In the Inventory tree, expand the **Jobs** folder.
3. Select the job that contains the transformation, then select **View ▶ View Job** from the menu bar. The process flow diagram for the job displays on the **Process Editor** tab in the Process Designer window.
4. Select the metadata for the desired transformation, then select **File ▶ Properties** from the menu bar. The properties window displays the metadata for the transformation.

Working with Change Management

About Change Management

SAS Data Integration Studio enables you to create metadata objects that define sources, targets, and the transformations that connect them. This metadata is saved to one or more repositories.

When administrators create the metadata repositories for a project, they usually put them under change-management control. Change management enables multiple SAS Data Integration Studio users to work with the same metadata repository at the same time without overwriting each other's changes.

Under change management, SAS Data Integration Studio users without administrative privilege cannot directly update any metadata in a change-managed repository. Instead, each user must check out metadata from the change-managed repository and into a project repository for that user. The user can update the metadata as needed, and when finished, they can check it back into the change-managed repository. As long as the metadata for a resource is checked out by one person, it is locked so that it cannot be updated by another person until the metadata has been checked back into the repository.

Adding New Metadata

Preparation

Before you can add metadata to a repository that is under change-management control, you must open an appropriate metadata profile, as described in "Opening a Metadata Profile" on page 95.

Note: When you add a new metadata object, such as the metadata for a table, the metadata goes directly into the Project tree on the SAS Data Integration Studio desktop. △

Task Summary

In SAS Data Integration Studio, you can add metadata for a library, a table, a job, or some other resource. The metadata for the new resource will be added to the Project tree on the SAS Data Integration Studio desktop. For details about adding metadata for libraries, tables and jobs, see the following references:

- "Registering Libraries" on page 59
- "Registering Sources and Targets" on page 97
- Chapter 10, "Example Process Flows," on page 149

Next Tasks

After you have added new metadata to the Project tree, you can update it. After you have finished any updates, you can check in the metadata to the change-managed repository. See "Checking In Metadata" on page 115.

Checking Out Existing Metadata

Preparation

Before you can check out metadata from a change-managed repository, you must open an appropriate metadata profile, as described in "Opening a Metadata Profile" on page 95.

Remember the following about check-out operations:

- You cannot update metadata in the Inventory tree or the Custom tree. You must check out the metadata to the Project tree and update it there.
- After the metadata for a resource has been checked out by one person, it is locked so that it cannot be updated by another person until the metadata has been checked back in.
- If two or more tables share a common metadata object—such as the metadata for a primary key, a note, or a document—and you check out one of these tables, only you can check out the other tables that share that common object. (Other users cannot access the common metadata object that you have checked out, and the shared object is required in order to check out a table that uses that object.)

Task Summary

1. On the SAS Data Integration Studio desktop, click the **`Inventory`** tab or the **`Custom`** tab. The appropriate tree displays.
2. Open the folder for the kind of metadata that you want to check out, such as the **`Tables`** folder for tables in the Inventory tree.
3. Right-click the metadata that you want to check out and select **Change-Management ▶ Check Out**. You can also left-click the metadata that you want to check out, then go the drop-down menu and select **Project ▶ Check Out**. The metadata is checked out and displays in the Project tree.

Next Tasks

After you have checked out metadata to the Project tree, you can update it. After you have finished any updates, you can check in the metadata to the change-managed repository.

Checking In Metadata

Preparation

When you are finished working with all of the metadata that is displayed in the Project tree, use the check-in feature to store the objects in the change-managed repository.

Note: A check-in operation checks in all metadata objects that are in the Project tree. You cannot check in selected objects and leave other objects in the Project tree. △

Accordingly, you might find it convenient to work with small sets of related objects in the Project tree.

Task Summary

1. On the SAS Data Integration Studio desktop, click the **`Project`** tab. The Project tree displays.
2. Right-click the project repository icon and select **`Check In Repository`**. You can also left-click the project repository icon, open the drop-down menu, and select **Project ▶ Check In Repository**. The Check In window displays.
3. Enter meaningful comments in the **`Name`** field (and perhaps in the **`Description`** field) about the changes that were made to all of the objects that you are about to check in. The text entered here becomes part of the check in/check out history for all objects that you are checking in. If you do not enter meaningful comments, the check in/check out history is less useful.
4. When finished entering comments in the Check In window, click **`OK`**. All metadata objects that are in the project repository are checked into the change-managed repository.

Additional Information About Change Management

The Help for SAS Data Integration Studio provides more details about change-management. To display the relevant Help topics, do the following:

1. From the SAS Data Integration Studio menu bar, select **Help ▶ Contents**. The Help window displays.
2. In the left pane of the Help window, select **Task Overviews ▶ SAS Data Integration Studio Task Reference ▶ Using Change Management in SAS Data Integration Studio**.

Working with Impact Analysis and Reverse Impact Analysis (Data Lineage)

Impact analysis displays information about how data is used. Reverse impact analysis displays information about how data was developed. Analytical results are derived from the current metadata repository and any parent repositories.

Impact analysis shows you the jobs, tables, and cubes that make use of a selected table or column. This information is helpful before you modify or delete data. For example, if you perform impact analysis on a column, the Impact Analysis window might show that the selected column is used to build an OLAP cube. If you deleted that column, you might also have to change the job that builds the cube. You can also track the usage of generated transformations using impact analysis. In this case, the Impact Analysis window shows all of the jobs that make use of the generated transformation.

Reverse impact analysis shows you the lineage of the data in a selected table, column, or cube. This information is useful when you need to trace data errors or data sources. For example, if you perform reverse impact analysis on a table, the results might show that the data in the table was validated by a job that contains a Data Validation transformation. The source for validation job might be an Oracle table. Data errors might be present in the lookup table that provides valid values, or in the original Oracle data.

Working with OLAP Cubes

Overview of OLAP Cubes

Online analytical processing (OLAP) cubes are logical sets of data that are structured in a hierarchical, multidimensional arrangement. Cubes are valuable analytical tools because they provide easily modified views of large data sets. Because of their size, cubes are built and stored on servers and viewed, or queried, from client cube viewers. To decrease the response time for commonly submitted queries, numeric data summaries are calculated at build time and stored with the cube data.

OLAP Capabilities in SAS Data Integration Studio

In SAS Data Integration Studio, you can create and update OLAP cubes with the Cube Designer, which is available in the Target Designer wizard. The Cube Designer

walks you through the process of specifying an OLAP schema, source data, and any other cube definitions such as calculated measures, drill-through tables, and aggregations.

Another method of creating and updating cubes is to write SAS programs in the Source Editor. The cubes are defined in OLAP procedures. When the programs are ready to run, you can submit them for execution from the Source Editor or you can include them in stored processes and run them at a later date.

In both the Cube Designer and in the OLAP procedure (PROC OLAP), you can choose to define cube metadata only and create the physical cube at a later date.

Cubes can be defined as targets in jobs, though the data that is written to those cubes is not available to cube viewers until the cube is updated.

Cubes that appear in the inventory of SAS Data Integration Studio are included in impact analyses that involve cube data or cube source data.

Prerequisites for Cubes

Follow these general steps to begin building and querying cubes with the Cube Designer:

1. Install a SAS OLAP Server.
2. Add metadata for the SAS OLAP Server. (You can specify the SAS OLAP Server as one component of the default SAS application server for SAS Data Integration Studio.)
3. Define an OLAP schema and assign the SAS OLAP Server to the schema.
4. Define cube source tables using the Source Designer.
5. Start the SAS OLAP Server.

For details about these tasks, see the *SAS OLAP Server: Administrator's Guide* and the *SAS Intelligence Platform: Administration Guide*.

Additional Information About Cubes

Extensive information on building and maintain cubes is available in the Help for SAS Data Integration Studio. To display a list of links to all cube-related Help topics, including examples, open the Help browser and search for "Maintaining Cubes".

The documentation for the SAS OLAP Server software provides complete coverage of SAS OLAP Cube Studio, the OLAP procedure, and the SAS OLAP Server Monitor in SAS Management Console. See the *SAS OLAP Server: User's Guide* and the *SAS OLAP Server: Administrator's Guide*.

For information about the configuration of SAS OLAP Servers on SAS Workspace Servers, see the *SAS Intelligence Platform: Administration Guide*.

Additional Information About User Tasks

The Help for SAS Data Integration Studio provides additional information about user tasks. To display Help topics about the main user tasks, follow these steps:

1. From the SAS Data Integration Studio menu bar, select **Help ▸ Contents**. The Help window displays.
2. In the left pane of the Help window, select **Task Overviews ▸ SAS Data Integration Studio Task Reference**.
3. See the section for user tasks.

CHAPTER

8

Registering Data Sources

Sources: Inputs to SAS Data Integration Studio Jobs

In general, a source is an input to an operation. In a SAS Data Integration Studio job, a source is a data store from which information will be extracted, transformed, and loaded into a target, which can be a data store in a data warehouse, a data mart, or another data collection.

After you complete the tasks that are described in "Preliminary Tasks for Users" on page 93, you can register the data sources that will be used in a job. To register a data source, you will enter metadata about it and save the metadata to a SAS Metadata Repository.

Your project plan should identify the data sources that are required for a particular job. For example, the sources that are required to answer specific business questions in the Orion Star Sports & Outdoors project are listed under each business question. See the Identifying Sources section under each business question in Chapter 5, "Example Data Warehouse," on page 43. Use the examples in this chapter, together with general methods that are described in "Registering Sources and Targets" on page 97, to register the sources that will be used in a SAS Data Integration Studio job.

Example: Using a Source Designer to Register SAS Tables

Preparation

Suppose that you wanted to create a report that extracted information from three operational tables: CUSTOMER, ORDERS, and ORDER_ITEM. Your first task would be to register these tables so that they can be included in a job that creates the report. Because these tables exist in physical storage, you could use a source designer wizard to register these tables.

Assume that the following preparations have been made:

- All of the tables are in SAS format and are stored in a SAS library named Ordetail.
- The Ordetail library has been registered in a current metadata repository. For details registering libraries, see "Registering Libraries" on page 59.
- You have selected a default SAS application server for SAS Data Integration Studio, as described in "Selecting a Default SAS Application Server" on page 96. This server can access the tables that you want to register.
- The main metadata repository is under change-management control. In the current example, after you register the tables, you must check in their metadata. For details about change management, see "Working with Change Management" on page 113.

Start SAS Data Integration Studio and Open the Appropriate Metadata Profile

Follow these steps to begin work in SAS Data Integration Studio:

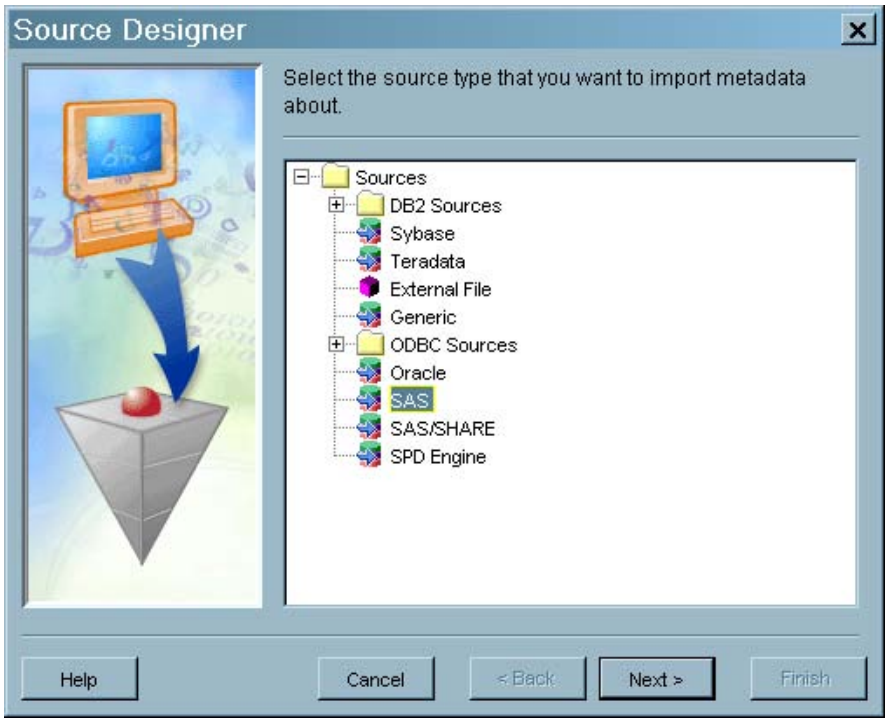
1. Start SAS Data Integration Studio as described in "Starting SAS Data Integration Studio" on page 93.
2. Open the appropriate metadata profile as described in "Opening a Metadata Profile" on page 95.

You do not need to check out a library in order to add metadata about tables in that library. Accordingly, the next task is to select the appropriate source designer.

Select the SAS Source Designer

From the SAS Data Integration Studio menu bar, select **Tools ▶ Source Designer**. The Source Designer selection window displays, as shown in the following display.

Display 8.1 Source Designer Selection Window



The list of available wizards on your machine might be somewhat different from the list shown in the previous display. Only those data formats for which source designer wizards have been installed are available. From this window, take the following actions:

1 Click the **SAS** icon.

2 Click **Next**.

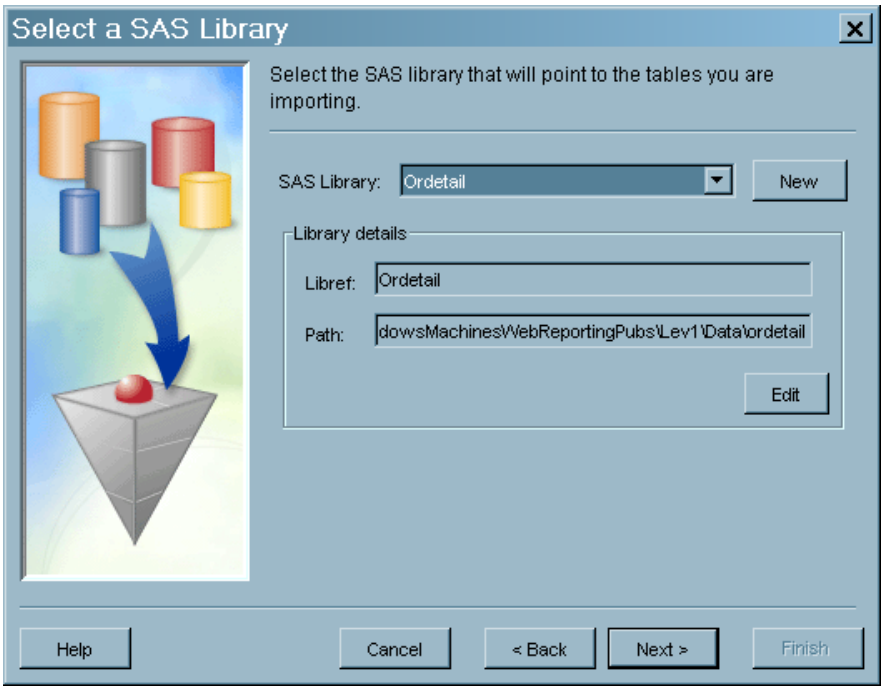
The wizard tries to open a connection to the default SAS application server. If there is a valid connection to this server, you might be prompted for a user name and a password. After you have provide that information, you will be taken directly to the Select a SAS Library window.

The next task is to select the library that contains the tables that you want to register.

Select the Library That Contains the Tables

After you have connected to a SAS application server, use the Select a SAS Library window to specify the SAS library that contains the tables that you want to register. For the current example, select the Ordetail library, as shown in the following display.

Display 8.2 Select a SAS Library Window



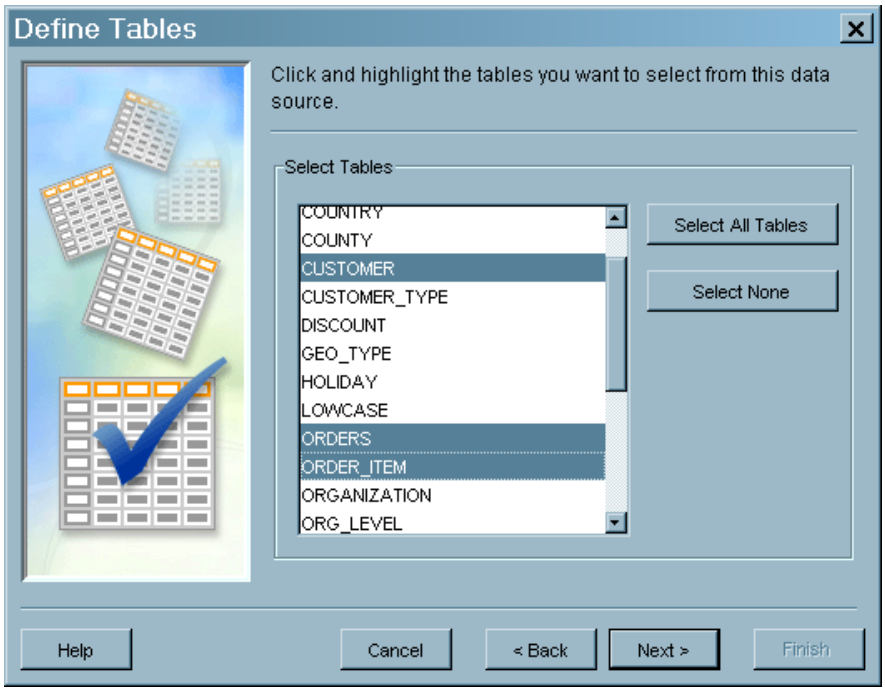
After selecting the appropriate library, click **`Next`**. The SAS application server accesses the library, and the Define Tables window displays.

The next task is to select the tables that you want to register.

Select the Tables

The following display shows the tables that are stored in the Ordetail library.

Display 8.3 Define Tables Window



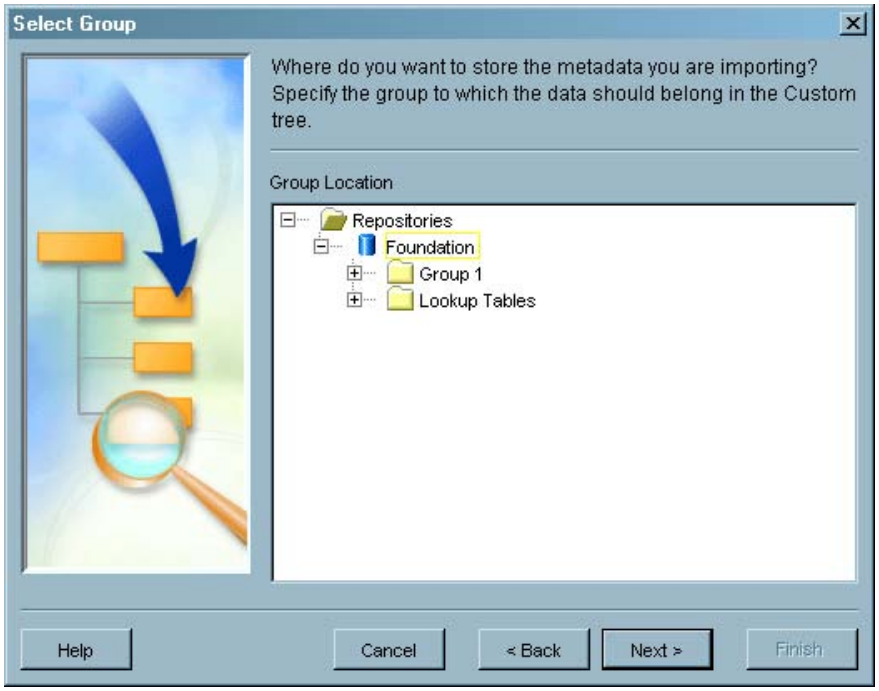
In this example, we want to create metadata objects for CUSTOMER, ORDERS, and ORDER_ITEM. Accordingly, select these tables and click **`Next`**.

The next task is to specify a Custom tree group for the table metadata that you are creating.

Specify a Custom Tree Group

Use the Select Group window to specify a Custom tree group for the table metadata that you are creating. A Custom tree group is a folder that you can use to keep similar kinds of metadata together in the Custom tree on the SAS Data Integration Studio desktop.

Display 8.4 Select Group Window

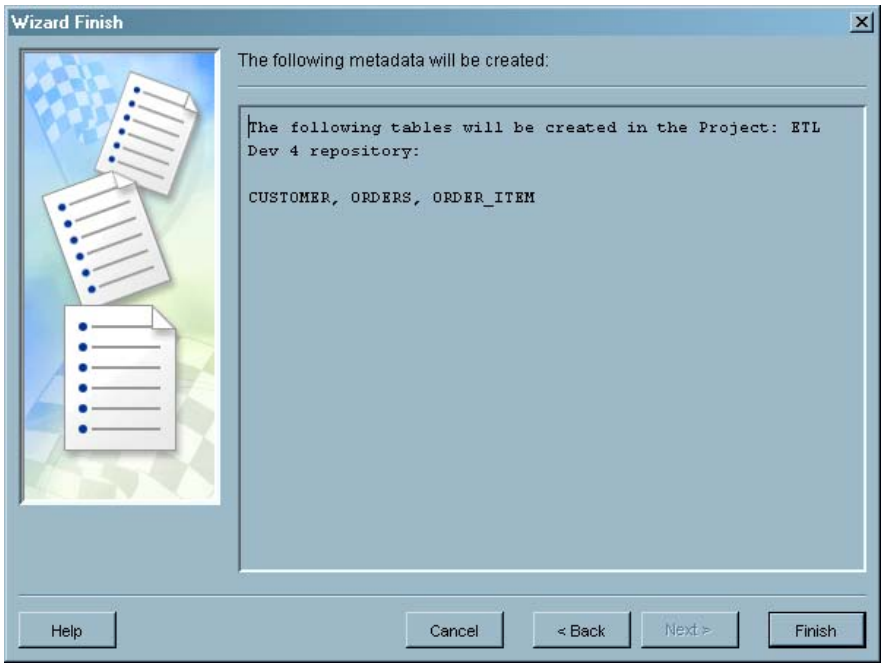


For this example, assume that you do not want to specify a user-defined group for the table. Select **`Foundation`** and click **`Next`**. The Wizard Finish window displays.

Save the Metadata for the Tables

After you specify a group for storing the source tables, use the Wizard Finish window to review the metadata that you entered.

Display 8.5 Wizard Finish Window



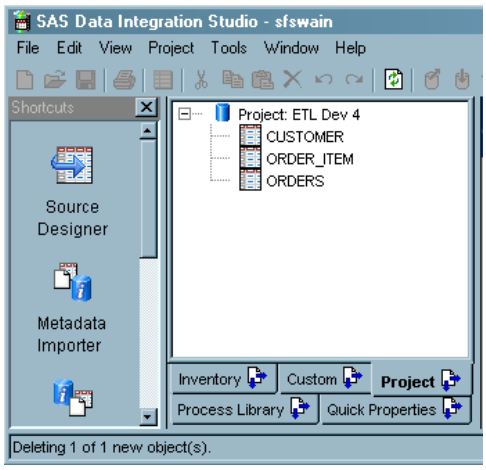
Review the text in the metadata pane on the Wizard Finish window. Shown previously is the metadata for the CUSTOMER, ORDERS, and ORDER_ITEM source tables. If the metadata is correct, click **`Finish`**. The metadata for the tables is written to the current metadata repository.

At this point, you could add the metadata that you just created to a job, or you could check in the metadata.

Check In the Metadata

Under change management, new metadata objects are added to the Project tree on the SAS Data Integration Studio desktop, as shown in the following display.

Display 8.6 Project Tree with Metadata Objects for Three Tables



You must check in the new table metadata in order to save it to the change-managed repository.

1. In the Project tree, select the repository icon (*Project: etlDev4*).
2. From the SAS Data Integration Studio menu bar, select **Project ▶ Check In Repository**.

All metadata objects in the project repository will be checked into the change-managed repository. The new objects will be visible in the Inventory tree.

Example: Using a Source Designer to Register an External File

Preparation

An external file is a file that is created and maintained by a host operating system or by another vendor's software application. SAS can read data from and route output to external files.

Suppose that you wanted to create a report that requires information from a comma-delimited external file. Your first task would be to register this file so that it an be included in a job that creates the report. Because the file is in comma-delimited format, you would use the Delimited External File source designer wizard to register the file.

Assume that the following preparations have been made:

- The comma-delimited external file is called employeeFlatFile.csv . It contains information about dependents of Orion Star Sports employees.
- You have selected a default SAS application server for SAS Data Integration Studio, as described in "Selecting a Default SAS Application Server" on page 96. This server can access the employeeFlatFile.csv file.

- The main metadata repository is under change-management control. In the current example, after you register the file, you must check in its metadata. For details about change management, see "Working with Change Management" on page 113.

About External File Source Designers

External file source designers enable you to do the following tasks:

- Register delimited, fixed-width, and user-defined external files. The supported file types are TXT, DAT, and CSV.
- Process variable-length records and fixed-length records.
- Process character, numeric and nonstandard numeric data (such as currency data or signed numbers).
- Specify how missing values should be treated.
- Process data in which one record is spanned over multiple lines, as well as data in which multiple records are included in a single data line.
- Remove columns, arrange the order of the columns, change attributes of any column, and add new columns in the metadata.

The external file source designers use a sample of data from the external file, together with metadata that you enter, to estimate the length and data type of the columns. You can specify the rows used in sampling of data by specifying the start record and how many records should be included in the sample. To find additional information about all of the external file source designers, see the relevant topics in the SAS Data Integration Studio Help. From the SAS Data Integration Studio desktop, follow these steps to display the relevant topics:

1. From the menu bar, select **Help ▶ Contents** to open the "Introduction to SAS Data Integration Studio" topic.
2. Select **Examples ▶ Source Designer Examples** to open the "Source Designer Examples" topic.
3. Select the topic for the type of external file that you need to process.

   Note: For information about using external files in SAS Data Integration Studio, follow the reference to the "Managing External Files in SAS Data Integration Studio" topic found at the end of the "Overview" section in each of the external file examples. △

Start SAS Data Integration Studio and Open the Appropriate Metadata Profile

Follow these steps to begin work in SAS Data Integration Studio:

1. Start SAS Data Integration Studio as described in "Starting SAS Data Integration Studio" on page 93.
2. Open the appropriate metadata profile as described in "Opening a Metadata Profile" on page 95.

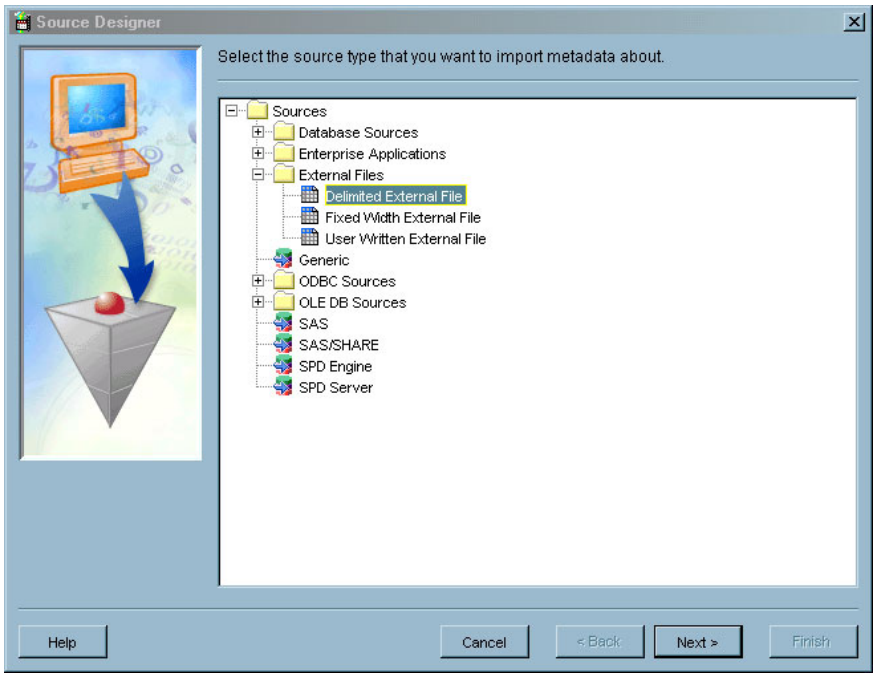
The next task is to select an external file source designer.

Select an External File Source Designer

To select an external file source designer, from the SAS Data Integration Studio menu bar, select **Tools ▶ Source Designer**.

The Source Designer selection window displays, as shown in the following display.

Display 8.7 Source Designer Selection Window



From this window, take the following actions:

1. Open **`External Files`**.
2. Click **`Delimited External File`**.
3. Click **`Next`**.

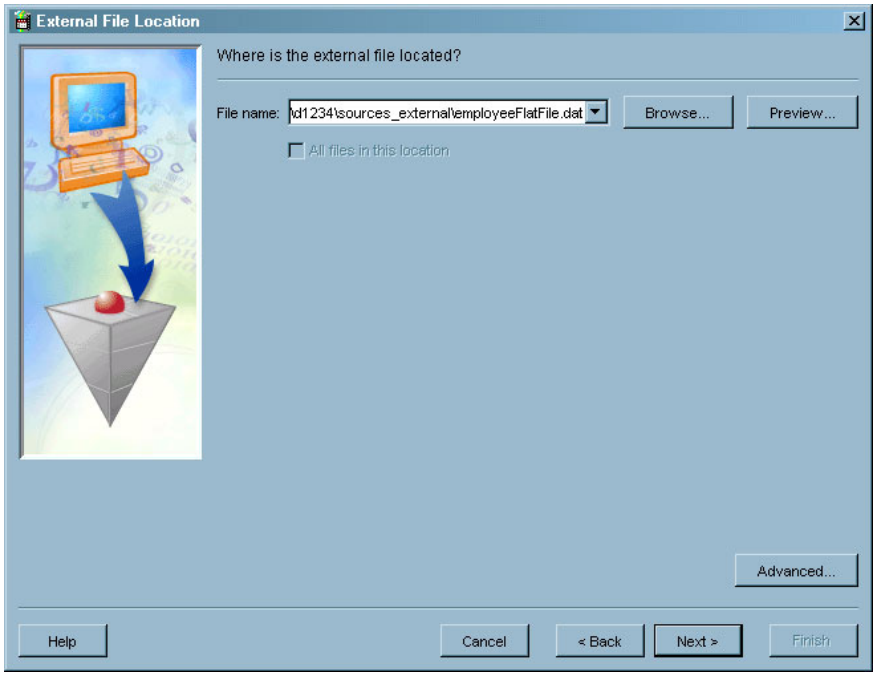
The wizard attempts to open a connection to the default SAS application server. If there is a valid connection to this server, you might be prompted for a user name and a password. After you provide that information, the External File Location window displays.

Specify Location of the External File

Follow these steps to specify the location of the external file:

1 Specify a physical path to the external file in the **`File name`** field, as follows: *\\d1234\sources_external\employeeFlatFile.csv*. The following display shows the completed External File Location window with the path specified to the external file.

Display 8.8 External File Location Window



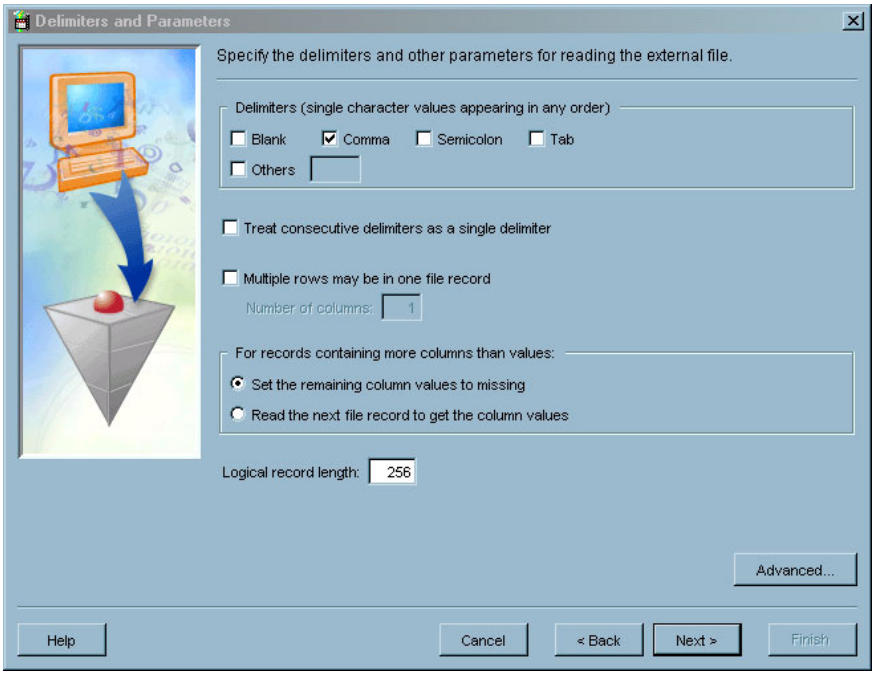
2 When you have set the correct path to the physical file, click **`Next`**. The Delimiters and Parameters window displays.

Set Delimiters and Parameters

Follow these steps to set delimiters and parameters for the external file:

1 Deselect the **Blank** check box in the **Delimiters** group box. Then, select the **Comma** check box, as shown in the following display. This step is necessary because the **Blank** check box is selected by default. However, the employeeFlatFile.csv file is comma-delimited. This change ensures that the wizard parameters match the incoming data.

Display 8.9 Delimiters and Parameters Window



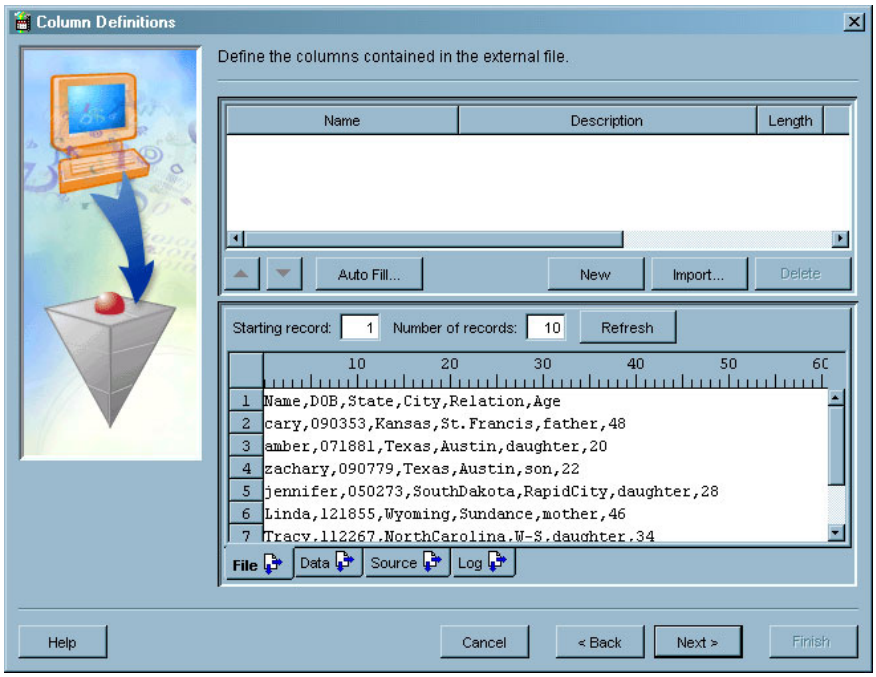
2 Accept the remaining defaults and click **Next**. The Column Definitions window displays.

Define the Columns for the External File Metadata

Follow these steps to create column definitions for the external file:

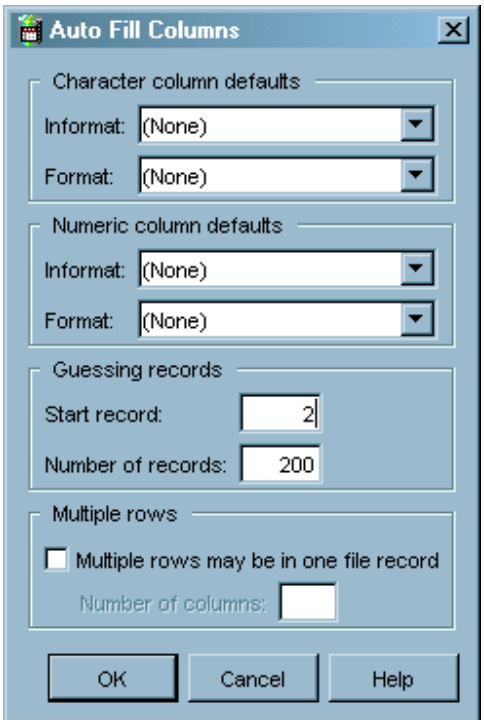
1 Click **`Refresh`** to view the data from employeeFlatFile.csv on the **`File`** tab in the view pane at the bottom of the window, as shown in the following display.

Display 8.10 Delimited Data in the Column Definitions Window



2 The next step populates preliminary data into the columns component of the Columns Definitions window. To begin, click **`Auto Fill`**. The Auto Fill Columns window displays, as shown in the following display.

Display 8.11 Auto Fill Columns Window

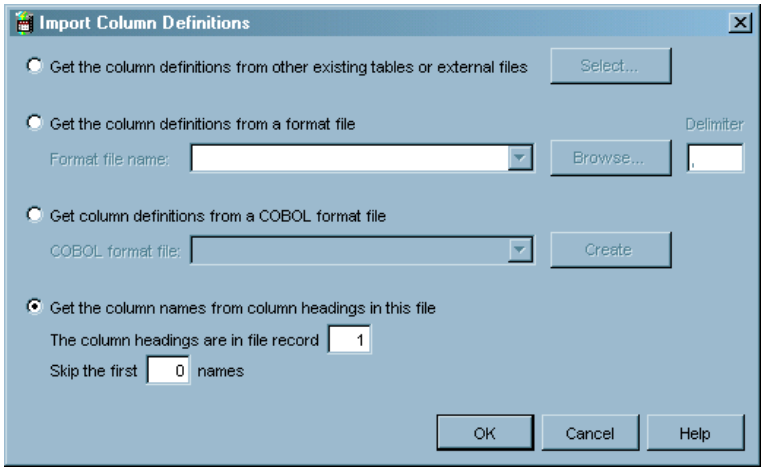


3 Change the value that was entered in the **`Start record`** field in the **`Guessing records`** group box to *2*. This setting ensures that the guessing algorithm begins with the second data record in the external file. The first data record in the file is unique because it holds the column names for the file. Thus, all data that it contains uses the character data type. The other rows in the file begin with a character-based name, but the remaining values are numeric. Therefore, excluding the first data from the guessing process yields more accurate preliminary data.

4 Accept all of the remaining default settings. Then, click **`OK`** to return to the Column Definitions window.

5 Use the Import function to simplify the task of entering column names. After you use the Auto Fill function to fill in the preliminary data, click **`Import`**. The Import Column Definitions window displays, as shown below.

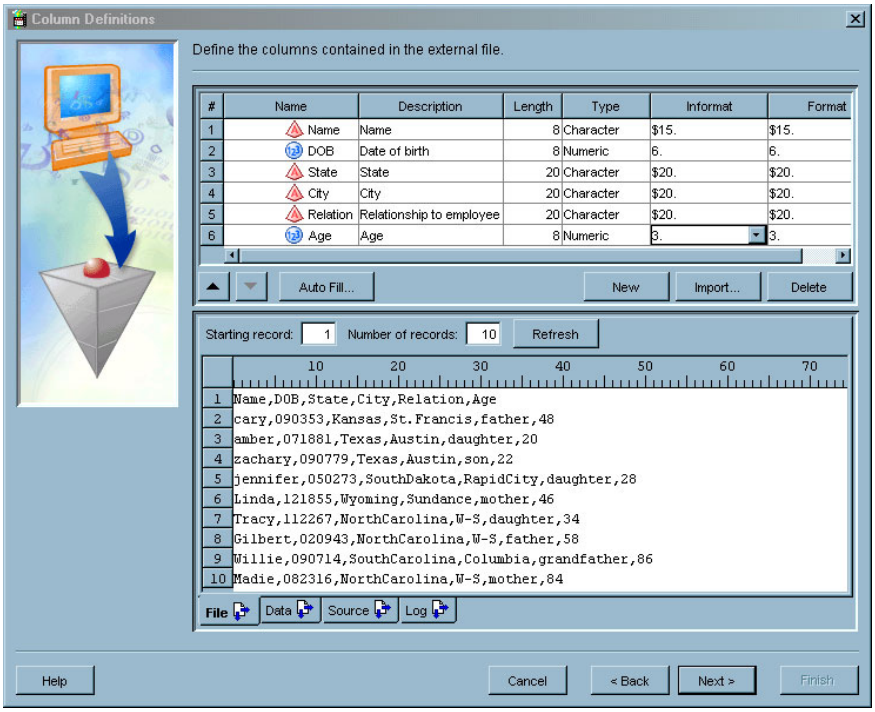
Display 8.12 Import Column Definitions Window



6 Select the **`Get the column names from column headings in this file`** radio button and keep the default settings for the fields underneath it. Click **`OK`** to save the settings and return to the Column Definitions window. The names from the first record in the external file are populated in the **`Name`** column. You now can edit them as needed. The preliminary metadata is shown in the following display.

Note: If you use the get column names from column headings function, the value in the **`Starting record`** field on the **`Data`** tab in the view pane in the Column Definitions window is automatically changed. The new value is 1 greater than the value in the **`The column headings are in file record`** field in the Import Column Definitions window. △

Display 8.13 Preliminary Column Metadata



The columns component at the top of the window is populated with preliminary metadata.

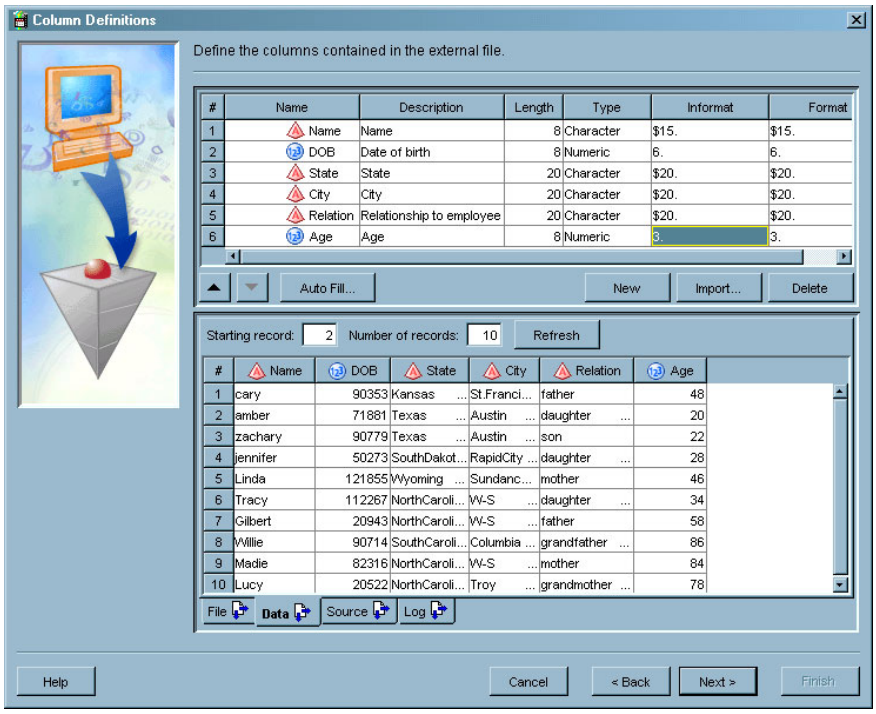
7 The preliminary metadata that is populated into the columns component typically includes column names and descriptions that are too generic to be useful for SAS Data Integration Studio jobs. Fortunately, you can modify the columns component by clicking in the cells that you need to change and entering the correct data. Enter the values that are in the following table into the component:

Name	Description	Length	Type	Informat	Format
Name	Name	8	Char.	$15.	$15.
DOB	Date of birth	8	Num.	6.	6.
State	State	20	Char.	$20.	$20.
City	City	20	Char.	$20.	$20.
Relation	Relation ship to employee	20	Char.	$20.	$20.
Age	Age	8	Num.	3.	3.

You need to change only a few values. First, use the values that are shown in the first record of the data that is shown in the view pane at the bottom of the Column Definitions window as a guide when you fill in the values for the **`Name`** column in the column component section of the screen. These values served as the column names in the employeeFlatFile.csv external file. Also, the **`Informat`** and **`Format`** columns for the numeric values are based on the values included in the

sample data that is processed by the guessing function. The results are accurate for this particular set of records, but you should still examine them to make sure that they are representative for employee dependent account records in general. The following display shows the completed Column Definitions window.

Display 8.14 Corrected Column Metadata

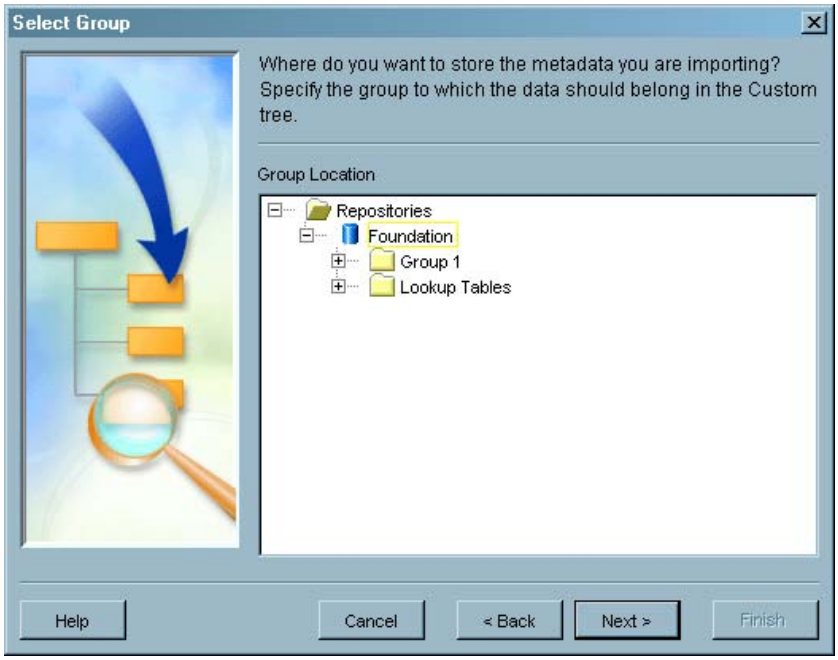


8 To verify that the metadata you have entered is appropriate for the data in the external file, click the **Data** tab and then click **Refresh**. If the metadata matches the data, the data displays properly on the **Data** tab. If the data does not display properly, update the column metadata and click **Refresh** to verify that the appropriate updates have been made.

To view the code that will be generated for the external file, click the **Source** tab. To view the SAS log for the generated code, click the **Log** tab. The code that displays on the **Source** tab is the code that is generated for the current external file when it is included in a SAS Data Integration Studio job.

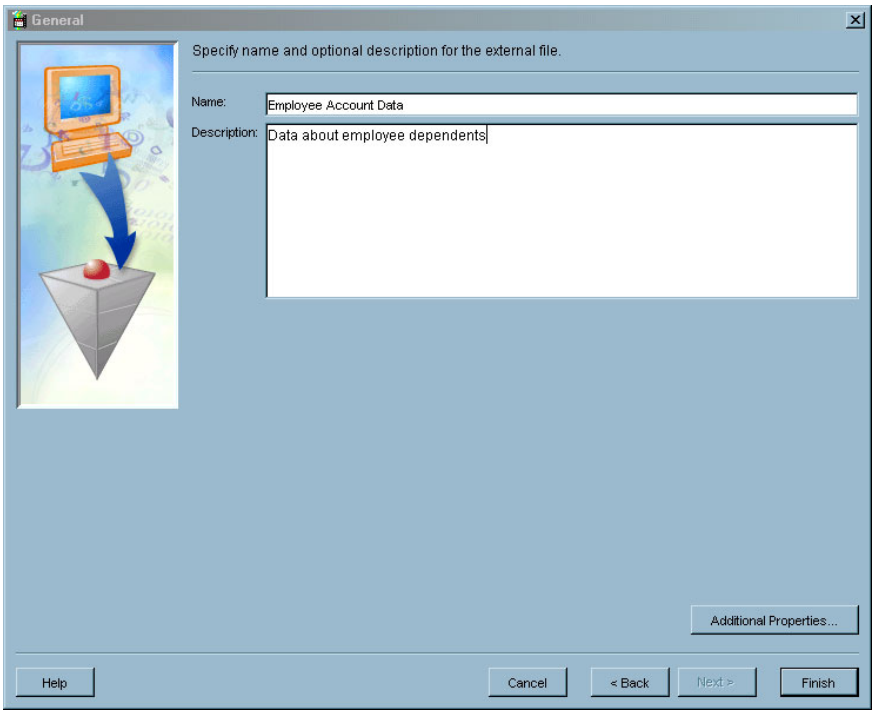
9 Click **Next**. The Select Group window displays.

Display 8.15 Select Group Window



For this example, assume that you do not want to specify a Custom Tree group for the table. Select **Foundation** and click **Next**. The General window displays. You have the option to enter a name and description for the external file metadata, as shown in the following display.

Display 8.16 General Window



If you leave the **Name** field blank, the file is identified by its path.

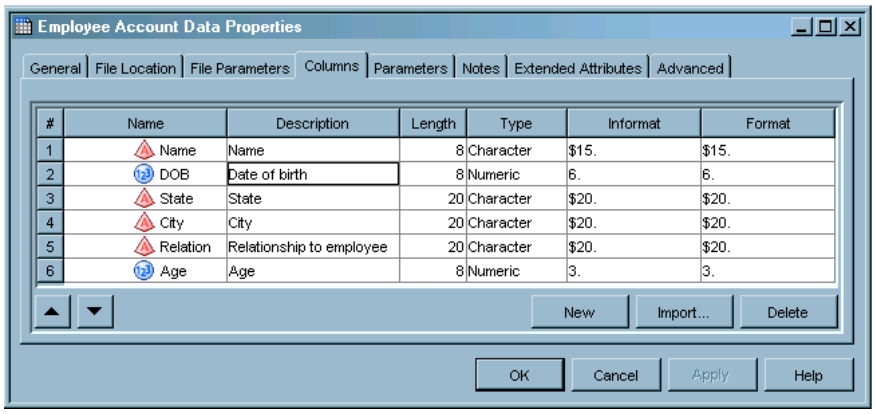
10 Click **Finish** to save the metadata and exit the wizard.

View the External File Metadata

After you have generated the metadata for an external file, you can use SAS Data Integration Studio to view (and possibly make changes to) that metadata. For example, you might want to remove a column from a table or change the data type of a column. Any changes that you make to this metadata will not affect the physical data in the external file. However, the changes will affect the data that is surfaced when the external table is used in SAS Data Integration Studio. Follow these steps to view or update external file metadata:

1 In the Project tree, right-click the icon for the Employee Account Data file and select **Properties** from the pop-up menu. Then, select the **Columns** tab. The Employee Account Data Properties window displays, as shown in the following display.

Display 8.17 Employee Account Data Properties Window



2 Use the tabs in this window to view or update the external file metadata. (Each tab has its own **Help** button.)

3 Click **OK** to save any changes and close the properties window.

View the Data in the External File

To view the data in the Employee Account Data external file, right-click on its metadata object in the Project tree. Then select **View Data** from the pop-up menu. The Employee Account Data external file displays in the View Data window, as shown in the following display.

Display 8.18 Data in the External File, As Specified in Metadata

View Data: Employee Account Data (rows 1-12) - View

Starting row: 1 Go to row

#	Name	DOB	State	City	Relation	Age
1	cary	90353	Kansas	St.Francis	father	48
2	amber	71881	Texas	Austin	daughter	20
3	zachary	90779	Texas	Austin	son	22
4	jennifer	50273	SouthDakota	RapidCity	daughter	28
5	Linda	121855	Wyoming	Sundance	mother	46
6	Tracy	112267	NorthCarolina	W-S	daughter	34
7	Gilbert	20943	NorthCarolina	W-S	father	58
8	Willie	90714	SouthCarolina	Columbia	grandfather	86
9	Madie	82316	NorthCarolina	W-S	mother	84
10	Lucy	20522	NorthCarolina	Troy	grandmother	78
11	Stella	10118	NorthCarolina	Troy	greataunt	81
12	John	90513	SouthCarolina	Columbia	greatuncle	87

If the data in the Employee Account Data external file displays correctly, the metadata for the file is correct and the table is available for use in SAS Data Integration Studio. If you need to review the original data for the file, right-click on its metadata object in the Project tree. Then select **View File** from the pop-up menu.

At this point, you could add the metadata that you just created to a job, or you could check in the metadata.

Check In the Metadata

To check in the metadata object for an external file, follow these steps:

1. In the Project tree, select the repository icon.
2. From the SAS Data Integration Studio menu bar, select **Project ▶ Check In**. All objects in the project repository are checked into the change-managed repository.

Next Tasks

After you have specified the metadata for one or more sources, you can specify the metadata for the corresponding targets—the data stores that will contain the information that will be extracted and transformed from the sources.

CHAPTER

9

Registering Data Targets

Targets: Outputs of SAS Data Integration Studio Jobs

In general, a target is an output of an operation. In a SAS Data Integration Studio job, the main targets are data stores in a data warehouse, data mart, or another data collection.

After you have specified the sources for a SAS Data Integration Studio job, you can specify the targets for that job. Your project plan should identify the targets that are required for a particular job. For example, the targets that are required to answer specific business questions in the Orion Star Sports & Outdoors project are listed under each business question. See the Identifying Targets section under each business question in Chapter 5, "Example Data Warehouse," on page 43.

Use the examples in this chapter, together with general methods that are described in "Registering Sources and Targets" on page 97, to specify metadata for the targets that will be used in a SAS Data Integration Studio job.

Example: Using the Target Table Designer to Register SAS Tables

Preparation

Suppose that you wanted to create a report that shows which sales person is making the most sales, as described in "Which Salesperson Is Making the Most Sales?" on page 45. You decide to extract columns from several existing tables, write that information to a new table, and run a report on the new table. An example of this report is shown in the following display.

Display 9.1 Total Sales by Employee Report

Name	Total Revenue	Emp ID	Job Title	Company	Dept	Section	Group
Alban Kingston	$5,030.00	120170	Sales Rep II	Orion France	Sales	Sales	Racket Sports
Andre Noel	$4,535.00	120411	Sales Rep I	Orion France	Sales	Sales	Winter Sports
Giulia Buono	$6,230.00	120599	Sales Rep II	Orion Italy	Sales	Sales	Children Sports

One way to create a new table whose columns are extracted from existing tables is to create a job that joins the required tables and writes the output of the join to a new table. The new table would be the input to a report transformation. An example of such a process flow is shown in Display 10.3 on page 153.

However, in order to create this process flow, you must first register both the source tables and the new target table. SAS Data Integration Studio needs the target table metadata to specify the content and structure of the information that is extracted from the source tables.

This example describes how to use the Target Table Designer wizard to register a new table whose columns are modeled after the columns in several existing tables. Assume that the following preparations have been made:

- The sales report will be based on a new table that is named Total_Sales_By_Employee. This table will have the same columns in the same order as the report that is shown in Display 9.1 on page 140.
- Total_Sales_By_Employee will be created by a SAS Data Integration Studio job that joins two existing tables: ORDER_FACT and ORGANIZATION_DIM.
- Metadata for ORDER_FACT and ORGANIZATION_DIM is available in a current metadata repository.
- Metadata for the following columns will be imported from the ORGANIZATION_DIM table into the Total_Sales_By_Employee table: Employee_Name, Employee_ID, Job_Title, Company, Department, Section, Org_Group. These fields are needed to identify the employee in the report.
- Metadata for the following column will be imported from the ORDER_FACT table into the Total_Sales_By_Employee table: Total_Retail_Price. This field lists the price of each item sold by an employee. This information will be used calculate the total sales for each employee.
- The Total_Sales_By_Employee table will be in SAS format and will be stored in a SAS library named Report Table Lib.
- The Report Table Lib library has been registered in a current metadata repository. For details about libraries, see "Registering Libraries" on page 59.

- The main metadata repository is under change-management control. In the current example, you must check in the metadata for the target table that you are registering. For details about change management, see "Working with Change Management" on page 113.

Start SAS Data Integration Studio and Open a Metadata Profile

Follow these steps to begin work in SAS Data Integration Studio:

1. Start SAS Data Integration Studio as described in "Starting SAS Data Integration Studio" on page 93.
2. Open the appropriate metadata profile as described in "Opening a Metadata Profile" on page 95.

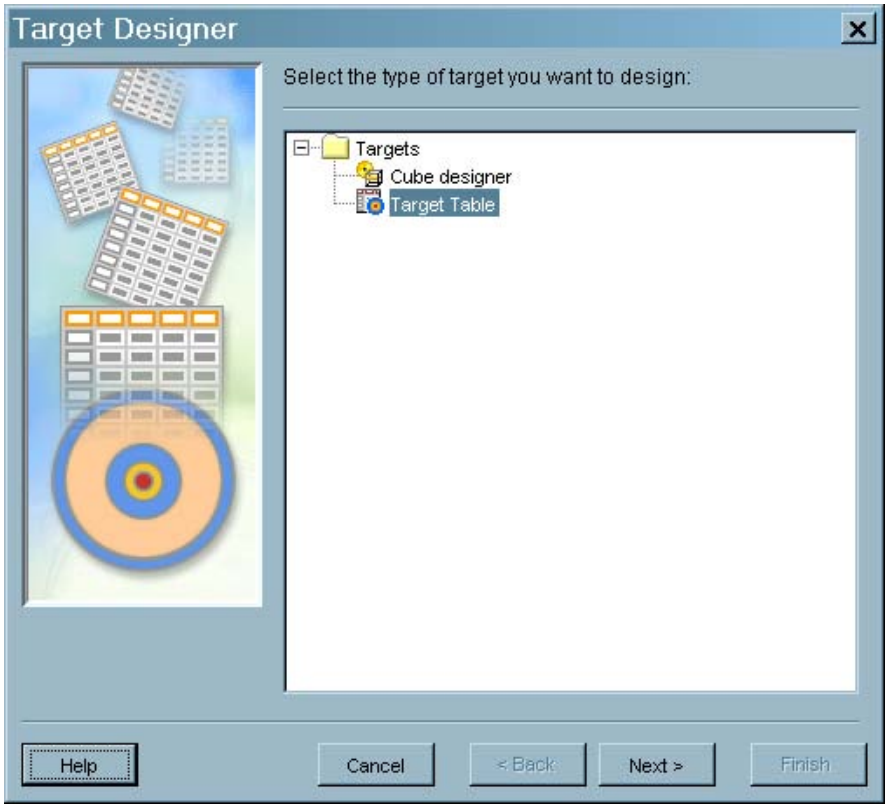
You do not need to check out a library in order to add metadata for tables in that library. Accordingly, the next task is to select the Target Table Designer.

Select the Target Table Designer

Follow these steps to select the Target Table Designer:

1. From the SAS Data Integration Studio menu bar, select **Tools ▶ Target Designer**. The Target Designer selection window displays.

Display 9.2 Target Designer Selection Window



2. The list of wizards on your machine might differ from the list that is shown. Only target types for the Target Designer wizards that have been installed are available.

Follow these steps from this window:

a Click the **Target Table** icon.
b Click **Next**.

Because you have set up a default SAS application server, the wizard tries to open a connection to this default server. If a valid connection to this server exists, you are prompted for a valid user name and a password. After you provide that information, the name and description window displays in the Target Table Designer.

Enter a Name and Description

After you have connected to a SAS application server, use the first window in the wizard to enter a name and description for the table that you want to register.

Note: The metadata object might or might not have the same name as the corresponding physical table. You will specify a name for the physical table later in another window in this wizard. △

In this example, in the **Name** field, enter *Total_Sales_By_Employee* as the name of the metadata object. In the **Description** field, enter the following: **Provides a total sales figure and job information for each salesperson. You can create the table by joining the source tables ORDER_FACT and ORGANIZATION_DIM.**
When the text is complete, click **Next**. The column selection window displays.

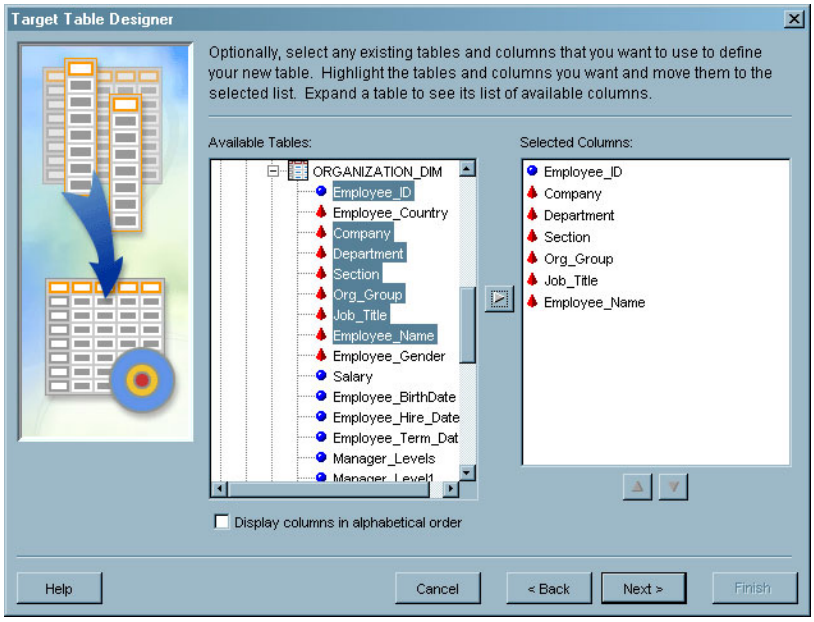
Select Column Metadata from Existing Tables

As noted in "Preparation" on page 140, you will import column metadata from the ORGANIZATION_DIM and ORDER_FACT tables into the metadata for the Total_Sales_By_Employee table. Follow these steps:

1 In the **Available Tables** pane, expand the folder for the repository that contains metadata for the ORGANIZATION_DIM and ORDER_FACT tables.
2 Expand any subfolders, such as the **All Tables** subfolder, to find the metadata for the tables.
3 Find the icon for the table ORGANIZATION_DIM, expand it, and select the desired columns: Employee_Name, Employee_ID, Job_Title, Company, Department, Section, Org_Group.

4 Click the right arrow to copy the selected columns into the **`Selected Columns`** box. The column selection window will look similar to the following display.

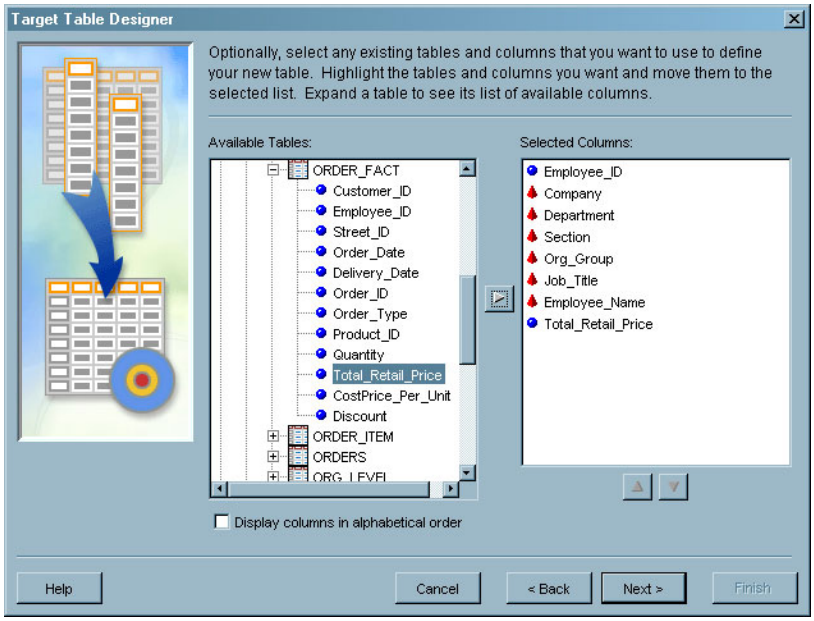
Display 9.3 Columns Selected from the ORGANIZATION_DIM Table



5 Find the icon for the table ORDER_FACT, expand it, and select the desired column: Total_Retail_Price.

6 Click the right arrow to copy the selected column into the **`Selected Columns`** list box. The column selection window will look similar to the following display.

Display 9.4 Column Selected from the ORDER_FACT Table



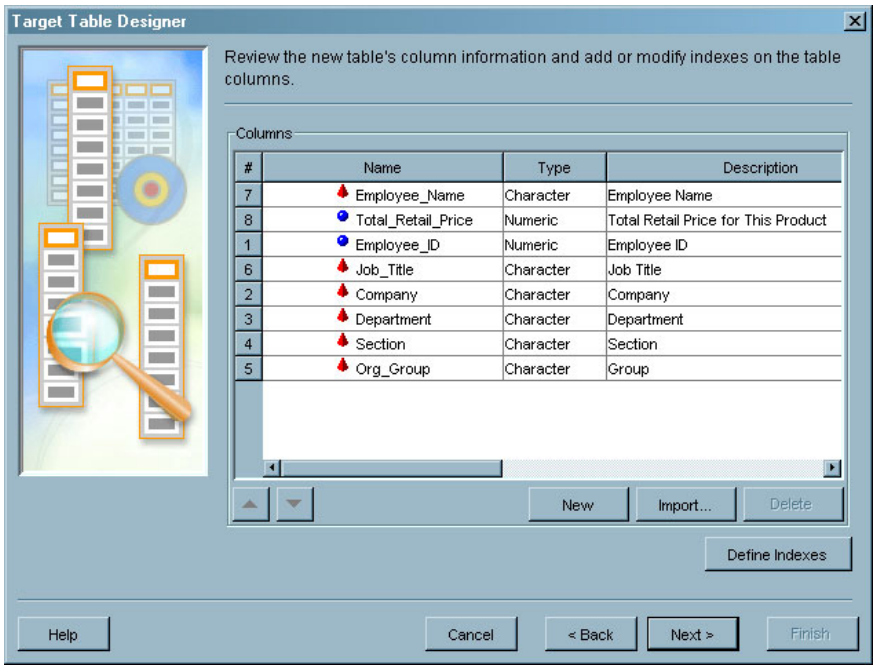
7 When finished, click **`Next`**. The column management window displays.

Specify Column Metadata for the New Table

Use the column management window to accept or modify any column metadata that you selected in the previous window. You can add new columns or modify existing columns in various ways. (For details, click the **Help** button for the window.)

In the current example, we know that a report will be generated from the target table that we are registering. We could reorder the imported columns to more closely match the columns in the report. To reorder columns, click the row number for a column and drag it to the desired place in the list. If we were to reorder the column metadata to match the draft report shown in Display 5.1 on page 45, the column management window would look similar to the following display.

Display 9.5 Column Management Window



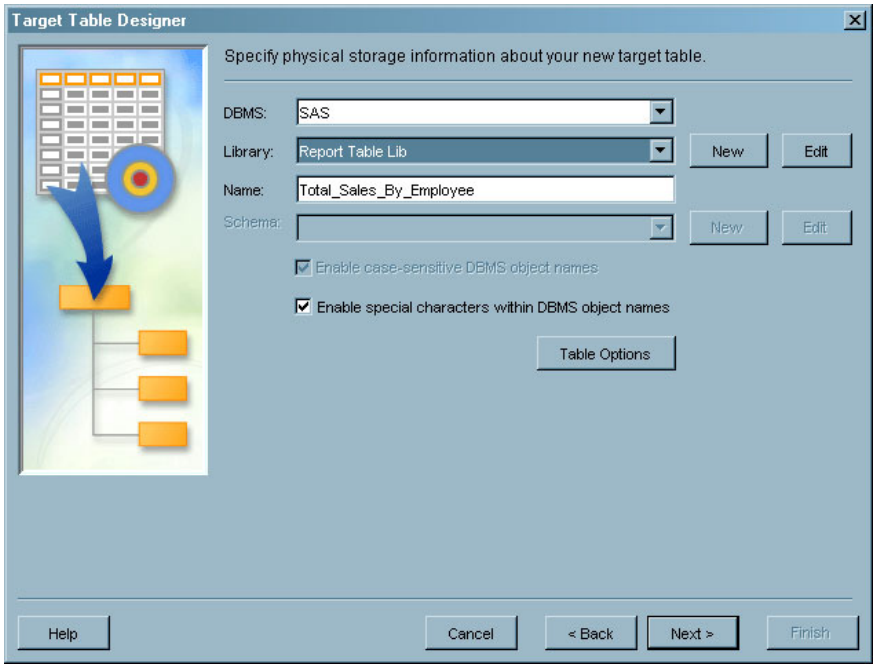
When you are finished reviewing the column metadata, click **Next**. If you have changed the default order of the column metadata, you are prompted to save the new order. In the current example, you would select **Yes** to preserve the order of the columns.

The physical storage window displays.

Specify Physical Storage Information for the New Table

Use the physical storage window to specify the format and location of the table that you are registering, as shown in the following display.

Display 9.6 Physical Storage Window



In this window you specify the database management system that is used to create the target, the library where the target is to be stored, and a valid name for the target.

Follow these steps:

1. Because the target is a SAS data set, in the **DBMS** field, accept the default value of **SAS**.
2. In the **Library** field, use the selection arrow to select **Report Table Lib**.
3. In the **Name** field, enter a valid SAS name for the target. (For details about SAS names, see the SAS Data Integration Studio Help.) For this example, accept the default: Total_Sales_By_Employee.
4. When you are finished specifying physical storage information, click **Next**. The Custom tree group selection window displays.

Usage Hints for the Physical Storage Window

Remember the following as you use the physical storage window:

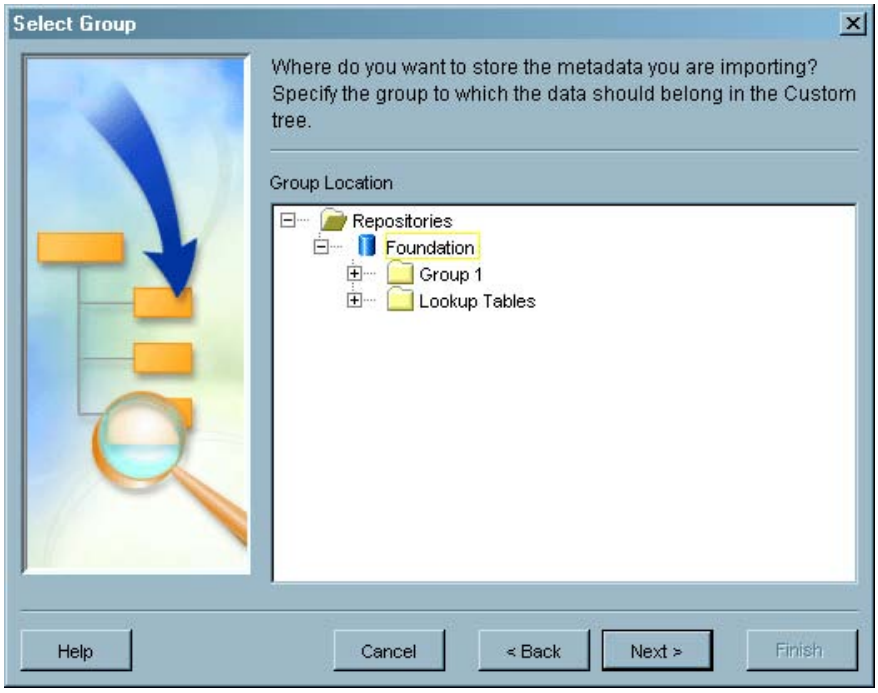
- The name that you specify in the **Name** field must follow the rules for table names in the format that you select in the **DBMS** field. For example, if SAS is the selected DBMS, the name must follow the rules for SAS data sets. If you select another DBMS, the name must follow the rules for tables in that DBMS.
- For a SAS table or a table in a database management system, you can enable the use of mixed-case names or special characters in names. See "Setting Name Options for Individual Tables" on page 109.
- You can specify new libraries or edit the metadata definitions of existing libraries using the **New** and **Edit** buttons.

- You can use the **`Table Option`** button to specify options for SAS tables and tables in a DBMS.

Specify a Custom Tree Group for the Current Metadata

Use the Select Group window to specify a Custom tree group for the table that you are registering. A Custom tree group is a folder that you can use to keep similar kinds of metadata together in the Custom tree on the SAS Data Integration Studio desktop.

Display 9.7 Select Group Window

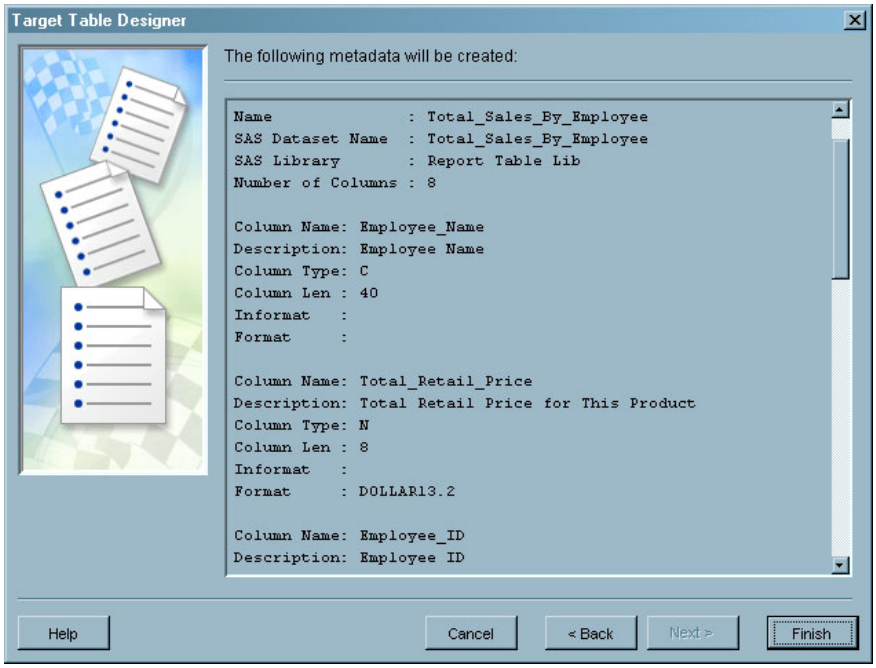


For this example, assume that you do not want to specify a Custom tree group for the table. Select **`Foundation`** and click **`Next`**. The finish window displays.

Save Metadata for the Table

Use the finish window to review the metadata that you entered.

Display 9.8 Finish Window



If the metadata is correct, click **`Finish`**. The metadata for the table is written to a current metadata repository.

Note: The table does not yet exist on the file system. You have only specified the metadata that is used to create the table. △

At this point, you could add the metadata that you just created to a job, or you could check in the metadata.

Check In the Metadata

Follow these steps to check in the metadata for the target table:

1. In the Project tree, select the repository icon.
2. From the SAS Data Integration Studio menu bar, select **Project ▶ Check In Repository**. All objects in the project repository are checked into the change-managed repository.

The new table is now ready to be used in a job, as described in “Example: Creating a Job That Joins Two Tables and Generates a Report” on page 150.

Next Tasks

After you have specified the metadata for one or more targets, you can specify metadata for the job that will read the appropriate sources and create the desired targets on a file system.

CHAPTER 10

Example Process Flows

Using Jobs to Create Process Flows

After you have specified the metadata for one or more sources and targets, you can specify metadata for the job that will read the appropriate sources and create the desired targets in physical storage. Use the examples in this chapter, together with the general steps that are described in “Creating, Running, and Verifying Jobs” on page 99, to specify jobs that will create and load the desired targets.

Example: Creating a Job That Joins Two Tables and Generates a Report

Preparation

Suppose that you wanted to create a report that shows which sales person is making the most sales, as described in "Which Salesperson Is Making the Most Sales?" on page 45. You decide to extract columns from several existing tables, write that information to a new table, and run a report on the new table. An example of this report is shown in the following display.

Display 10.1 Total Sales by Employee Report

Name	Total Revenue	Emp ID	Job Title	Company	Dept	Section	Group
Alban Kingston	$5,030.00	120170	Sales Rep II	Orion France	Sales	Sales	Racket Sports
Andre Noel	$4,535.00	120411	Sales Rep I	Orion France	Sales	Sales	Winter Sports
Giulia Buono	$6,230.00	120599	Sales Rep II	Orion Italy	Sales	Sales	Children Sports

One way to create this report is to create a SAS Data Integration Studio job that joins the required tables and writes the output of the join to a new table. The new table would be the input to a report transformation. This example demonstrates one way to create such a process flow.

In the example, columns from source tables (ORGANIZATION_DIM and ORDER_FACT) are extracted and mapped to columns in the main target table (Total_Sales_By_Employee). The main target table, in turn, is the input to a report transformation (Publish to Archive) that creates the desired report.

Assume the following about the example job:

- Metadata for both source tables in the job (ORGANIZATION_DIM and ORDER_FACT) is available in a current metadata repository. "Identifying Targets" on page 48 describes how these tables were created by combining information from other tables.
- The ORGANIZATION_DIM and ORDER_FACT tables have the same key column: Employee_ID. In the current example, these tables will be joined on the Employee_ID column.
- Metadata for the main target table in the job (Total_Sales_By_Employee) is available in a current metadata repository. "Example: Using the Target Table Designer to Register SAS Tables" on page 140 describes how the metadata for this table could be specified. As described in that section, the Total_Sales_By_Employee table has only those columns that are required for the report. The metadata for these columns is shown in Display 9.5 on page 144.
- You have selected a default SAS application server for SAS Data Integration Studio, as described in "Selecting a Default SAS Application Server" on page 96. This server can access all tables that are used in the job.
- The main metadata repository is under change-management control. For the current example, the metadata for the Total_Sales_By_Employee table must be checked out because (a) the metadata for this table was created and checked in earlier, as described in "Example: Using the Target Table Designer to Register

SAS Tables" on page 140, and (b) the metadata for the table must be updated for the current job. When you are finished with the current job, metadata for the new job and the updated metadata for the Total_Sales_By_Employee table will be checked in. For details about change management, see "Working with Change Management" on page 113.

- It is assumed that you have started SAS Data Integration Studio and have opened the appropriate metadata profile.

The first task is to check out any existing metadata that must be updated for the current job.

Check Out Existing Metadata That Must Be Updated

You do not have to check out the metadata for a table in order to add it as a source or a target in a job. However, the metadata for the Total_Sales_By_Employee table must be checked out because its metadata must be updated for the current job. The Total_Retail_Price column in the table must be updated so that it summarizes a total revenue number from individual sales.

Follow these steps to check out existing metadata:

1. On the SAS Data Integration Studio desktop, select the Inventory tree.
2. In the Inventory tree, open the **`Tables`** folder.
3. Select the table that must be updated for the current job: Total_Sales_By_Employee.
4. Select **Project ▶ Check Out** from the menu bar. The metadata for this table will be checked out and will appear in the Project tree.

The next task is to create and populate the job. In this example, you will populate the job in two stages. First, you will create and test the process flow that loads the Total_Sales_By_Employee table. Then you will add the report generation process to the end of the flow.

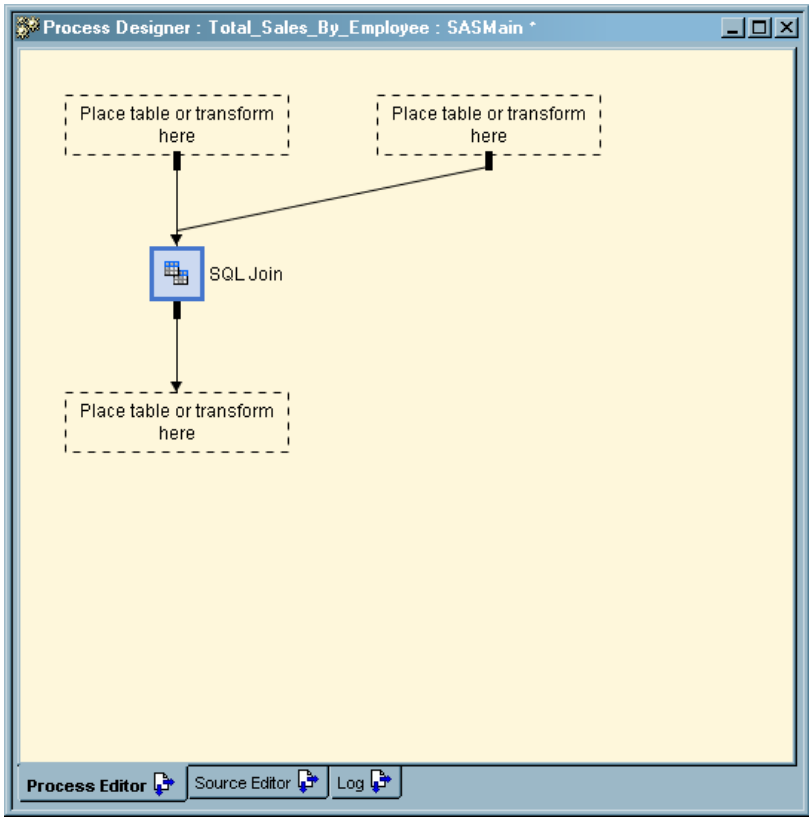
Create the New Job and Specify the Main Process Flow

Follow these steps to populate a job that loads the Total_Sales_By_Employee table. To populate a job means to create a complete process flow diagram, from sources, through transformations, to targets.

1. From the SAS Data Integration Studio menu bar, select **Tools ▶ Process Designer**. The New Job wizard displays.
2. Enter a name and description for the job. Type the name **`Total_Sales_By_Employee`**, press the TAB key, the enter the description **`Generates a report that ranks salespeople by total sales revenue`**.
3. Click **`Finish`**. An empty job will open in the Process Designer window. The job has now been created and is ready to be populated with two sources, a target, and an SQL Join transformation.
4. On the SAS Data Integration Studio desktop, click the **`Process Library`** tab to display the Process Library.
5. In the Process Library, open the **`Data Transforms`** folder.

6 Click, hold, and drag the **SQL Join** transformation into the empty Process Designer window. Release the mouse button to display the SQL Join transformation template in the Process Designer window for the new job. The SQL Join transformation template displays with drop zones for two sources and one target, as shown in the following display.

Display 10.2 The New SQL Join Transformation in the New Job



7 On the SAS Data Integration Studio desktop, click the **Inventory** tab to display the Inventory tree.

8 In the Inventory tree, open the **Tables** folder.

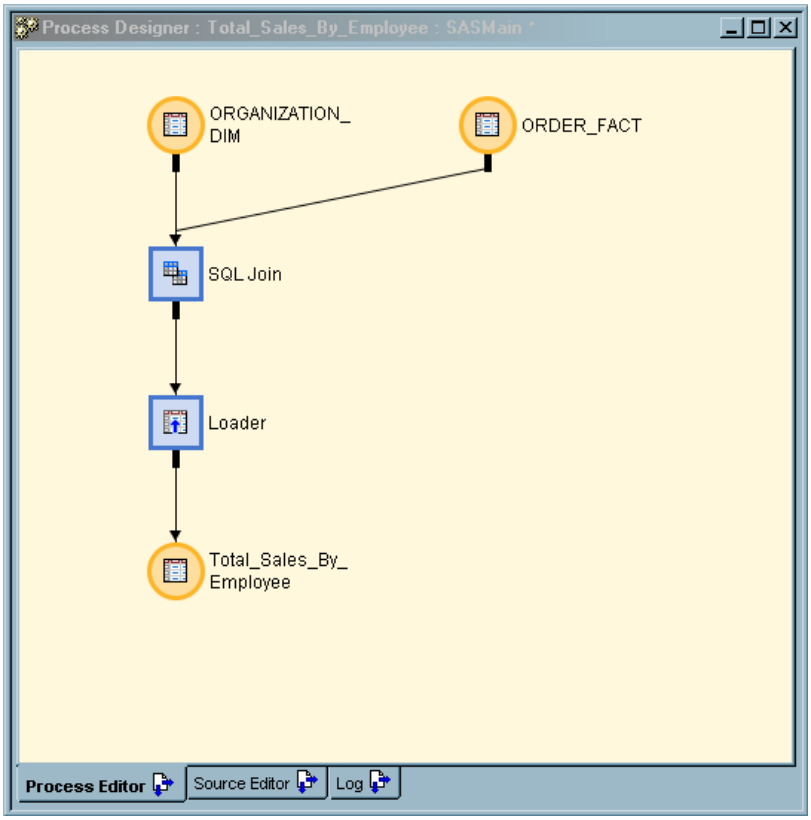
9 In the **Tables** folder, click and drag the **ORGANIZATION_DIM** table into one of the two input drop zones in the Process Designer window, then release the mouse button. The **ORGANIZATION_DIM** table appears as a source in the new job.

10 Repeat the preceding step to specify the **ORDER_FACT** table as the second of the two sources in the new job.

11 On the desktop, click the **Project** tab to display the Project tree. You will see the new job and the Total_Sales_By_Employee table that you checked out.

12 Click and drag **`Total_Sales_By_Employee`** table into the output drop zone in the Process Designer window. The target replaces the drop zone and a Loader transformation appears between the target and the SQL Join transformation template, as shown in the following display.

Display 10.3 Sources and Targets in the Example Job



The job now contains the main flow diagram, from sources, through transformations, to target. The next task is to update the default metadata for the transformations and the target.

(Optional) Reduce the Amount of Data Processed by the Job

As you build the process flow for a job, you might want to create and test part of the flow before adding more processes and tables to the flow. However, repeated executions of a job can become tedious if you have to work with large amounts of data. One way to reduce the amount of data processed by a job is to specify the SAS System option OBS= on the **`Pre and Post Process`** tab in the properties window for the job.

Follow these steps to specify an OBS= option in the properties window for the job that was created in the previous section:

1 In the Process Designer window for the job, right-click the canvas and select **`Properties`** from the pop-up menu. The properties window for the job displays.

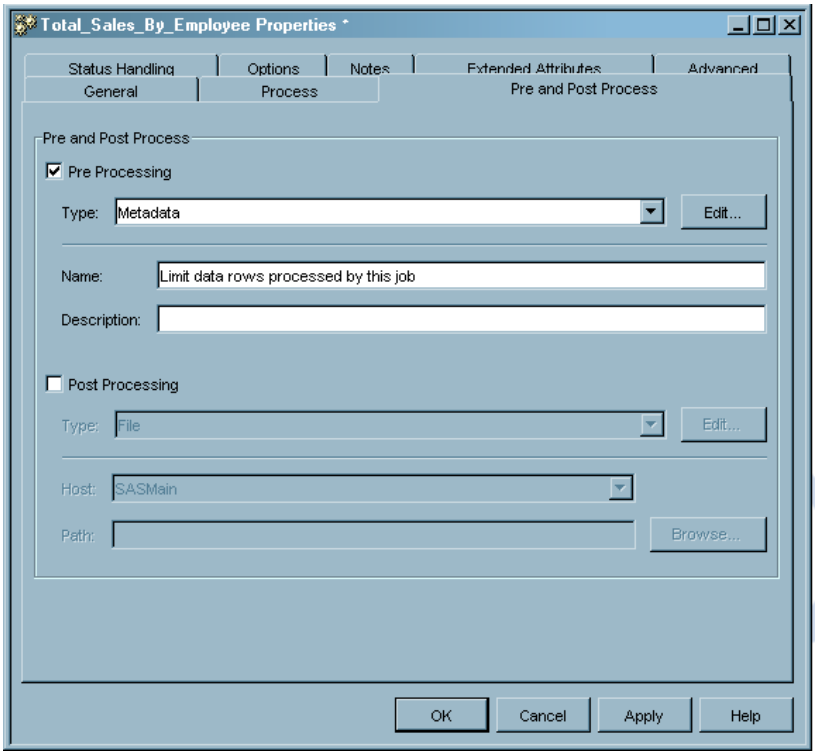
2 Click the **`Pre and Post Process`** tab.

3 Select the **`Pre Processing`** check box.

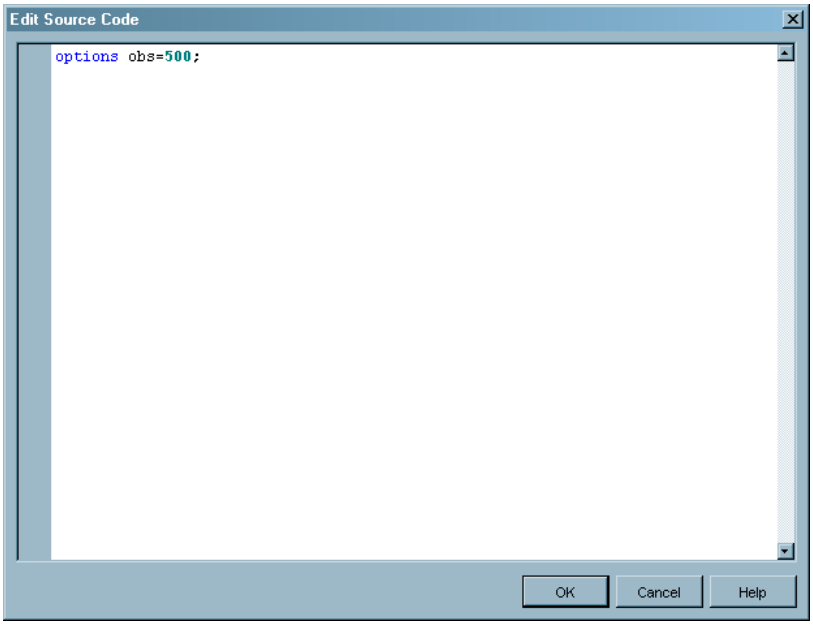
4 In the **`Type`** selection box, select **`Metadata`**.

5 In the **Description** field, enter a description of the option that you are about to set, such as *Limit data rows processed by this job*. The **Pre and Post Process** tab should resemble the following display.

Display 10.4 Pre and Post Processing Tab



6 Click **Edit** to display the window where you will specify the OBS= option. Specify a convenient number of observations (rows of data). For this example, specify OBS=500. The Edit Source Code window should resemble the following display.



The number that you specify in the OBS= option should be high enough so that the transformations in the job will be able to execute successfully; that is, the transformations will have enough data on which to perform their operations.

7 Click **OK** to save the OBS= option.

8 Click **OK** to save your changes in the job properties window.

Note: All inputs in the current job will be limited to the number of data rows that you specified in the OBS= option until you disable this option. △

One way to disable a pre-processing option is to (a) deselect the **Pre Processing** check box on the **Pre and Post Process** tab, and (b) save the job and close the Process Designer window. For more information about setting options on the **Pre and Post Process** tab in the job properties window, see "Adding SAS Code to the Pre and Post Processing Tab" on page 225.

Configure the SQL Join Transformation

Specify Column Mappings

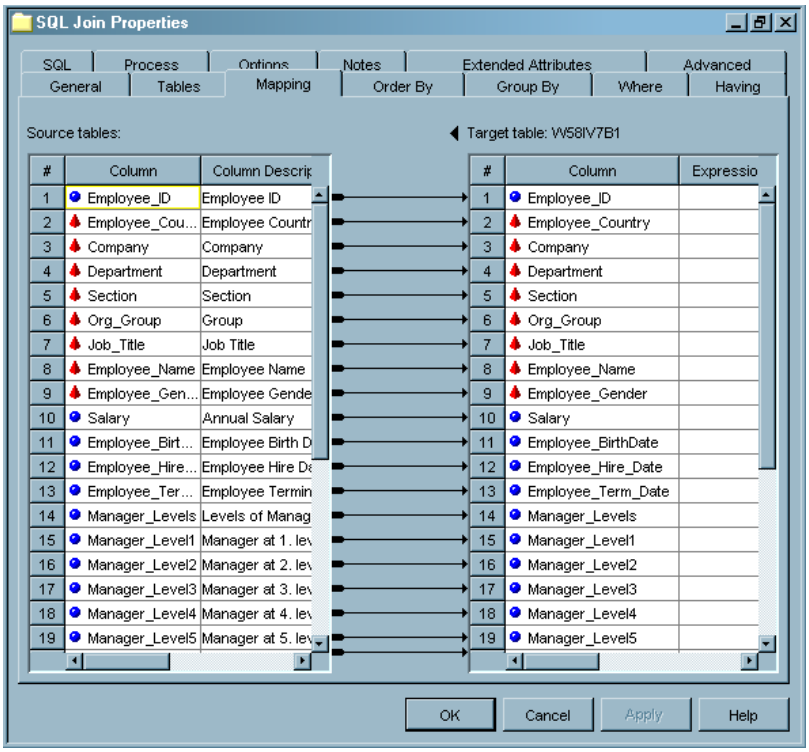
In this section, you will map some columns from the source tables to columns in the temporary output table for the SQL Join transformation. The goal is to map only the columns that are required for the report that you want to create, as shown in Display 10.1 on page 150. The required columns are Employee_Name, Employee_ID, Job_Title, Company, Department, Section, Org_Group, and Total_Retail_Price.

Follow these steps to specify column mappings for the SQL Join transformation:

1 In the Process Designer window, select the **SQL Join** transformation object. Then select **File ▶ Properties** from the menu bar. A properties window displays.

2 Click the **Mapping** tab. By default, the SQL Join transformation maps all columns in the source tables to the same columns in the temporary output table, as shown in the following display.

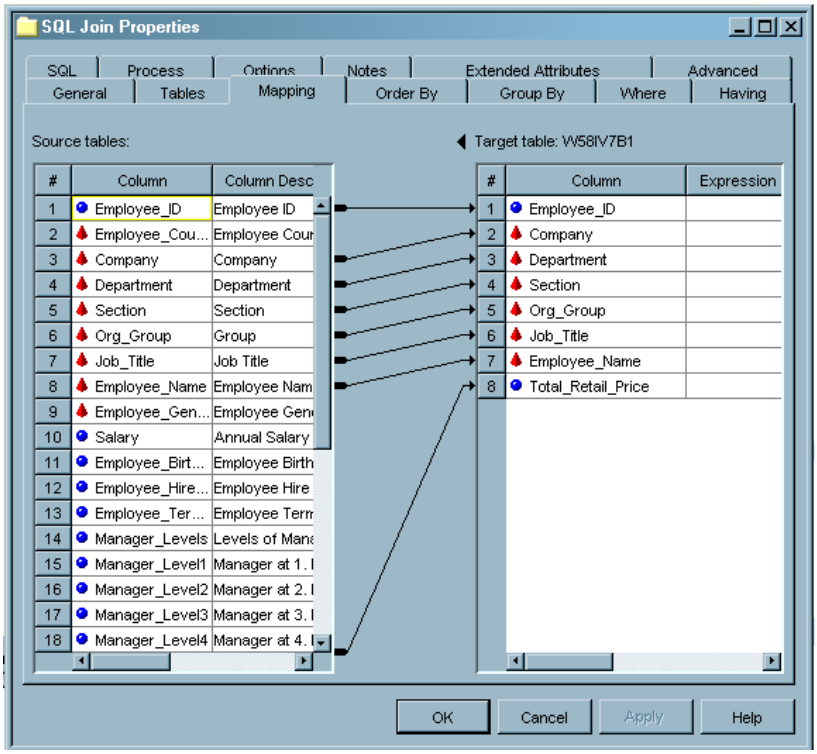
Display 10.5 SQL Join Mapping Tab: Before Extra Columns Are Deleted



However, you need only some of these columns for the report that you want to create. You can simplify the transformation by deleting the metadata for any unneeded columns in the target table.

3 In the **Target Table** pane on the **Mapping** tab, press the CTRL key, left-click the name of each column to be deleted, and select **Delete** from the pop-up menu. When you are finished, the **Mapping** tab will resemble the following display.

Display 10.6 SQL Join Mapping Tab: After Extra Columns Are Deleted



4 Click **Apply** to save your changes without closing the properties window.

5 (Optional) To see how the SQL code is updated based on the contents of the **Mapping** tab and other tabs in the SQL Join transformation, click the **SQL** tab. The code on the **SQL** tab should resemble the following sample:

```
SELECT 'ORGANIZATION_DIM'n.'Employee_ID'n,
 'ORGANIZATION_DIM'n.'Company'n, 'ORGANIZATION_DIM'n.'Department'n,
 'ORGANIZATION_DIM'n.'Section'n, 'ORGANIZATION_DIM'n.'Org_Group'n,
 'ORGANIZATION_DIM'n.'Job_Title'n, 'ORGANIZATION_DIM'n.'Employee_Name'n,
 'ORDER_FACT'n.'Total_Retail_Price'n
FROM 'orstar'n.'ORGANIZATION_DIM'n
 INNER JOIN 'orstar'n.'ORDER_FACT'n
 ON ('ORDER_FACT'n.'Employee_ID'n = 'ORGANIZATION_DIM'n.'Employee_ID'n)
```

The previous SQL statement selects the mapped columns from the ORGANIZATION_DIM and ORDER_FACT tables and joins the result on the Employee_ID column.

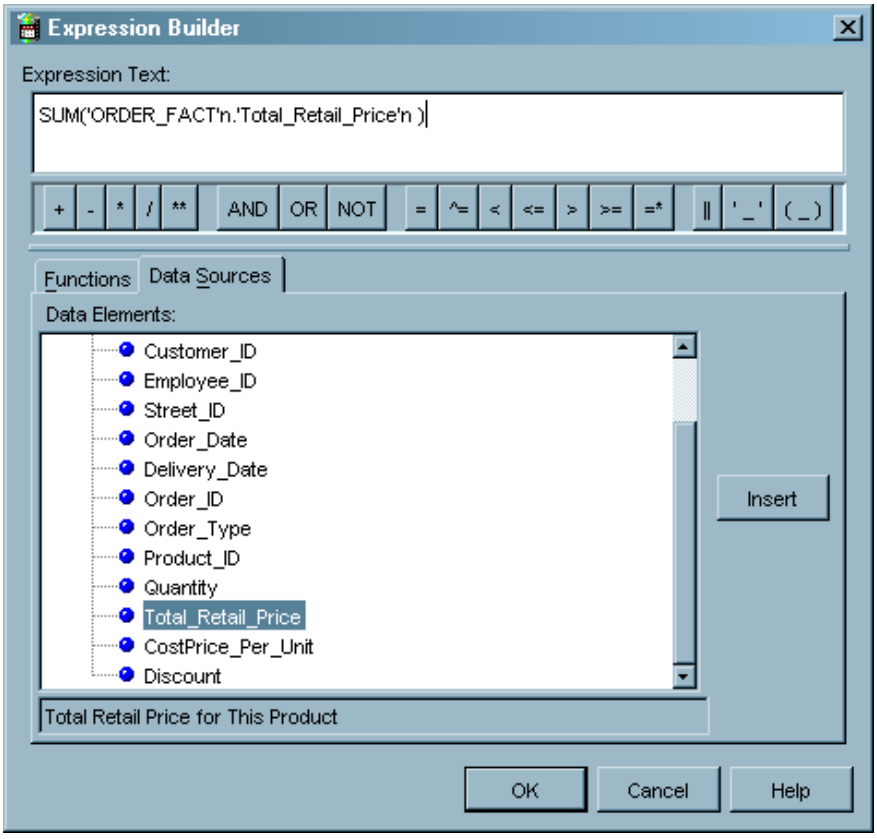
Change One Column to a Calculated Column

The Total_Retail_Price column from the ORDER_DETAIL table contains the price for a particular item that was sold by an employee. However, the report that you want to create shows the total sales for each employee. (See Display 10.1 on page 150.)

Perform these steps to change the Total_Retail_Price column into a derived column (calculated column) that totals all sales for each employee:

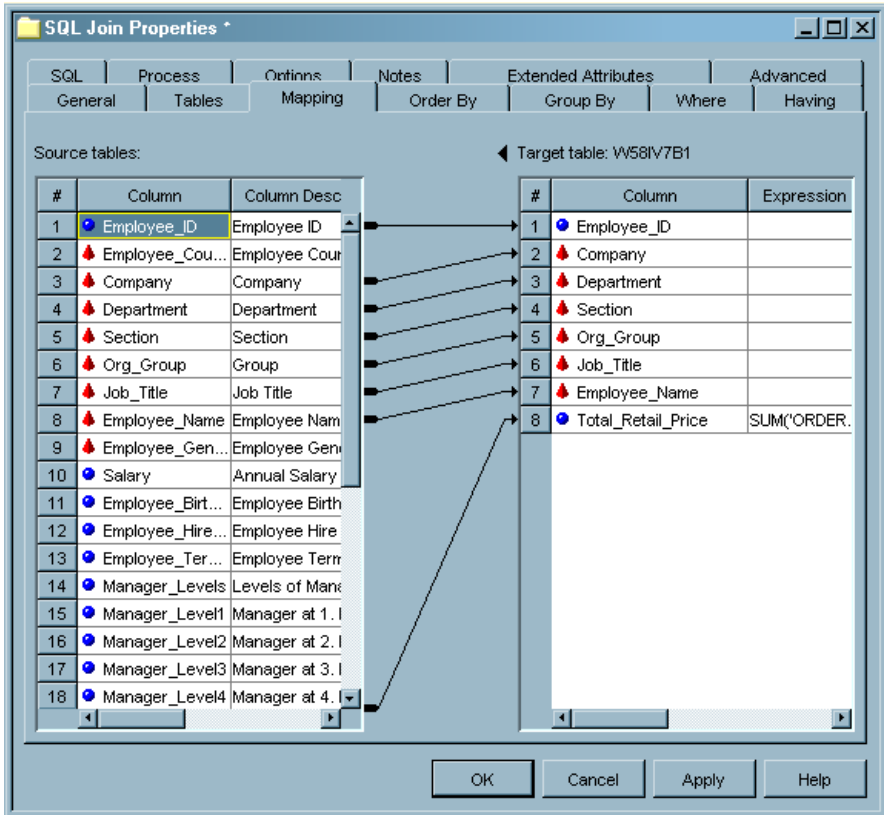
1. In the **Target Table** pane on the right of the **Mapping** tab, scroll to the Total_Retail_Price column.
2. Click twice in the **Expression** attribute for Total_Retail_Price. Then click again in the icon that appears on the right side of the field. This action displays the Expression Builder, which will be used to enter the expression that will summarize individual sales into a total revenue number for each salesperson.
3. In the Expression Builder window, on the **Functions** tab, select the **All Functions** folder. A list of SAS functions is displayed.
4. Scroll to the **SUM(argument)** function, select it, and click **Insert**. The **SUM(argument)** function appears in the **Expression Text** area of the Expression Builder. The argument portion of the expression is selected. The next step is to supply the argument: a column name whose contents are to be summed.
5. Click the **Data Sources** tab in the Expression Builder. A list of tables that are inputs to the current transformation appears.
6. Expand the icon for the ORDER_FACT table, select the Total_Retail_Price column, and click **Insert**. The completed expression appears in the **Expression Text** pane in the Expression Builder, as shown in the following display.

Display 10.7 Completed SUM Expression



7 Click **OK** to save the expression. The Expression Builder window closes. The expression appears in the **Expression** column on the **Mapping** tab, as shown in the following display.

Display 10.8 SQL Join Mapping Tab With a SUM Expression



8 Click **Apply** to save your changes without closing the properties window.

9 (Optional) To see how the SQL code is changed by the expression that you just defined, click the **SQL** tab. The code on the **SQL** tab should resemble the following sample:

```
SELECT 'ORGANIZATION_DIM'n.'Employee_ID'n,
 'ORGANIZATION_DIM'n.'Company'n, 'ORGANIZATION_DIM'n.'Department'n,
 'ORGANIZATION_DIM'n.'Section'n, 'ORGANIZATION_DIM'n.'Org_Group'n,
 'ORGANIZATION_DIM'n.'Job_Title'n, 'ORGANIZATION_DIM'n.'Employee_Name'n,
      SUM('ORDER_FACT'n.'Total_Retail_Price'n) format=8. AS
      'Total_Retail_Price'n
FROM 'orstar'n.'ORGANIZATION_DIM'n
 INNER JOIN 'orstar'n.'ORDER_FACT'n
 ON ('ORDER_FACT'n.'Employee_ID'n = 'ORGANIZATION_DIM'n.'Employee_ID'n)
```

The previous SQL statement selects the mapped columns from the ORGANIZATION_DIM and ORDER_FACT tables, summarizes the contents of the Total_Retail_Price column, and joins the result on the Employee_ID column.

Specify GROUP BY Options for the Join

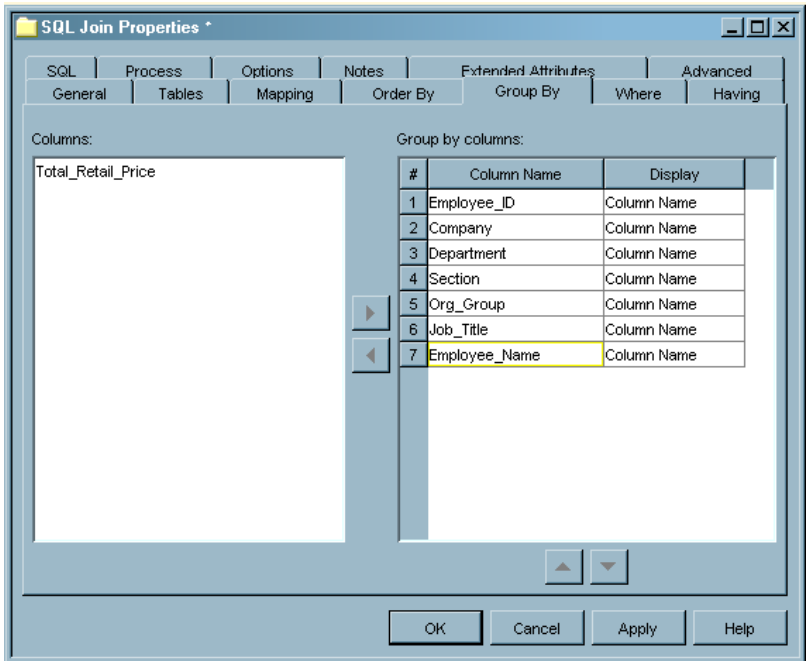
In the previous section, the Total_Retail_Price column in the join result table was changed to a calculated column that sums all of its values. All retail transactions for all employees would be added together. However, the report that you want to create summarizes the retail transactions for each employee.

To limit the scope of the Total_Retail_Price column to each employee, add a GROUP BY option that applies to all of the other mapped columns in the join. That is, you would specify the following columns in the **`Group By`** tab of the SQL Join transformation: Employee_Name, Employee_ID, Job_Title, Company, Department, Section, and Org_Group.

Follow these steps to specify GROUP BY options in the SQL Join transformation:

1. In the properties window for the SQL Join transformation, click the **`Group By`** tab.
2. In the **`Columns`** pane, select all columns except the Total_Retail_Price column. Then click the right arrow to move the columns into the **`Group by columns`** pane. The **`Group By`** tab should resemble the following display.

Display 10.9 Columns Specified on the Group By Tab



3. Click **`Apply`** to save your changes without closing the properties window.
4. (Optional) To see how the SQL code is changed by the expression that you just defined, click the **`SQL`** tab. The code on the **`SQL`** tab should resemble the following sample:

```
SELECT 'ORGANIZATION_DIM'n.'Employee_ID'n,
 'ORGANIZATION_DIM'n.'Company'n, 'ORGANIZATION_DIM'n.'Department'n,
 'ORGANIZATION_DIM'n.'Section'n, 'ORGANIZATION_DIM'n.'Org_Group'n,
 'ORGANIZATION_DIM'n.'Job_Title'n, 'ORGANIZATION_DIM'n.'Employee_Name'n,
          SUM('ORDER_FACT'n.'Total_Retail_Price'n) format=8. AS
          'Total_Retail_Price'n
FROM 'orstar'n.'ORGANIZATION_DIM'n
 INNER JOIN 'orstar'n.'ORDER_FACT'n
 ON ('ORDER_FACT'n.'Employee_ID'n = 'ORGANIZATION_DIM'n.'Employee_ID'n)
```

```
GROUP BY 'ORGANIZATION_DIM'n.'Employee_ID'n,
 'ORGANIZATION_DIM'n.'Company'n, 'ORGANIZATION_DIM'n.'Department'n,
 'ORGANIZATION_DIM'n.'Section'n, 'ORGANIZATION_DIM'n.'Org_Group'n,
 'ORGANIZATION_DIM'n.'Job_Title'n, 'ORGANIZATION_DIM'n.'Employee_Name'n
```

5 The updates to the SQL Join transformation are complete. Click **OK** to save your changes and close the properties window for the transformation.

Update the Metadata for the Total Sales By Employee Table

When the metadata for the Total Sales By Employee table was created, it was not fully optimized for the report shown in Display 10.1 on page 150. In order to produce the report, you must make the following changes to the column metadata for the Total_Sales_By_Employee table:

- Rename the **Total_Retail_Price** column to **Total_Revenue** to better match the summarized data that you specified in "Change One Column to a Calculated Column" on page 157.
- Specify formatting for the **Total_Revenue** column so that its contents will display properly in the report.

Follow these steps to update column metadata for Total Sales By Employee table:

1 In the Process Designer window, select **Total_Sales_By_Employee**, then select **File ▶ Properties** from the menu bar. A properties window displays.

2 In the properties window, click the **Columns** tab.

3 Click the **Total_Retail_Price** column. Change the name to **Total_Revenue**.

4 In the **Total_Revenue** column, scroll right to display the **Format** column. Enter the format **DOLLAR13.2** to specify the appearance of this column in the HTML output file.

5 In the **Total_Revenue** column, click twice in the **Sort** column to display a pull-down icon. Click the icon and select the DSCFORMATTED option. This option sorts the rows in descending order, based on the formatted value of the **Total_Revenue** column.

6 Click **OK** to save your changes and close the properties window for the table.

Configure the Loader Transformation

You must make two updates to the Loader transformation for the Total Sales By Employee table:

- On the **Mapping** tab in the transformation properties window, map the source table column **Total_Retail_Price** to the target table column **Total_Revenue**. The individual sales figures in the **Total_Retail_Price** column are summarized in the **Total_Revenue** column., so these two columns must be mapped to each other. The Loader transformation will automatically map source and target columns with the same name, but not map columns with different names.
- On the **Load Techniques** tab, select the **Drop Table** option. For SAS tables such as Total Sales By Employee, the **Drop Table** option conserves disk space.

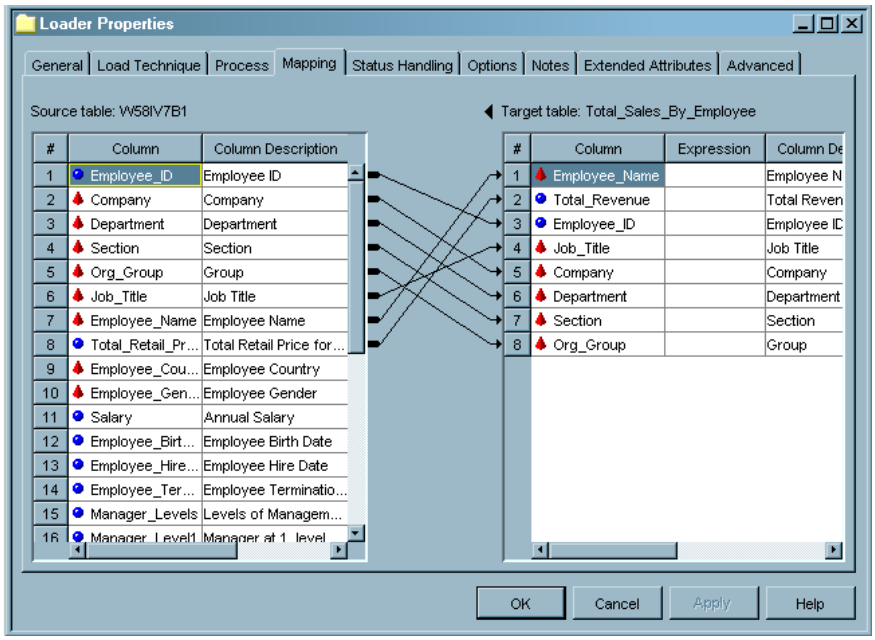
Follow these steps to update the Loader transformation for the Total Sales By Employee table:

1 In the Process Designer window, select the **Loader** transformation. Then select **File ▶ Properties** from the menu bar. A properties window displays.

2 In the properties window, click the **Mapping** tab.

3 On the **Mapping** tab, position the cursor on the name of the source column **Total_Retail_Price**, then click and drag the mapping arrow to the target column **Total_Revenue**. These two columns are now mapped. The **Mapping** tab should resemble the following display.

Display 10.10 Column Mapping in the Loader



4 In the properties window, click the **Load Technique** tab. Select the **Drop Target** radio button to replace the physical table each time the job is run.

The Loader is now configured and is ready to run.

Run the Job and Check the Log

At this point, you have a process flow that will join two source tables, extract information from columns, calculate total sales revenue for each employee, and write the results to the Total_Sales_By_Employee table. Before you add a transformation that will produce the desired report, you might want to verify that the current process flow will produce the information that is required in the report.

1 With the job displayed in the Process Designer window, select **Process ▶ Submit** from the menu bar. SAS Data Integration Studio generates code for the job and submits the code to a SAS application server. The server executes the code. A pop-up window displays to indicate that the job is running.

2 If a pop-up error message appears, or if you want to look at the log for the completed job, click the **Log** tab in the Process Designer window.

3 On the **Log** tab, scroll through the SAS log information that was generated during the execution of the job.

The code that was executed for the job is available on the **Source Code** tab in the Process Designer window.

4 If you find errors in the source code for a step, select the corresponding transformation in the process flow diagram, then select **File ▶ Properties** from the menu bar. A properties window displays.

5 Correct the metadata and resubmit the job until there are no more errors.

6 After the job runs without error, save the job. Select **File ▶ Save** from the menu bar.

The next task is to verify that the job created the correct output.

Verify the Contents of the Total_Sales_By_Employee Table

The data that was loaded into the Total_Sales_By_Employee table should be appropriate for the report that you want to create (Display 10.1 on page 150). To view data in the Total_Sales_By_Employee table, select the table, then select **View ▶ View Data** from the menu bar. The View Data window opens, as shown in the following display.

Display 10.11 Contents of the Total_Sales_By_Employee Table

View Data: Total_Sales_By_Employee (rows 1-100)

#	Employee Name	Total Revenue	Employee ID	Job Title	Company	Departr
1	Irenie Elvish	$322,680.49	120121	Sales Rep. II	Orion Australi...	Sales
2	Christina Ngan	$307,296.90	120122	Sales Rep. II	Orion Australi...	Sales
3	Kimiko Hotstone	$197,793.43	120123	Sales Rep. I	Orion Australi...	Sales
4	Lucian Daymond	$307,615.35	120124	Sales Rep. I	Orion Australi...	Sales
5	Fong Hofmeister	$54,703.41	120125	Sales Rep. IV	Orion Australi...	Sales
6	Satyakam Denny	$11,170.96	120126	Sales Rep. II	Orion Australi...	Sales
7	Sharryn Clarkson	$409,696.11	120127	Sales Rep. II	Orion Australi...	Sales
8	Monica Kletschkus	$22,322.56	120128	Sales Rep. IV	Orion Australi...	Sales
9	Alvin Roebuck	$92,939.92	120129	Sales Rep. III	Orion Australi...	Sales
10	Kevin Lyon	$53,388.15	120130	Sales Rep. I	Orion Australi...	Sales
11	Marinus Surawski	$198,058.55	120131	Sales Rep. I	Orion Australi...	Sales
12	Fancine Kaiser	$238,503.22	120132	Sales Rep. III	Orion Australi...	Sales
13	Petrea Soltau	$17,860.90	120133	Sales Rep. II	Orion Australi...	Sales

The data in the Total_Sales_By_Employee table appears to be appropriate for the report. The next task is to add a report transformation to the end of the process flow.

Add the Publish to Archive Transformation to the Process Flow

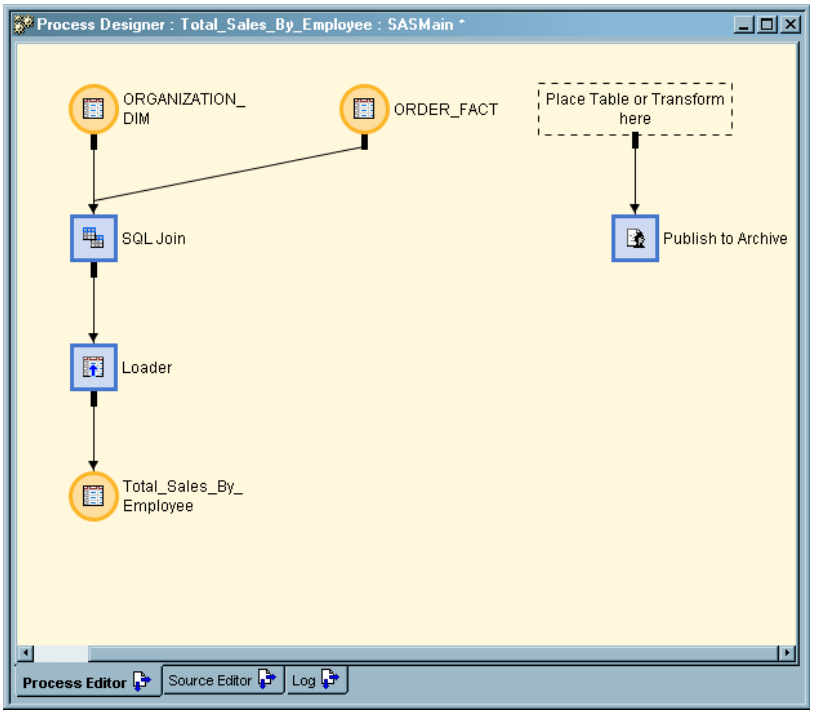
The Publish to Archive transformation generates a SAS package file and an optional HTML report. The package file can be published by SAS programs that use the publishing functions in SAS Integration Technologies software.

Follow these steps to add the Publish to Archive transformation to the process flow:

1 From the SAS Data Integration Studio desktop, click the **`Process`** tab to display the Process Library.

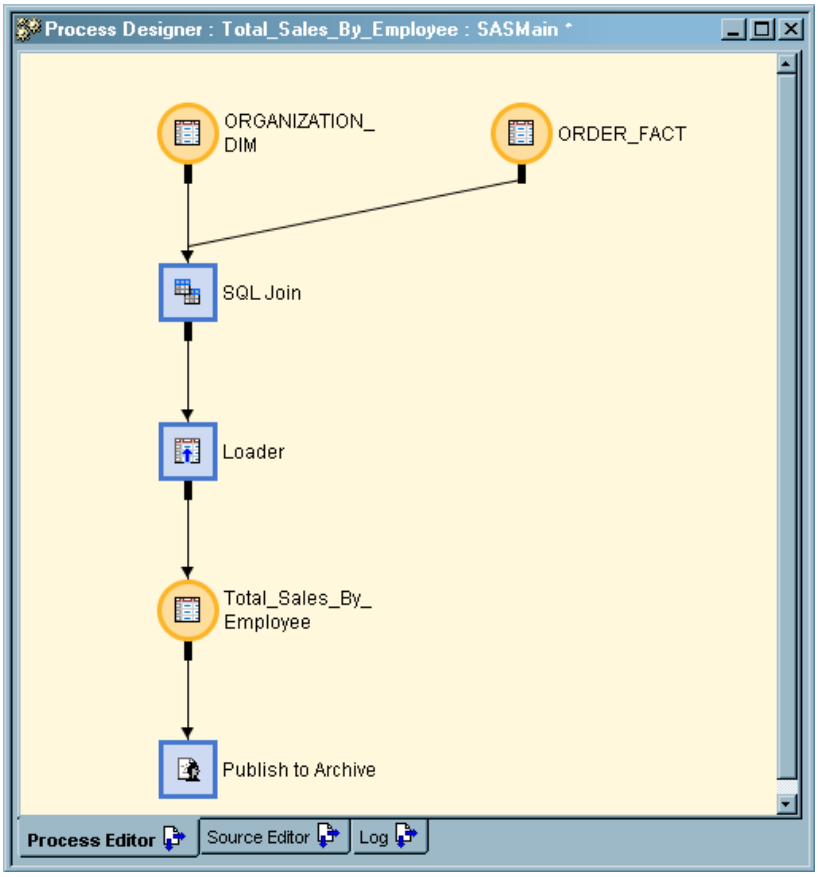
2 In the Process Library, open the **Publish** folder. Click and drag the **Publish to Archive** transformation into any location in the Process Designer and release the mouse button. As shown in the following display, an icon and an input drop zone appear in the Process Designer.

Display 10.12 Example Job with Publish to Archive



3 In the Process Designer window, click and drag the target **`Total_Sales_By_Employee`** over the input drop zone for the **`Publish to Archive`** transformation. Release the mouse button to identify the target as the source for **`Publish to Archive`**, as shown in the following display.

Display 10.13 Publish to Archive Transformation at the End of a Process Flow



4 You now have a complete process flow, including a transformation that will generate an HTML report. The next task is to configure the report transformation so that it will create the desired report.

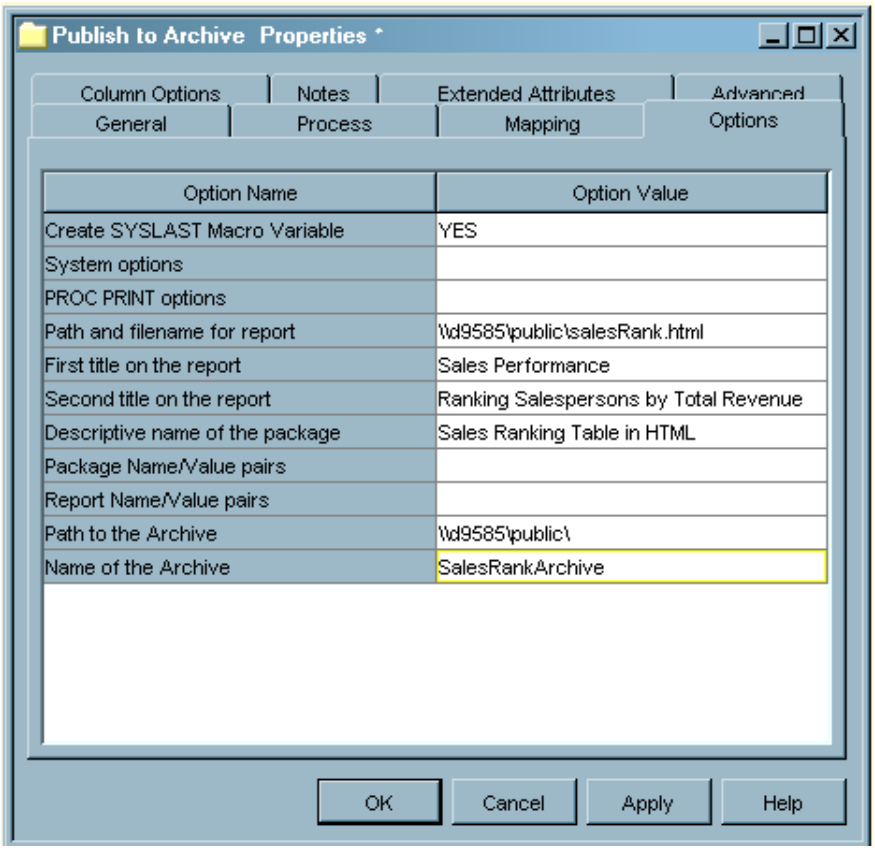
Configure the Publish to Archive Transformation

Follow these steps to configure HTML output using the Publish to Archive transformation.

1 In the Process Designer window, select the **`Publish to Archive`** transformation. Then select **File ▶ Properties** from the menu bar. A properties window displays.

2 In the properties window, click the **`Options`** tab. Type in values for the fields that are shown in the following display.

Display 10.14 Options in the Publish to Archive Transformation



3 Click **`OK`** to save input and close the properties window. The Publish to Archive transformation, and the entire job, are now ready to run.

Run the Job and Check the Log

Same as "Run the Job and Check the Log" on page 162. The next task is to verify that the job created the desired report.

Check the HTML Report

Using the path and file name that you specified in the Publish to Archive transformation, open the HTML report in a browser. The report should resemble the following display.

Display 10.15 HTML Report Generated by the Example Job

Sales Performance
Ranking Salespersons by Total Revenue

Obs	Employee_Name	Total_Revenue	Employee_ID	Job_Title	Company	Department	Section	Org_Group
1	Irenie Elvish	$322,680.49	120121	Sales Rep. II	Orion Australia	Sales	Sales	Assorted Sports Articles
2	Christina Ngan	$307,296.90	120122	Sales Rep. II	Orion Australia	Sales	Sales	Assorted Sports Articles
3	Kimiko Hotstone	$197,793.43	120123	Sales Rep. I	Orion Australia	Sales	Sales	Assorted Sports Articles
4	Lucian Daymond	$307,615.35	120124	Sales Rep. I	Orion Australia	Sales	Sales	Assorted Sports Articles
5	Fong Hofmeister	$54,703.41	120125	Sales Rep. IV	Orion Australia	Sales	Sales	Children Sports

Compare the HTML report to the draft report shown in Display 10.1 on page 150. If changes are needed, change the properties of the Total_Sales_By_Employee table or the transformations in the flow. If the report is correct, you can check in the job.

Check In the Metadata

To check in all objects in the Project tree:

1. In the Project tree, select the repository icon.
2. From the SAS Data Integration Studio menu bar, select **Project ▸ Check In Repository**. All of the objects in the project repository are checked into the change-managed repository.

Example: Creating a Data Validation Job

Preparation

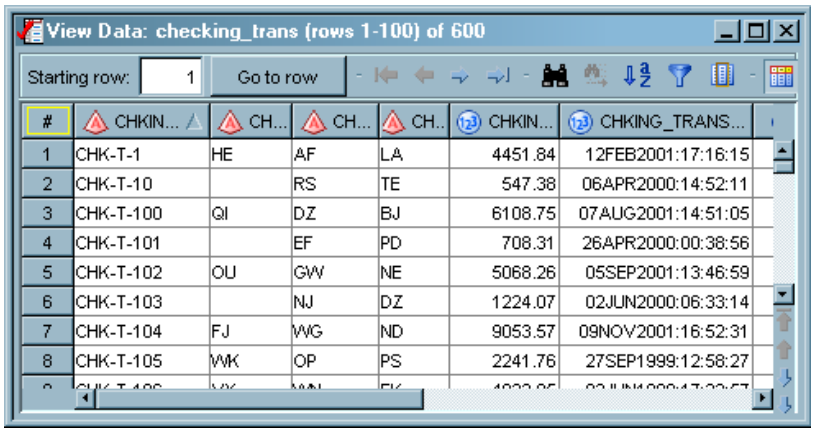
Suppose that you wanted to validate the data from a source table before writing it to a target table. SAS Data Integration Studio has several features that can help you to improve the quality of your data, as described in "Working With SAS Data Quality Software" on page 104.

This example demonstrates how to use the Data Validation transformation in a job that prepares checking transaction data for loading into a fact table. In the example job, the Data Validation transformation writes erroneous rows into an error table. Errors are detected based on a comparison of a source column to a column of valid values in a lookup table. The transformation also replaces missing values in another source column with a standard text string; any duplicate values that are found in a third source column result in the termination of the job.

Assume that the following preparations have been made.

- The source table contains information on checking account transactions, including status codes and the transaction amount, date, time.

Display 10.16 Contents of the Source Table CHECKING_TRANS



#	CHKIN...	CH...	CH...	CH...	CHKIN...	CHKING_TRANS...
1	CHK-T-1	HE	AF	LA	4451.84	12FEB2001:17:16:15
2	CHK-T-10		RS	TE	547.38	06APR2000:14:52:11
3	CHK-T-100	QI	DZ	BJ	6108.75	07AUG2001:14:51:05
4	CHK-T-101		EF	PD	708.31	26APR2000:00:38:56
5	CHK-T-102	OU	GW	NE	5068.26	05SEP2001:13:46:59
6	CHK-T-103		NJ	DZ	1224.07	02JUN2000:06:33:14
7	CHK-T-104	FJ	WG	ND	9053.57	09NOV2001:16:52:31
8	CHK-T-105	WK	OP	PS	2241.76	27SEP1999:12:58:27

- The lookup table CHECKING_TRANS_LOOKUP contains several columns of valid values that can be used to test incoming source values.
- The error table CHECKING_TRANS_ERROR and the target table CHECKING_TRANS_VALID have the same column definitions (column metadata) as the source table.
- Metadata for all four tables is available in a current metadata repository.
- The main metadata repository is under change-management control. You do not have to check out the metadata for any of the tables used in the current example. (The metadata for these tables will not be updated.)
- You have selected a default SAS application server for SAS Data Integration Studio, as described in "Selecting a Default SAS Application Server" on page 96. This server can access all tables that are used in the job.
- It is assumed that you have started SAS Data Integration Studio and have opened the appropriate metadata profile.

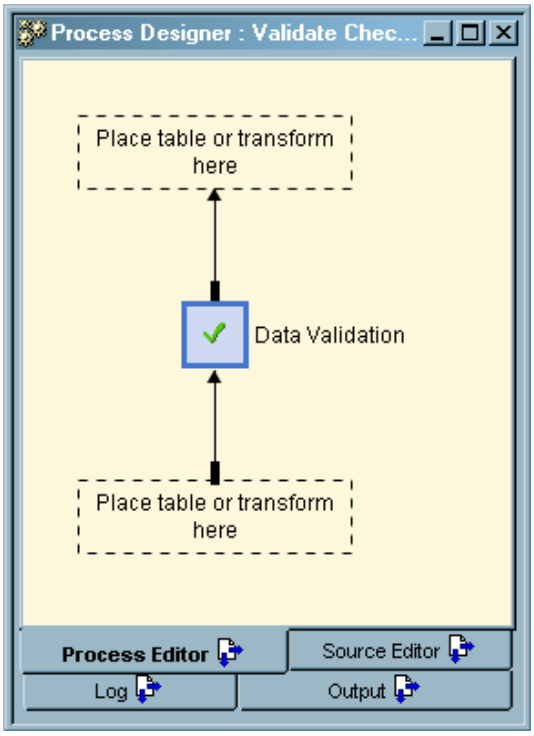
Create and Populate the New Job

Follow these steps to create and populate the new job:

1. From the SAS Data Integration Studio menu bar, select **Tools ▶ Process Designer** to display the New Job wizard.
2. Type the job name **`Validate Checking Transactions`**, press the TAB key, and the enter the description **`Cleanses checking account transaction data prior to loading`**.
3. Click **`Finish`**. An empty job opens in the Process Designer window. The job has now been created and is ready to be populated.
4. From the SAS Data Integration Studio desktop, click the **`Process Library`** tab to display the Process Library.
5. In the Process Library, open the **`Data Transforms`** folder.
6. Click, hold, and drag the **`Data Validation`** transformation into the empty Process Designer window. Release the mouse button to display the transformation in the

Process Designer window. The Data Validation transformation displays with drop zones for the source and target, as shown in the following display.

Display 10.17 The Data Validation Transformation in the New Job



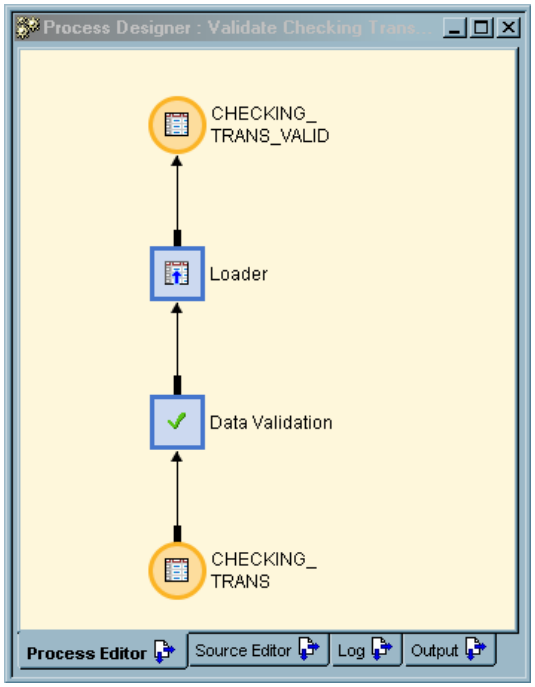
7 From the SAS Data Integration Studio desktop, click the **Inventory** tab to display the Inventory tree.

8 In the Inventory tree, open the **Tables** folder.

9 In the **Tables** folder, click and drag the **CHECKING_TRANS** table into the source drop zone in the Process Designer window, then release the mouse button. The table appears as the source in the new job.

10 Click and drag the table **CHECKING_TRANS_VALID** into the target drop zone. The table appears as the target in the new job, along with a Loader transformation, as shown in the following display.

Display 10.18 Sources and Targets in the Data Validation Job



The job now contains a complete process flow diagram. The next task is to configure the Data Validation transformation.

Configure the Data Validation Transformation

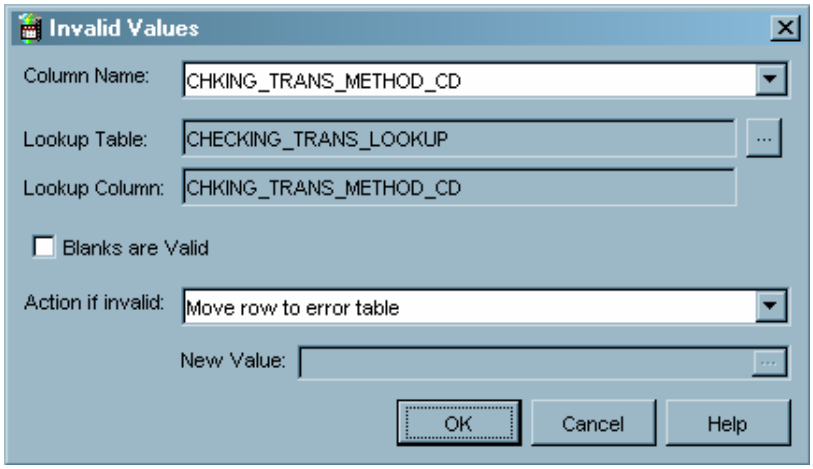
The example job has now been populated with a source, a target, and some transformations. Follow these steps to configure the Data Validation transformation:

1 In the Process Designer window, double-click the **Data Validation** transformation to display the transformation's properties window. To use a lookup table to validate values, click the **Invalid Values** tab.

2 On the **Invalid Values** tab, click **New** to display the Invalid Values window. Click **Column Name**, display the columns of the CHECKING_TRANS table, and select the column CHKING_TRANS_METHOD_CD.

3 Click the browse button to the right of the **Lookup Table** field and select CHECKING_TRANS_LOOKUP. Then click **OK** to return to the Data Validation properties window.

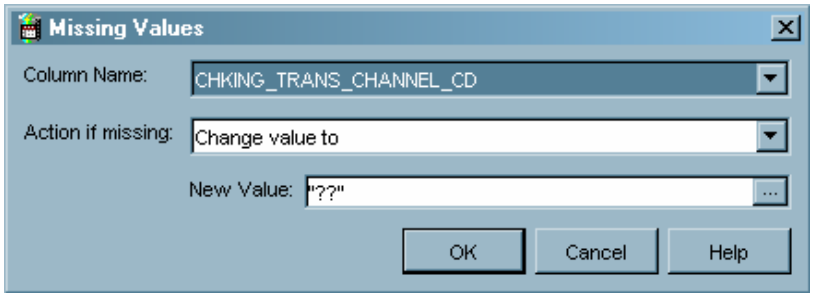
Display 10.19 Invalid Values Window



4 To specify the name of the error table, click the **Options** tab. In the Error Table row, enter the SAS libref and filename (rather than the metadata names) of the error table, such as **ORGOLD.CHECKING_TRANS_ERROR**. Click **Apply** to store your input.

5 To replace missing values, click the **Missing Values** tab, then click **New**. In the Missing Values window, select the column CHKING_TRANS_CHANNEL_CD. In the **Action if missing field**, select **Change value to**. In the New Value field, type **"??"**, with quotation marks. Click **OK** to return to the properties window.

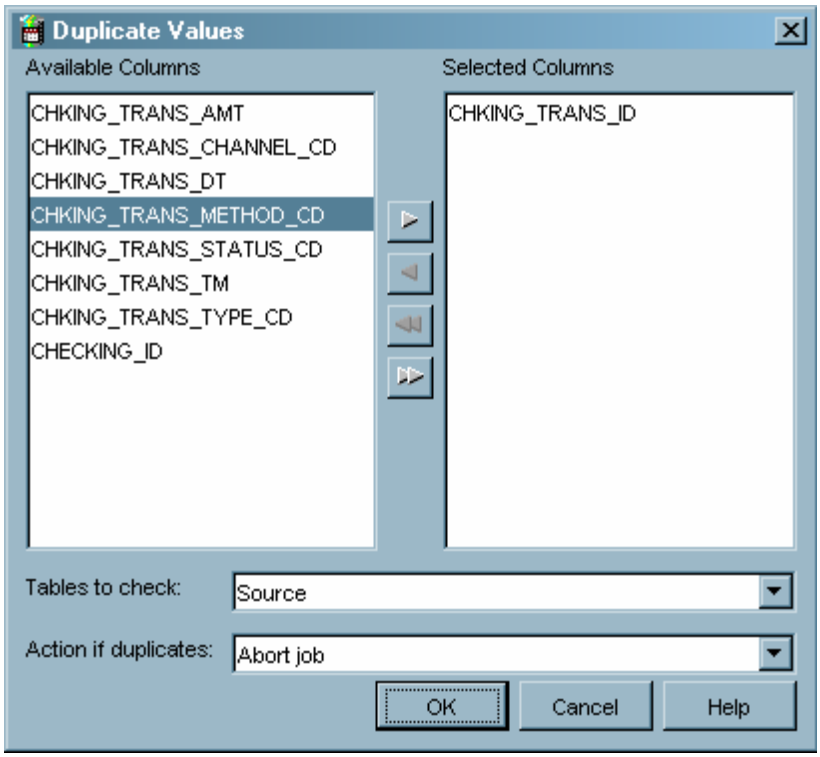
Display 10.20 Missing Values Window



6 To configure the detection of duplicate values, click the **Duplicate Values** tab. Click **New** to display the Duplicate Values window. In the **Available Columns** list, select CHKING_TRANS_ID, then click the arrow to move that column name into the **Selected Columns** list.

7 To specify that the job is to be aborted if a duplicate transaction ID is found, select **`Abort Job`** in the **`Action if duplicates`** field.

Display 10.21 The Duplicate Values Window



8 In the properties window, click **`OK`** to store your input and close the window.

The transformation is now fully configured and the job is ready to run. The next step is to submit the job and check the SAS log.

Run the Job and Check the Log

After the metadata for a job is complete, you submit the job for execution in order to load data into the target.

1 With the job displayed in the Process Designer window, select **Process ▶ Submit** from the menu bar. SAS Data Integration Studio generates code for the job and submits the code to a SAS application server. The server executes the code.

2 If a pop-up error message appears, or if you want to look at the log for the completed job, click the **`Log`** tab in the Process Designer window.

3 On the **`Log`** tab, scroll through the SAS log information that was generated during the execution of the job. The code that was executed for the job is available in the **`Source Code`** tab.

4 If you find errors in the source code, correct the properties windows of the affected transformations.

5 Correct the metadata and resubmit the job until there are no more errors.

6 After the job runs without error, save the job. Select **File ▶ Save** from the menu bar.

The next task is to verify that the job created the correct output.

Verify Job Outputs

After the job runs without error and has been saved, open the target table CHECKING_TRANS_VALID and the error table CHECKING_TRANS_ERROR to check the results.

1 To view the data in CHECKING_TRANS_VALID, right-click the table in the Project tree or in the Process Designer and select **`View Data`**.

2 To compare the target to the source, repeat the previous step for the source table CHECKING_TRANS. As shown in the following display, three missing source values have received the value ?? in the target in place of the former missing values.

Display 10.22 Contents of the Target Table CHECKING_TRANS_VALID

View Data: CHECKING_TRANS_VALID (rows 1-100) of 597

Starting row: 1 Go to row

#	CHKING_TRANS_ID	CHKING_TRANS_CHANNEL_CD	CHKING_TRA
1	CHK-T-10	??	RS
2	CHK-T-101	??	EF
3	CHK-T-103	??	NJ
4	CHK-T-104	FJ	WG
5	CHK-T-105	WK	OP
6	CHK-T-106	VX	WN
7	CHK-T-107	XO	ZT
8	CHK-T-108	LD	YU

3 The source table contains three invalid values. These values were correctly identified as invalid because they did not appear in the lookup table CHECKING_TRANS_LOOKUP. The following display confirms that three invalid values were written into the error table. A careful look at the target table in the preceding display shows that the source rows were not written into the target. These errors are also noted in the log.

Display 10.23 Contents of CHECKING_TRANS_ERROR

View Data: CHECKING_TRANS_ERROR (rows 1-3) of 3

Starting row: 1 Go to row

#	CHKING_TRANS_ID	CHKING_TRANS_CHAN...	CHKING_TRANS_METHOD_CI
1	CHK-T-1	HE	AF
2	CHK-T-100	QI	DZ
3	CHK-T-102	OU	GW

4 The data validation job was configured to abort if a duplicate value was found in the CHKING_TRANS_ID column. The source contained no duplicate values in that column, so the job ran to completion.

Example: Using a Generated Transformation in a Job

Preparation

Suppose that you need to perform a special task with a data set, and none of the standard transformations in the SAS Data Integration Studio Process Library support this task. An administrator can use the Transformation Generator wizard to create a custom transformation and make it available in the Process Library. The generated transformation can then be used in any SAS Data Integration Studio job.

For example, suppose that you needed to create a report that displays hitting statistics for baseball teams. The following display shows the kind of output that is desired.

Display 10.24 Tigers Hitting Statistics 2002 Report

```
                Tigers Hitting Statistics 2002                          5
                                              12:44 Tuesday, September 27, 2005

 Obs    Name                 G      AB      HR     RBI

  1     Smithy Jones       158     548      26     100
  2     Gary Troy          135     492      25      84
  3     Rafael Fernando    154     636       8      47
  4     Andy Hitfield      154     560      35      94
  5     Vinny Toredo       143     543      12      61
                                            ===
                                            106
```

An administrator could use the Transformation Generator wizard to create a transformation that reads an input table with a certain column structure, enables you to specify certain options, calculates hitting statistics, and displays the result to the **Output** tab in the Process Designer window. SAS Data Integration Studio users could then use this transformation in any job.

Assume that the following preparations have been made:

- An administrator has created the generated transformation (PrintHittingStats), and this transformation is available in the Process Library. For more information about creating generated transformations, see “Maintaining Generated Transformations” on page 75.
- The input to the PrintHittingStats transformation is a table that contains batting statistics for a baseball team. The columns in the source table are assumed to be similar to the columns shown in the following display.

Display 10.25 Contents of TigersHitting2002 Table

View Data: TIGERSHITTING2002 (rows 1-5) of 5

Starting row: 1 Go to row

#	Name	G	AB	R	H	HR	TB	RBI	BB	SO
1	Smithy Jones	158	548	90	179	26	294	100	107	89
2	Gary Troy	135	492	82	151	25	252	84	72	53
3	Rafael Fernando	154	636	95	175	8	246	47	43	114
4	Andy Hitfield	154	560	91	148	35	287	94	83	135
5	Vinny Toredo	143	543	56	126	12	189	61	22	69

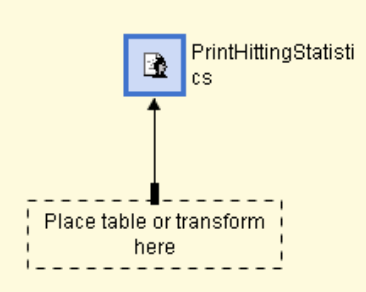
- Metadata for the source table, a SAS data set called TigersHitting2002, is available in a current metadata repository.
- Output for the report will be sent to the **Output** tab in the Process Designer window. The appropriate option must be set so that the **Output** tab appears in the Process Designer window. For details, see "Process Designer Window" on page 20.
- The main metadata repository is under change-management control. You do not have to check out the metadata for the TigersHitting2002 table. (The metadata for this table will not be updated.)
- You have selected a default SAS application server for SAS Data Integration Studio, as described in "Selecting a Default SAS Application Server" on page 96. This server can access the TigersHitting2002 table that are used in the job.
- It is assumed that you have started SAS Data Integration Studio and have opened the appropriate metadata profile. For this example, the appropriate metadata profile would specify the repository that will enable you to access the PrintHittingStatistics transformation and the metadata for the source table, which is TigersHitting2002.

Create and Populate the New Job

Follow these steps to create a complete process flow diagram, from sources, through transformations, to targets:

1. From the SAS Data Integration Studio menu bar, select **Tools ▶ Process Designer**. The New Job wizard displays.
2. Enter *PrintHittingStats Job* in the **Name** field. Then enter a description for the job in the **Description** field.
3. Click **Finish**. An empty job will open in the Process Designer window. The job has now been created and is ready to be populated with the PrintHittingStatistics transformation template and the source table, TigersHitting2002.
4. On the SAS Data Integration Studio desktop, click the **Process** tab to display the Process Library.
5. In the Process Library, open the **UserDefined** folder and the **Reports** subfolder.
6. Click, hold, and drag the **PrintHittingStatistics** transformation into the empty Process Designer window. Release the mouse button to display the template in the Process Designer window for the new job, as shown in the following display.

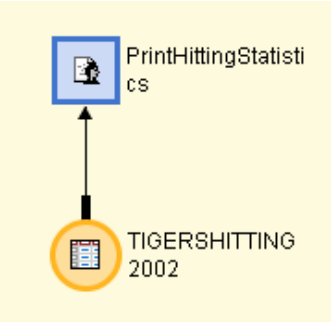
Display 10.26 PrintHittingStatistics Template, Unpopulated



7. On the desktop, click the **Inventory** tab to display the Inventory tree.
8. In the Inventory tree, click and drag the **TigersHitting2002** table into the drop zone (dashed-line box) in the Process Designer window, then release the mouse

button. The **`TigersHitting2002`** table appears as a source in the new job, as shown in the following display.

Display 10.27 PrintHittingStatistics Template, Populated



The job now contains a complete process flow diagram, from the source through the transformation. No target is required in the process flow diagram because output for the job will be sent to the **`Output`** tab in the Process Designer window.

The next task is to specify options in the PrintHittingStatistics transformation.

Configure the PrintHittingStatistics Transformation

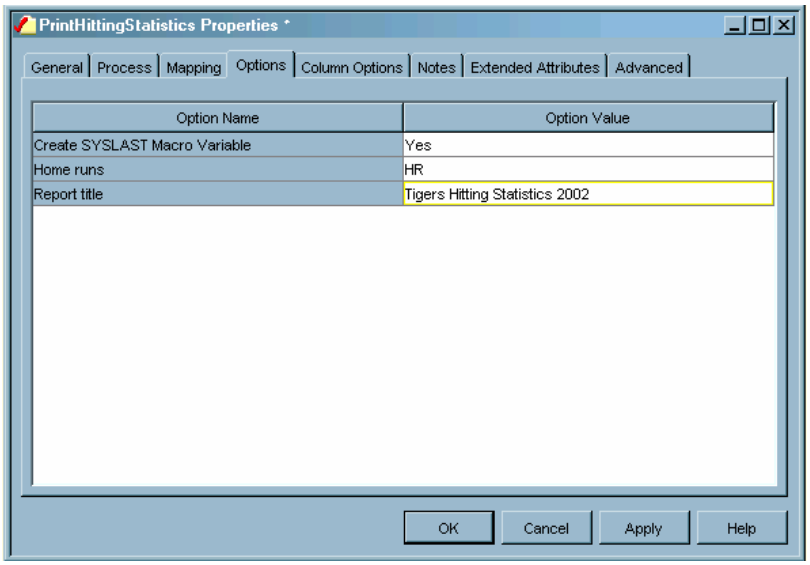
The example job now contains a complete process flow diagram. The job is not ready to run, however. In order to produce the report that is shown in Display 10.24 on page 174, a title must be specified, and a set of columns must be selected from the source. Then, the sum of the values in the **`HR`** column must be calculated. Assume that the steps for doing these tasks have been documented by the person who created the PrintHittingStatistics transformation.

Follow these steps to specify options for the PrintHittingStatistics transformation:

1 In the Process Designer window, select the **`PrintHittingStatistics`** transformation. Then select **File ▶ Properties** from the menu bar. A properties window displays.

2 Click the **Options** tab. The default options for the **PrintHittingStatistics** transformation are shown in the following display.

Display 10.28 Transformation Options Window



Using these settings, the following %LET statements are generated when you run the job:

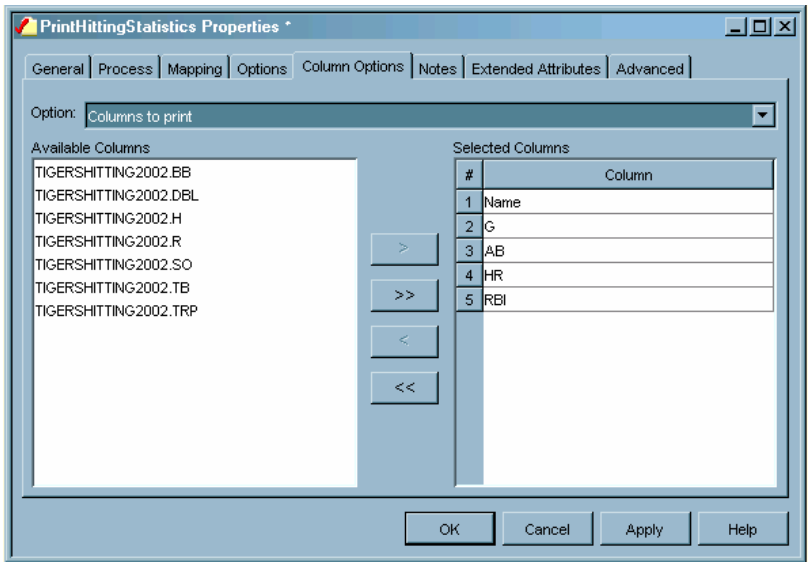
```
%let HomeRuns = %nrquote(HR);
%let ReportTitle = %nrquote(2002 Power Hitting Statistics);
```

3 In the **Home runs** field, enter the name of the source table column that contains home run values. In Display 10.25 on page 174, this is the **HR** column.

4 In the **Report title** field, enter a name for the report, such as **Tigers Hitting Statistics 2002**.

5 Click the **Column Options** tab. Use this tab to select columns from the source table that should appear in the report. For the report that is shown in Display 10.24 on page 174, select the columns **Name**, **G**, **AB**, **HR**, and **RBI**. When you are finished, the **Column Options** tab should resemble the following display.

Display 10.29 Column Options Window



Using these settings, the following %LET statement is generated when you run the job:

```
%let ColumnsToPrint = "Name"n "G"n "AB"n "HR"n "RBI"n;
```

6 When you are finished entering metadata, click **OK** to save your changes.

The job is now ready to run.

Run the Job and Check the Log

After the metadata for a job is complete, you must submit the job for execution in order to create targets on the file system.

1 With the job displayed in the Process Designer window, select **Process ▸ Submit** from the menu bar. SAS Data Integration Studio generates code for the job and submits the code to a SAS application server.

2 If a pop-up error message appears, or if you want to look at the log for the completed job, click the **Log** tab in the Process Designer window.

3 On the **Log** tab, scroll through the SAS log information that was generated during the execution of the job, as shown in the following display.

Display 10.30 Log Tab with Text from the PrintHittingStats Job

```
570        /* Options  */
571        %let HomeRuns = %nrquote(HR);
572        %let ReportTitle = %nrquote(Tigers Hitting Statistics 2002);
573        %let ColumnsToPrint = Name G AB HR RBI;
574
575        PROC PRINT DATA = &SYSLAST;
576        SUM &HomeRuns;
577        VAR &ColumnsToPrint;
578        Title "&ReportTitle";
579        RUN;
```

The code that was executed for the job is available in the **`Source Editor`** tab in the Process Designer window.

4 If you find errors in the source code for a step, select the corresponding transformation in the process flow diagram. Then select **File ▶ Properties** from the menu bar. A properties window displays.

5 Correct the metadata and resubmit the job until there are no more errors.

6 After the job runs without error, save the job. Select **File ▶ Save** from the menu bar.

The next task is to verify that the job created the correct output.

Verify Job Outputs

After the job runs without error and has been saved, you should confirm that the target table(s) contain the data you need, in the format that best communicates the purpose of the targets. In the current example, the output is sent to the **`Output`** tab in the Process Designer window. When you click that tab, a report similar to the one shown in Display 10.24 on page 174 should display.

If the report needs to be improved, change the properties of the transformation that feeds data to the report. If the outputs are correct, you can check in the job.

Check In the Metadata

Check in all objects in the Project tree as follows:

1 In the Project tree, select the repository icon.

2 From the SAS Data Integration Studio menu bar, select **Project ▶ Check In Repository**. All of the objects in the project repository are checked into the change-managed repository.

CHAPTER

11

Optimizing Process Flows

Building Efficient Process Flows

Introduction to Building Efficient Process Flows

Building efficient processes to extract data from operational systems, transform it, and load it into the star schema data model is critical to the success of your process flows. Efficiency takes on greater importance as data volumes and complexity increase. This section describes some simple techniques that can be applied to your processes to improve their performance.

Choosing Between Views or Physical Tables

In general, each step in a process flow creates an output table that becomes the input for the next step in the flow. Consider what format would be best for transferring data between steps in the flow. There are two choices:

- write the output for a step to disk (in the form of SAS data files or RDBMS tables)
- create views that process input and pass the output directly to the next step, with the intent of bypassing some writes to disk

SAS supports two kinds of views, SQL views and DATA Step views, and the two types of views can behave differently. Switching from views to physical tables or tables to views sometimes makes little difference in a process flow. At other times, improvements can be significant. The following tips are useful:

- If the data that is defined by a view is only referenced once in a process flow, then a view is usually appropriate.
- If the data that is defined by a view is referenced multiple times in a process flow, then putting the data into a physical table will likely improve overall performance. As a view, SAS must execute the underlying code repeatedly, each time the view is accessed.
- If the view is referenced once in an process flow, but the reference is a resource-intensive procedure that performs multiple passes of the input, then consider using a physical table.
- If the view is SQL and is referenced once, but the reference is another SQL view, then consider using a physical table. SAS SQL optimization can be less effective when views are nested. This is especially true if the steps involve joins or RDBMS sources.
- If the view is SQL and involves a multi-way join, it is subject to performance limitations and disk space considerations.

Assess the overall impact to your process flow if you make changes based on these tips. In some circumstances, you might find that you have to sacrifice performance in order to conserve disk space.

Some of the standard transformations provided with SAS Data Integration Studio have a **`Create View`** option on their **`Options`** tabs, or a check box that serves the same purpose. Some of the transformations that enable you to specify a view format or a physical table format for their temporary output tables include the following:

- Append
- Data Validation
- Extract

- Library Contents
- Lookup
- SQL Join

Use the appropriate control in the interface to make the switch, and test the process.

Cleansing and Validating Data

Clean and deduplicate the incoming data early in the process flow so that extra data that might cause downstream errors in the flow is caught and eliminated quickly. This process can reduce the volume of data that is being sent through the process flow.

To clean the data, consider using the Sort transformation with the NODUPKEY option and/or the Data Validation transformation. The Data Validation transformation can perform missing-value detection and invalid-value validation in a single pass of the data. It is important to eliminate extra passes over the data, so try to code all of these validations into a single transformation. The Data Validation transformation also provides deduplication capabilities and error-condition handling. See "Example: Creating a Data Validation Job" on page 167. See also "Create Match Code and Apply Lookup Standardization Transformations" on page 105.

Managing Columns

Drop Columns That Are Not Needed

As soon as the data comes in from a source, consider dropping any columns that are not required for subsequent transformations in the flow. Drop columns and make aggregations early in the process flow instead of late so that extraneous detail data is not being carried along between all transformations in the flow. The goal is to create a structure that matches the ultimate target table structure as closely as possible, early in an process flow, so that extra data is not being carried along.

To drop columns in the output table for a SAS Data Integration Studio transformation, click the **Mapping** tab and remove the extra columns from the **Target table** area on the tab. Use derived mappings to create expressions to map several columns together. You can also turn off automatic mapping for a transformation by right-clicking the transformation in the process flow, then deselecting the **Automap** option in the popup menu. You can then build your own transformation output table columns to match your ultimate target table and map.

Do Not Add Unneeded Columns

As data is passed from step to step in an process flow, columns could be added or modified. For example, column names, lengths, or formats might be added or changed. In SAS Data Integration Studio, these modifications to a table, which are done on a transformation's **Mapping** tab, often result in the generation of an intermediate SQL view step. In many situations, that intermediate step adds processing time. Try to avoid generating more of these steps than is necessary.

Accordingly, instead of doing column modifications or additions throughout many transformations in an process flow, rework your flow so that these activities are consolidated within fewer transformations. Avoid using unnecessary aliases; if the mapping between columns is one-to-one, then keep the same column names. Avoid multiple mappings on the same column, such as converting a column from a numeric to

a character value in one transformation and then converting it back from a character to a numeric value in another transformation. For aggregation steps, do any column renaming within those transformations, rather than in subsequent transformations.

Aggregate Columns for Efficiency

When you add column mappings, also consider the level of detail that is being retained. Ask these questions:

- Is the data being processed at the right level of detail?
- Can the data be aggregated in some way?

Aggregations and summarizations eliminate redundant information and reduce the number of records that have to be retained, processed, and loaded into a data collection.

Match the Size of Column Variables to Data Length

Verify that the size of the column variables in the data collection is appropriate to the data length. Consider both the current and future uses of the data:

- Are the keys the right length for the current data?
- Will the keys accommodate future growth?
- Are the data sizes on other variables correct?
- Do the data sizes need to be increased or decreased?

Data volumes multiply quickly, so ensure that the variables that are being stored in the data warehouse are the right size for the data.

Managing Disk Space Use for Intermediate Files

Deleting Intermediate Files at the End of Processing

As described in “How Are Intermediate Files Deleted?” on page 8, intermediate files are usually deleted after they have served their purpose. However, it is possible that some intermediate files might be retained longer than desired in a particular process flow. For example, some user-written transformations might not delete the temporary files that they create.

The following is a post-processing macro that can be incorporated into an process flow. It uses the DATASETS procedure to delete all data sets in the Work library, including any intermediate files that have been saved to the Work library.

```
%macro clear_work;
 %local work_members;
 proc sql noprint;
 select memname
 into :work_members separated by ","
 from dictionary.tables
 where
  libname = "WORK" and
  memtype = "DATA";
 quit;
 data _null_;
  work_members = symget("work_members");
  num_members = input(symget("sqlobs"), best.);
```

```
 do n = 1 to num_members;
  this_member = scan(work_members, n, ",");
  call symput("member"||trim(left(put(n,best.))),trim(this_member));
 end;
 call symput("num_members", trim(left(put(num_members,best.))));
run;
%if #_members gt 0 %then %do;
 proc datasets library = work nolist;
  %do n=1 %to #_members;
   delete &&member&n
  %end;
 quit;
%end;
%mend clear_work;
%clear_work
```

Note: The previous macro deletes all data sets in the Work library. △

For details about adding a post process to a SAS Data Integration Studio job, see "Adding SAS Code to the Pre and Post Processing Tab" on page 225.

Deleting Intermediate Files at the End of Processing

The transformation output tables for an process flow remain until the SAS session that is associated with the flow is terminated. Analyze the process flow and determine whether there are output tables that are not being used (especially if these tables are large). If so, you can add transformations to the flow that will delete these output tables and free up valuable disk space and memory. For example, you could add a generated transformation that would delete output tables at a certain point in the flow. For details about generated transformations, see "Adding a Generated Transformation to the Process Library" on page 228.

Minimizing Remote Data Access

Avoid or minimize remote data access in a process flow. For more information about remote data access, see "Supporting Multi-Tier (N-Tier) Environments" on page 64.

Setting Options for Table Loads

SAS Data Integration Studio provides several different transformations for loading output tables in a process flow, as shown in the following table.

Table 11.1 Loader Transformations

Table Loader	reads a source table and writes to a target table. This transformation is added automatically to a process flow when a table is specified as a source or a target.
SCD Type 2 Loader	loads source data into a dimension table, detects changes between source and target rows, updates change tracking columns, and applies generated key values. This transformation implements slowly changing dimensions.
SPD Server Table Loader	reads a source and writes to an SPD Server target. This transformation is automatically added to a process flow when an SPD Server table is specified as a source or as a target. Enables you to specify options that are specific to SPD Server tables.

In some cases, you can improve the performance of a job by specifying certain options for one or more loader transformations in the job. In other cases, you must use other methods to improve load performance.

In general, when you are reloading more than 10% of the data in an existing table, you'll get better performance if you drop and re-create the indexes after the load. To enable this option for the Loader transformation, right-click the Loader transformation in the job and select **`Properties`** from the pop-up menu. Click the **`Load Technique`** tab, then select the **`Drop Indexes`** option. The default load technique for RDBMS tables in SAS Data Integration Studio is **`Truncate`**, and that option should be acceptable for most RDBMS data loads. When the **`Drop`** load technique is specified for a Loader transformation, consider whether it is acceptable to lose data constraints. SAS Data Integration Studio will rebuild some constraints, notably indexes, but others, such as keys, will not be rebuilt. Also, not all user IDs have the required privilege to re-create tables in a database.

Consider bulk loading the data into database tables, by using the optimized SAS/ACCESS engine bulk loaders. Bulk load options are set in the metadata for a RDBMS library. To set these options from the SAS Data Integration Studio Inventory tree, right-click an RDBMS library, then select **Properties ▶ Options ▶ Advanced Options ▶ Output**. Select the check box that will enable the RDBMS bulk load facility for the current library.

You can set additional bulk load options for the tables in an RDBMS library. To set these options from the SAS Data Integration Studio Inventory tree, right-click an RDBMS table, then select **Properties ▶ Physical Storage ▶ Table Options**. Specify the appropriate bulk load option for the table.

Also, consider using native SAS/ACCESS engine libraries instead of ODBC libraries or OLEDB libraries for RDBMS data.

Using Transformations for Star Schemas and Lookups

Consider using the Lookup transformation when building process flows that require lookups such as fact table loads. The Lookup transformation is built using a fast in-memory lookup technique known as DATA step hashing that is available in SAS 9. The transformation allows for multi-column keys and has useful error handling techniques such as control over missing-value handling and the ability to set limits on errors.

When you are working with star schemas, consider using the SCD Type 2 transformation. This transformation efficiently handles change data detection, and has been optimized for performance. Several change detection techniques are supported: date-based, current indicator, and version number. For details about the SCD Type 2 transformation, see Chapter 12, "Using Slowly Changing Dimensions," on page 195.

Using Surrogate Keys

Another technique to consider when you are building the data warehouse is to use incrementing integer surrogate keys as the main key technique in your data structures. Surrogate keys are values that are assigned sequentially as needed to populate a dimension. They are very useful because they can shield users from changes in the operational systems that might invalidate the data in a warehouse (and thereby require redesign and reloading). Using a surrogate key can avoid issues if, for example, the operational system changes its key length or type. In this case, the surrogate key remains valid, where an operational key would not.

The SCD Type 2 transformation includes a surrogate key generator. You can also plug in your own methodology that matches your business environment to generate the keys and point the transformation to it. There is also a Surrogate Key Generator transformation that can be used to build incrementing integer surrogate keys.

Avoid character-based surrogate keys. In general, functions that are based on integer keys are more efficient because they avoid the need for subsetting or string partitioning that might be required for character-based keys. Numeric strings are also smaller in size than character strings, thereby reducing the storage required in the warehouse.

For details about surrogate keys and the SCD Type 2 transformation, see "SCD and SAS Data Integration Studio" on page 198.

Working from Simple to Complex

When you build process flows, build complexity up rather than starting at a complex task. For example, consider building multiple individual jobs and validating each rather than building large, complex jobs. This will ensure that the simpler logic produces the expected results.

Also, consider subsetting incoming data or setting a pre-process option to limit the number of observations that are initially being processed in order to fix job errors and validate results before applying processes to large volumes of data or complex tasks. For details about limiting input to SAS Data Integration Studio jobs and transformations, see "Verifying a Transformation's Output" on page 188.

Analyzing Process Flow Performance

Introduction to Analyzing Process Flow Performance

Occasionally a process flow might run longer than you expect, or the data that is produced might not be what you anticipate (either too many records or too few). In such cases, it is important to understand how a process flow works, so that you can correct errors in the flow or improve its performance.

A first step in analyzing process flows is being able to access information from SAS that will explain what happened during the run. If there were errors, you need to

understand what happened before the errors occurred. If you are having performance issues, then the logs will explain where you are spending your time. Finally, if you know what SAS options are set and how they are set, this can help you determine what is going on in your process flows. The next step in analyzing process flows is interpreting the information that you have obtained.

This section describes how to do the following tasks:

- use simple debugging techniques
- use the SAS log to gather information
- analyze the log
- determine option settings
- specify status codes for jobs and transformations
- add custom debugging code to a process flow
- save the temporary output tables after the process flow has finished so that you can review what is being created

Simple Debugging Techniques

Monitoring Job Status

See “Monitoring the Status of Jobs” on page 103.

Verifying a Transformation’s Output

If a job is not producing the expected output, or if you suspect that something is wrong with a particular transformation, you can view the output tables for the transformations in the job in order to verify that each transformation is creating the expected output. See “Analyzing Transformation Output Tables” on page 192.

Limiting a Transformation’s Input

When you are debugging and working with large data files, you might find it useful to decrease some or all of the data that is flowing into a particular step or steps. One way of doing this is to use the OBS= data set option on input tables of data steps and procedures.

To specify the OBS= for an entire job in SAS Data Integration Studio, add the following code to the **Pre and Post Processing** tab in the job’s property window:

```
options obs=<number>;
```

For an example of this method, see “(Optional) Reduce the Amount of Data Processed by the Job” on page 153.

To specify the OBS= for a transformation within a job, you can temporarily add the option to the system options field on the **Options** tab in the transformation’s property window. Alternatively, you can edit the code that is generated for the transformation and execute the edited code. For more information about this method, see “Replacing the Generated Code for a Transformation with User-Written Code” on page 226.

Important considerations when you are using the OBS= system option include the following:

- All inputs into all subsequent steps will be limited to the specified number, until the option is reset.
- Setting the number too low prior to a join or merge step can result in few or no matches, depending on the data.

- In the SAS Data Integration Studio Process Editor, this option will stay in effect for all runs of the job until it is reset or the Process Designer window is closed.

The syntax for resetting the option is as follows:

```
options obs=MAX;
```

Note: Removing the OBS= line of code from the Process Editor does not reset the OBS= system option. You must reset it as shown previously, or by closing the Process Designer window. △

Redirecting Large SAS Logs to a File

The SAS log for a job provides critical information about what happened when a job was executed. However, large jobs can create large logs, which can slow down SAS Data Integration Studio considerably. In order to avoid this problem, you can re-direct the SAS log to a permanent file, then turn off the **Log** tab in the Process Designer window. For details, see "Using SAS Logs to Analyze Process Flows" on page 189.

Setting SAS Options for Jobs and Transformations

When you submit a SAS Data Integration Studio job for execution, it is submitted to a SAS Workspace Server component of the relevant SAS application server. The relevant SAS Application Server is one of the following:

- the default server that is specified on the **SAS Server** tab in the Options window
- the SAS Application Server to which a job is deployed with the **Deploy for Scheduling** option

To set SAS invocation options for all SAS Data Integration Studio jobs that are executed by a particular SAS server, specify the options in the configuration files for the relevant SAS Workspace Servers, batch or scheduling servers, and grid servers. (You would not set these options on SAS Metadata Servers or SAS Stored Process Servers.) Examples of these options include UTILLOC, NOWORKINIT, or ETLS_DEBUG. For more information, see "Modifying Configuration Files or SAS Start Commands" on page 224.

To set SAS global options for a particular job, you can add these options to the **Pre and Post Process** tab in the Properties window for a job. For more information, see "Adding SAS Code to the Pre and Post Processing Tab" on page 225.

The property window for most transformations within a job has an **Options** tab with a **System Options** field. Use the **System Options** field to specify options for a particular transformation in a job's process flow. For more information, see "Specifying Options for Transformations" on page 225.

For more information about SAS options, search for relevant phrases such as "system options" and "invoking SAS" in SAS OnlineDoc.

Using SAS Logs to Analyze Process Flows

Introduction to Using SAS Logs to Analyze Process Flows

The errors, warnings, and notes in the SAS log provide information about process flows. However, large SAS logs can decrease performance, so the costs and benefits of large SAS logs should be evaluated. For example, in a production environment, you might not want to create large SAS logs by default.

Evaluating SAS Logs

The SAS logs from your process flows are an excellent resource to help you understand what is happening as the flows execute. For example, when you look at the run times in the log, compare the real-time values to the CPU time (user CPU plus system CPU). For read operations, the real time and CPU time should be close. For write operations, however, the real time could substantially exceed the CPU time, especially in environments that are optimized for read operations. If the real time and the CPU time are not close, and they should be close in your environment, investigate what is causing the difference.

If you suspect that you have a hardware issue, see *A Practical Approach to Solving Performance Problems with the SAS System*, a document that is available from the Scalability and Performance Papers page on support.sas.com (support.sas.com/rnd/scalability/papers/).

If you determine that your hardware is properly configured, then review the SAS code. Transformations generate SAS code. Understanding what this code is doing is very important to insure that you do not duplicate tasks, especially SORTs, which are resource-intensive. The goal is to configure the hardware so that there are no bottlenecks, and to avoid needless I/O in the process flows.

Capturing Additional SAS Options in the SAS Log

To analyze performance, we recommend that you turn on the following SAS options so that detailed information about what the SAS tasks are doing is captured in the SAS log:

```
FULLSTIMER
MSGLEVEL=I (this option prints additional notes pertaining to index, merge
   processing, sort utilities, and CEDA usage, along with the standard notes,
   warnings, and error messages)
SOURCE, SOURCE2
MPRINT
NOTES
```

To interpret the output from the FULLSTIMER option, see *A Practical Approach to Solving Performance Problems with the SAS System*, a document that is available from the Scalability and Performance Papers page on support.sas.com (support.sas.com/rnd/scalability/papers/).

In addition, the following SAS statements will also echo useful information to the SAS log:

```
PROC OPTIONS OPTION=UTILLOC; run;
PROC OPTIONS GROUP=MEMORY; run;
PROC OPTIONS GROUP=PERFORMANCE; run;
LIBNAME _ALL_ LIST;
```

The PROC OPTIONS statement will echo SAS options and their current settings to the SAS log. There are hundreds of SAS options, so if, for example, you prefer to see which value has been set to the SAS MEMORY option, you can issue the PROC OPTIONS statement with the GROUP=MEMORY parameter. The same is true if you want to see only the SAS options that pertain to performance.

The LIBNAME _ALL_ LIST statement will echo to the SAS log information (physical path location, engine being used, etc.) regarding each libref that is currently assigned to the SAS session. This is helpful for understanding where all the work occurs during the process flow. For details about setting SAS invocation options for SAS Data Integration Studio, see “Setting SAS Options for Jobs and Transformations” on page 189.

Redirecting SAS Data Integration Studio's Log to a File

The SAS log for a job provides critical information about what happened when a job was executed. However, large jobs can create large logs, which can slow down SAS Data Integration Studio. In order to avoid this problem, you can re-direct the SAS log to a permanent file, then turn off the **Log** tab in the Process Designer window.

When you install SAS Data Integration Studio, the Configuration Wizard enables you to set up as permanent SAS log files for each job that is executed. The SAS log filenames will contain the name of the job that created the log, plus a timestamp of when the job was executed.

Alternatively, you can add the following code to the **Pre and Post Process** tab in the properties window for a job:

```
proc printto log=...path_to_log_file NEW; run;
```

For details about adding a pre-process to a SAS Data Integration Studio job, see "Adding SAS Code to the Pre and Post Processing Tab" on page 225. The previous code will cause the log to be redirected to the specified file. Be sure to use the appropriate host-specific syntax of the host where your job will be running when you specify this log file, and make sure that you have write access to the location where the log will be written.

Viewing or Hiding the Log in SAS Data Integration Studio

The Process Designer window in SAS Data Integration Studio has a **Log** tab that displays the SAS log for the job in the window. To display or hide the **Log** tab, complete these steps:

1. From the SAS Data Integration Studio menu bar, select **Tools ▶ Options** to display the Options window.
2. On the **General** tab, select or deselect the check box that controls whether the **Log** tab is displayed in the Process Designer window.

Using Status Codes to Analyze Process Flows

When you execute a job in SAS Data Integration Studio, a return code for each transformation in the job is captured in a macro variable. The return code for the job is set according to the least successful transformation in the job. SAS Data Integration Studio enables you to associate a return code condition, such as **Successful**, with an action, such as **Send Email** or **Abort**. In this way, users can specify how a return code is handled for the job or transformation.

For example, you could specify that a transformation in a process flow will abort, based on conditions that you define. This can reduce the log to just the transformations leading up to the problem being investigated, making the log more manageable and eliminating inconsequential error messages. For more information about status code handling, see "Monitoring the Status of Jobs" on page 103.

Adding Debugging Code to a Process Flow

If you are analyzing a SAS Data Integration Studio job, and the information that is provided by logging options and status codes is not enough, consider the following methods for adding debugging code to the process flow.

Table 11.2 Methods for Adding Custom Debugging Code

Method	Documentation
Replace the generated code for a transformation with user-written code.	"Replacing the Generated Code for a Transformation with User-Written Code" on page 226.
Add a User-Written Code transformation to the process flow.	"Adding a User-Written Code Transformation to the Process Flow for a Job" on page 227.
Add a generated transformation to the process flow.	"Adding a Generated Transformation to the Process Library" on page 228.

Custom code can direct information to the log or to alternate destinations such as external files, tables. Possible uses include tests of frequency counts dumping out SAS macro variable settings, or listing the runtime values of system options.

Analyzing Transformation Output Tables

Most transformations in a SAS Data Integration Studio job create at least one output table and store that table in the Work library on the SAS Workspace Server that executes the job. The output table for each transformation becomes the input to the next transformation in the process flow. All output tables are deleted when the job is finished or the current server session ends.

If a job is not producing the expected output, or if you suspect that something is wrong with a particular transformation, you can view the output tables for the transformations in the job to verify that each transformation is creating the expected output. The next sections describe how to view a transformation's output table from the Process Designer window, and how to preserve output tables so that you can view their contents by other means.

Note: In addition to being useful when analyzing process flows, output tables can be preserved to determine how much disk space they require, or to restart an process flow after it has failed at a particular step (transformation). △

Viewing the Output Table for a Transformation

See "View Data in a Transformation's Temporary Output Table" on page 111.

Setting SAS Options to Preserve Intermediate Files for Batch Jobs

When SAS Data Integration Studio jobs are executed in batch mode, a number of SAS options can be used to preserve intermediate files in the Work library. These system options can be set as described in "Setting SAS Options for Jobs and Transformations" on page 189.

Use the NOWORKINIT system option to prevent SAS from erasing existing Work files on invocation. Use the NOWORKTERM system option to prevent SAS from erasing existing Work files on termination.

For example, to create a permanent SAS Work library in UNIX and PC environments, you can start the SAS Workspace Server with the WORK option to redirect the Work files to a permanent Work library. The NOWORKINIT and NOWORKTERM options must be included.

```
C:\>"C:\Program Files\SAS\SAS 9.1\sas.exe"
-work "C:\Documents and Settings\sasapb\My Documents\My SAS Files\My SAS Work Folder"
-noworkinit
-noworkterm
```

This redirects the generated Work files in the folder My SAS Work Folder.

To create a permanent SAS Work library in the z/OS environment, edit your JCL statements and change the WORK DD statement to a permanent MVS data set. For example:

```
 //STEP1 EXEC SDSSAS9,REGION=50M
//* changing work lib definition to a permanent data set
//SDSSAS9.WORK DD DSN=userid.somethin.sasdata,DISP=OLD
//* other file defs
//INFILE DD ... .
```

CAUTION:

If you redirect Work files to a permanent library, you must manually delete these files to avoid running out of disk space. △

Using a Transformation's Property Window to Redirect Output Files

The default name for a transformation's output table is a two-level name that specifies the Work libref and a generated member name, such as work.W54KFYQY. The **Process** tab in the property windows for some SAS Data Integration Studio transformations has a **Data Location** button that enables you to specify the name and location of the output table for that transformation. Note that this location can be a SAS library or RDBMS library. This has the added benefit of providing users the ability to specify which output tables they want to retain and allow the rest to be deleted by default. Users can use this scheme as a methodology for checkpoints by writing specific output tables to disk when needed.

Note: If you want to save a transformation output table to a library other than the SAS User library, replace the default name for the output table with a two-level name. △

If you refer to an output table with a single-level name (for example, *employee*), instead of a two-level name (for example, *work.employee*), SAS automatically sends the output table into the User library, which defaults to the Work library. However, this default behavior can be changed by any SAS user. Through the USER= system option, a SAS user can redirect the User library to a different library. If the USER= system option is set, single-level tables are stored in the User library, which has been redirected to a different library, instead of to the Work library.

Adding a List Data Transformation to the Process Flow

In SAS Data Integration Studio, you can use the List Data transformation to print the contents of an output table from the previous transformation in an process flow. Add the List Data transformation after any transformation whose output table is of interest to you.

The List Data transformation uses the PRINT procedure to produce output. Any options associated with that procedure can be added from the **Options** tab in the transformation's property window. By default, output goes to the **Output** tab in the Process Designer window. Output can also be directed to an HTML file. For large data, customize this transformation to print just a subset of the data. For details, see "Limiting a Transformation's Input" on page 188.

Adding a User Written Code Transformation to the Process Flow

You can add a User Written Code transformation to the end of a process flow that would move or copy some of the data sets in the Work library to a permanent library. For example, assume that there are three tables in the Work library (test1, test2, and test3). The following code moves all three tables from the Work library to a permanent library named PERMLIB and then deletes them from the Work library.

```
libname permlib base
"C:\Documents and Settings\ramich\My Documents\My SAS Files\9.1";
proc copy move
in = work
out = permlib;
select test1 test2 test3;
run;
```

For information about User Written Code transformations, see “Adding a User-Written Code Transformation to the Process Flow for a Job” on page 227.

CHAPTER

12

Using Slowly Changing Dimensions

About Slowly Changing Dimensions

SCD Concepts

A dimension is a category of contextual data or detail data that is implemented in a data model such as a star schema. For example, in a star schema, a dimension named Customers might associate customer data with transaction identifiers and transaction amounts in a fact table.

A dimension table is a table that contains data about a particular dimension in a star schema or a snowflake schema. A primary key connects a dimension table to a related fact table. For example, if a dimension table named Customers has a primary key column named Customer ID, then a fact table named Customer Sales might specify the Customer ID column as a foreign key.

A fact table is the central table in a star schema or snowflake schema. A fact table typically contains numerical measurements or amounts and is supplemented by contextual information in dimension tables. For example, a fact table might include transaction identifiers and transaction amounts. Dimension tables could add contextual information about customers, products, and salespersons. Fact tables are associated with dimension tables via key columns. Foreign key columns in the fact table contain the same values as the primary key columns in the dimension tables.

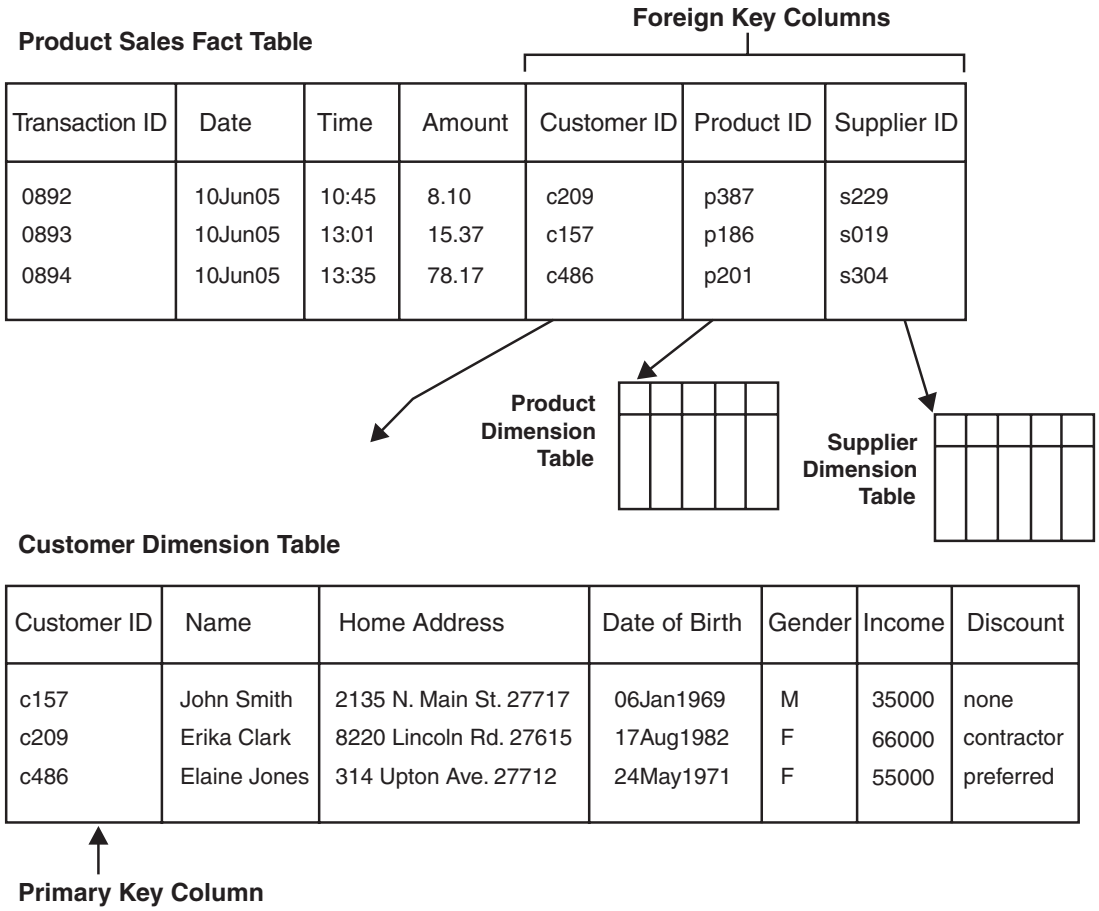
Slowly changing dimensions (SCD) is a technique for tracking changes to dimension table values in order to analyze trends. For example, a dimension table named Customers might have columns for Customer ID, Home Address, Age, and Income. Each time the address or income changes for a customer, a new row could created for that customer in the dimension table, and the old row could be retained. This historical record of changes could be combined with purchasing information to forecast buying trends and direct customer marketing campaigns.

Type 2 SCD Dimensional Model

Dimension tables store attribute values. Dimension tables are combined with fact tables in data structures known as star schemas or snowflakes. In these data structures, fact tables record numeric measures that are associated with events. Dimension tables record categories of attribute values that are associated with the facts.

Key columns associate facts with attributes. For example, consider a star schema that consists of a fact table and several dimension tables. The fact table records product sales. As shown in the following diagram, the numeric measures in the fact table are recorded in columns for amount, date, and time. The primary key column Transaction ID uniquely identifies each row in the fact table. The foreign key columns in the fact table provide referential integrity between the fact table and the dimension tables. The foreign key values enable each fact table row to accurately reference all of the attribute values that are associated with that event.

Figure 12.1 Foreign and Primary Key Columns in a Star Schema



The fact and dimension tables in the preceding diagram represent a valid star schema, but they do not as yet implement slowly changing dimensions. To implement Type 2 slowly changing dimensions, the dimension tables need new columns that track changes to attribute values.

The following diagram shows how the columns Begin Current, End Current, and Customer Generated Key have been added to the Customer Dimension Table. The columns Begin Current and End Current establish a time line of attribute value changes for each customer. The Customer Generated Key column provides unique identifiers for each row in the Customer Dimension table. To maintain referential integrity, the new generated keys are added to the fact table after the dimension table has been loaded.

Figure 12.2 Type 2 SCD Columns in the Customer Dimension Table

Customer Dimension Table

Cust. ID	Name	Home Address	...	Begin Current	End Current
c198	John Smith	2135 N. Main St. 27717	...	17Oct1999	current
c198	John Smith	1108 Lynn Rd. 27513	...	03Jun1994	17Oct1999
c198	John Smith	312 S. Maple St. 27501	...	15May1983	03Jun1994
c199	Erika Clark	8220 Lincoln Rd. 27615	...	29Jul2005	current

The previous diagram shows that customer John Smith has changed his home address three times. This information, coupled with other data in other columns (such as age and frequency of purchase), can be used in analyses. One such analysis could determine if Mr. Smith was a good candidate for a marketing campaign.

SCD and SAS Data Integration Studio

Transformations That Support SCD

SAS Data Integration Studio provides the following transformations that can be used to implement slowly changing dimensions:

Type 2 SCD Loader
: loads dimension tables, detects changes, tracks changes, and generates integer key values. Generated key values give the target a primary key that is not dependent on the business key in the source. For more information, see “About the SCD Type 2 Loader Transformation” on page 199. See also “Example: Using Slowly Changing Dimensions” on page 204.

Lookup
: loads source data into fact tables using key values from dimension tables. When dimension tables are loaded beforehand, and when the dimension tables contain newly generated primary key columns, the Lookup transformation efficiently pulls those generated key columns into the fact table to maintain referential integrity. The lookup process uses the latest hashing techniques for optimal performance. Exception handling enables selective responses to missing values and missing tables. WHERE-clause filtering is available to cleanse lookup table data.

Fact Table Lookup
: loads source data into fact tables using key values from dimension tables, in a manner that is similar to the more recent Lookup transformation. Use Fact Table Lookup instead of Lookup when you want to create and save a lookup table that you can use in subsequent transformations or jobs.

Key Effective Date
: updates dimension tables based on changes to the business key, when change detection is unnecessary.

Surrogate Key Generator
: generates unique key numbers for dimension tables, in a manner that is similar but less feature-rich than the SCD Type 2 Loader transformation. Use the Surrogate Key Generator when key generation is the sole task that is required at that point in the job.

To display Help topics that illustrate how these transformations can be used in SAS Data Integration Studio jobs, follow these steps:

1. From the SAS Data Integration Studio menu bar, select **Help ▸ Contents**. The Help window displays.
2. In the left pane of the Help window, select **Examples ▸ Process Library Examples**. See the examples in the **`Data Transforms`** folder.

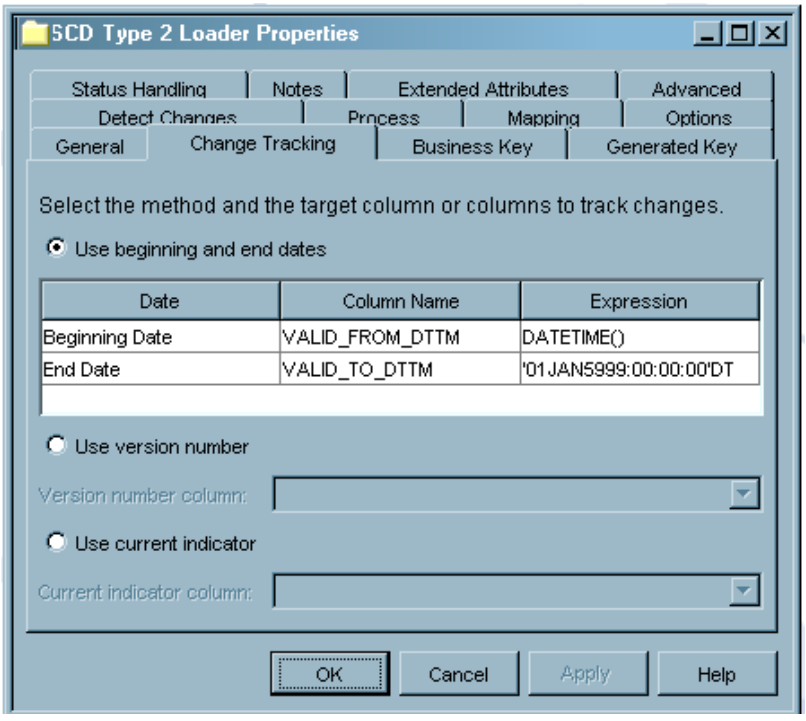
About the SCD Type 2 Loader Transformation

Use the SCD Type 2 Loader transformation to load dimension tables and track changes to dimension table values. The loader supports Type 2 slowly changing dimensions. That is, the loader populates special columns in a dimension table that indicate whether a record is current or historical.

Change Tracking Techniques

The SCD Type 2 Loader supports three different techniques for indicating current and historical data: effective date range, current indicator, or version number. The following display shows the **`Change Tracking`** tab in the properties window for the SCD Type 2 Loader. This tab enables you to select the technique for tracking changes.

Display 12.1 Change Tracking Tab



In the previous display, the technique **`Use beginning and end dates`** has been selected. The Beginning Date is the current date, and the End Date is some point in

the future. As more data points are added, the End Date can be changed to a termination date, and a new record for the current value can be entered.

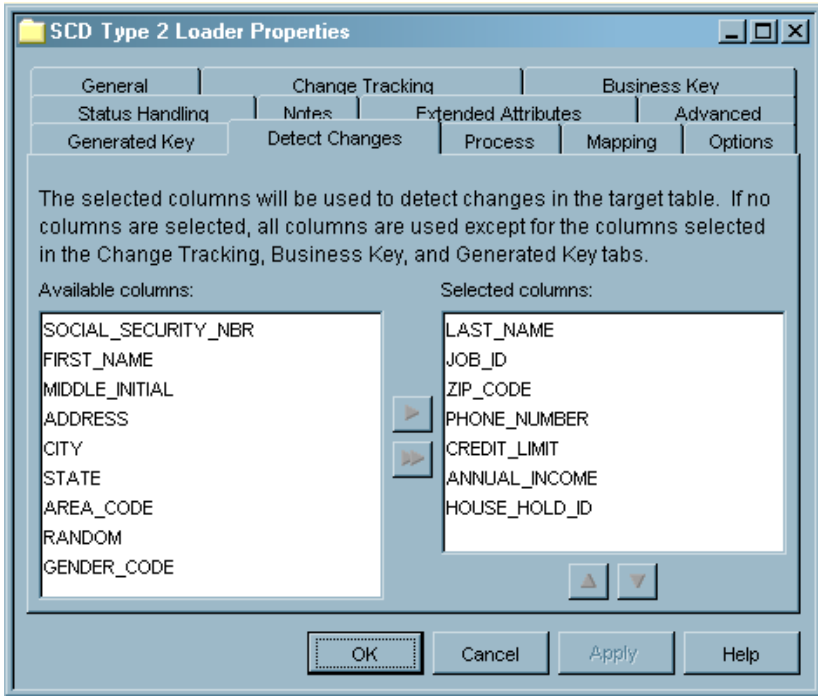
It is good practice to name the change tracking columns consistently across dimensions in the data model so that they are easily recognizable. In the previous display, the column names VALID_FROM_DTTM and VALID_TO_DTTM are used.

After a time/date range has been specified, you can find the desired value when querying the dimension table. Instead of just looking for a specific value in a column of data, you have the ability to perform a query for current and/or historical data. For example, a query could be performed to return the latest address for a household. Because the table has stored date information (in the Beginning Date and End Date), you can determine the value for this or any other data.

Selecting Columns for Change Detection

To optimize the loading of the dimension table, change detection should be enabled only for those columns that are relevant to the analysis that you will perform. For example, the following display shows the **`Detect Changes`** tab in the properties window for the SCD Type 2 Loader.

Display 12.2 Detect Changes Tab



In the previous display, change detection will be performed only for the dimension table columns in the **`Selected columns`** pane on the right. The columns that remain in the **`Available columns`** pane on the left are apparently not relevant to the analysis that will be performed on the dimension table.

The actual change detection is done by a checksum-like algorithm that computes a distinct value based on all the selected input columns. These checksum-like values are compared to detect change. This methodology is fast because it does not require large text-based or column by column compares to detect changes against each row, which quickly gets prohibitively expensive as data volumes rise.

It is possible to persist the checksum table to a permanent data set in the **`Options`** tab of the SCD Type 2 Loader. This saves the step of having to re-create the checksum

table each time incoming records are received for the dimension table, and this can improve performance.

Generating Surrogate Keys

Source data that has been captured from operational systems might not have unique identifiers across all sources. For example, in a collection of household census data, each state might have different ways of recording the household serial number: one state might use character variables and another might use numeric values as the identifier. However, each entry indicates a different household, and therefore each entry needs to have a unique value assigned to it when captured into a dimension table.

The process of creating a complete set of unique identifiers is called *generating a surrogate key*. Typically, the primary keys that are stored in a dimension table are the unique set of surrogate keys assigned to each entry in the dimension table. The original key value from the operational system is also stored in the dimensional table and serves as the business key, which can be used to identify unique incoming records from the operational system.

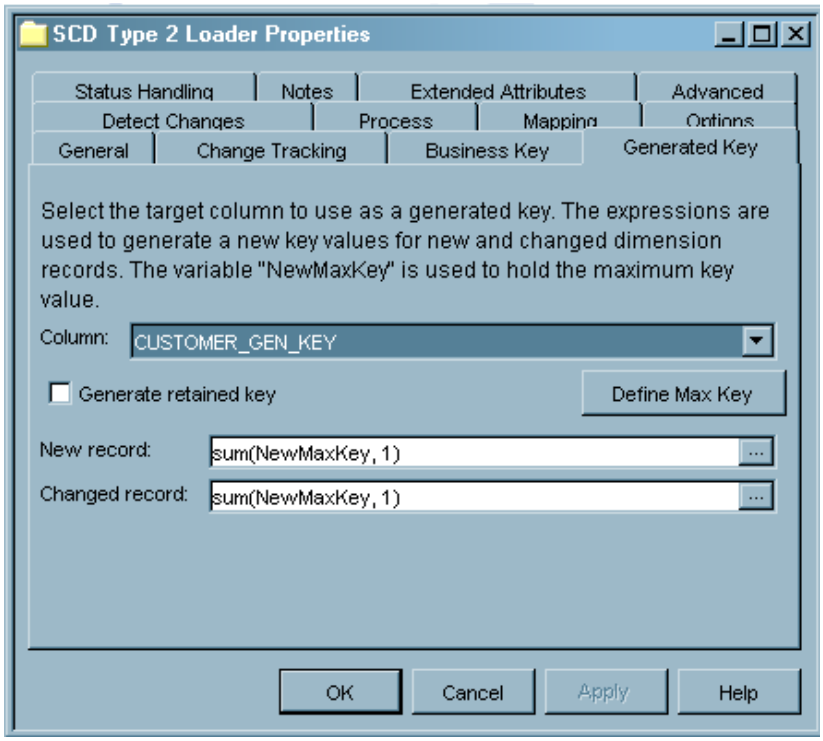
Surrogate keys are very useful because they can shield users from changes in the operational system that might invalidate the data in a data warehouse, thereby requiring redesign and reloading. Surrogate keys can help you avoid problems such as the operational system changing its key length or type. In this case the surrogate key remains valid, where an operational key would not.

It is best to avoid character-based surrogate keys. In general, functions that are based on integer keys are more efficient because they avoid the need for subsetting or string portioning that might be required for character keys. Numeric values are also smaller in size than character strings, which helps reduce the size of the field needed to store the key, and they are generally immune to length changes which would necessitate rebuilding the warehouse.

Several transformations in SAS Data Integration Studio enable you to generate surrogate keys. The SCD Type 2 Loader enables you to combine table loading and surrogate key generation in one transformation.

The following display shows the **`Generated Key`** tab in the properties window for the SCD Type 2 Loader. This tab enables you to set up surrogate key generation.

Display 12.3 Generated Key Tab



The SCD Type 2 Loader can generate the key using a simple increment algorithm, or you can provide your own key-generation algorithm to the transformation.

How Source Data Is Loaded a Dimension Table

Source data is loaded into a dimension table as follows:

- *Update the cross-reference table.*The SCD Type 2 Loader maintains a cross-reference table that is used to detect data changes. The cross-reference table is loaded with a subset of data that is copied from the dimension table. The data in the cross-reference table consists of the current (or "most recent") rows for each business key value. The business key is the primary key in the original source data.

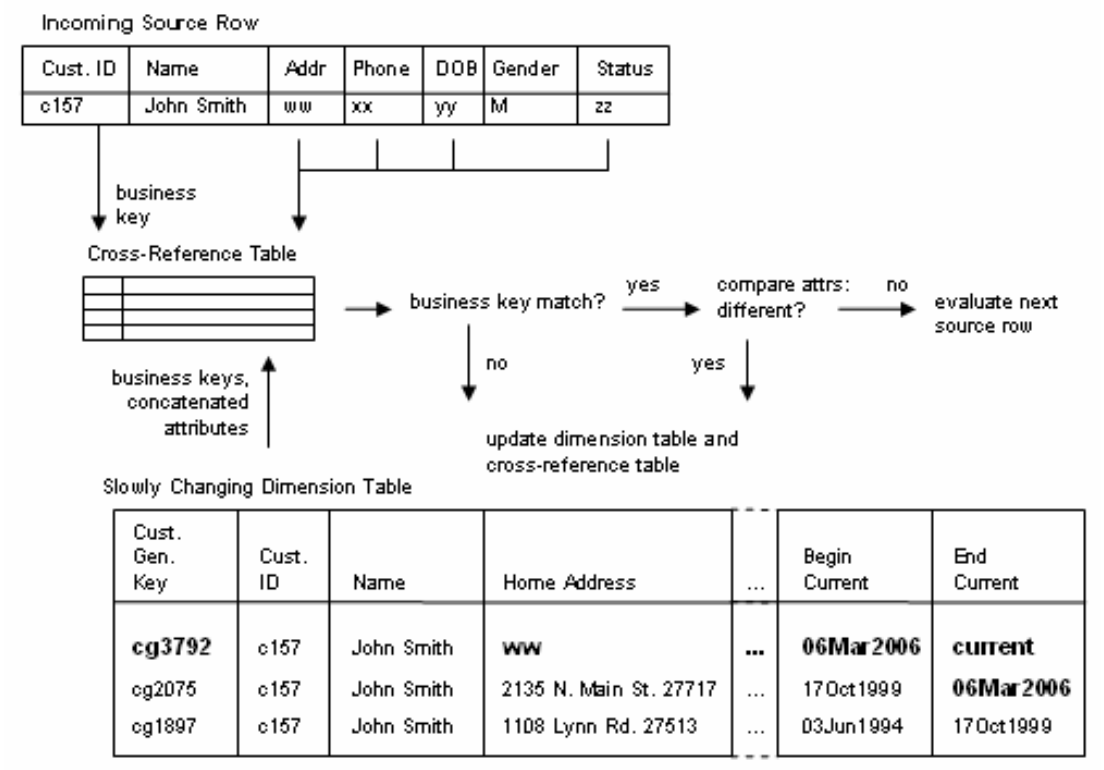
  Columns in the cross-reference table consist of the business key and a second column that contains concatenated attribute values. The concatenated attribute values are used to detect data changes between incoming source rows and the current rows in the dimension table.

  Note: The cross-reference table can be stored as a permanent table to improve performance. When loading the fact table, further performance improvement is gained by using the cross-reference table, rather than the dimension table, to load new generated keys.

- *Detect a new business key.* If the business key value in a source row is not found in the cross-reference table, add the source row to the dimension table. Add data to the dimension table as specified to complete the change tracking and generated key columns. Update the cross-reference table and evaluate to the next source row.

- *Detect an existing business key and changed data.* If an incoming source row contains a business key value that exists in the cross–reference table, the SCD Type 2 Loader compares attribute values between the source row and the cross-reference table. If the source data differs from the concatenated data, write the source row into the dimension table as the new current row for that business key. Add data as specified to complete the change tracking and generated key columns. Update the cross-reference table and evaluate to the next source row.

Figure 12.3 Change Detection, Change Tracking, and Key Generation for Slowly Changing Dimensions



The preceding diagram shows how the Begin Current and End Current columns are updated. The new current row receives a date/time value in Begin Current, and End Current receives a place-holder value. In the former current row, the place-holder value in the End Current column is replaced by a date/time value. The date/time values can be supplied in the source data, or they can be generated by the SCD Type 2 Loader using expressions.

- *Detect an existing business key with no data changes.* If the business key in a source row matches a business key in the cross-reference table, and if the attribute values in the source row match the attribute values in the cross-reference table, do not write the source row into the dimension table. Move to the next source row.

Example: Using Slowly Changing Dimensions

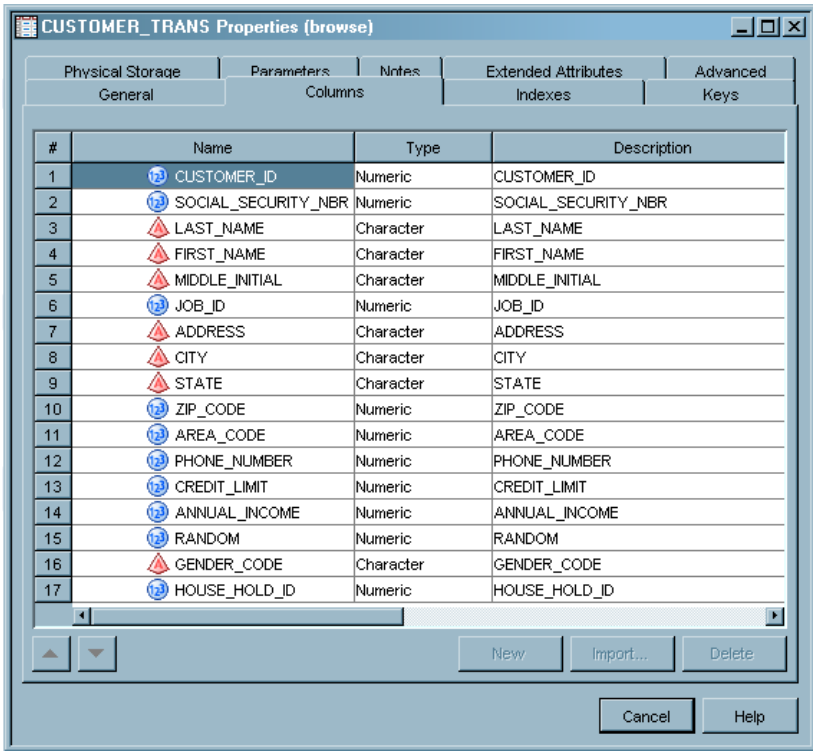
Preparation

This example illustrates how to use the SCD Type 2 Loader to load a customer dimension table in a star schema. Each time that the values in certain columns change for a customer, the old row of values are preserved for that customer, as well as the new row. Each row contains date/time information that establishes the beginning and end of the time when each row was (or is) the most current row for that dimension. The date/time information in the dimension table supports trend analysis and forecasting. The loader transformation is configured for change tracking, change detection, and surrogate key generation.

Assume the following about the process flow in the example.

- The source table for the customer dimension table is a SAS data set named CUSTOMER_TRANS.
- CUSTOMER_TRANS has the columns that are shown in the following display.

Display 12.4 Columns in CUSTOMER_TRANS



- The customer dimension table is a SAS data set named CUSTOMER_SCD.
- Initially, CUSTOMER_SCD has the same columns as CUSTOMER_TRANS. The Target Table Designer could be used to import the column metadata from CUSTOMER_TRANS into the metadata for CUSTOMER_SCD. For more information about the Target Table Designer, see “Example: Using the Target Table Designer to Register SAS Tables” on page 140.

- In this example, three additional columns will be specified for CUSTOMER_SCD: two change-tracking columns (VALID_FROM_DTTM and VALID_TO_DTTM) and a column for generated keys (CUSTOMER_GEN_KEY).
- CUSTOMER_TRANS and CUSTOMER_SCD have been registered in a current metadata repository.
- You have selected a default SAS application server for SAS Data Integration Studio, as described in "Selecting a Default SAS Application Server" on page 96. This server can access all tables that are used in the job.
- The main metadata repository is under change-management control. For details about change management, see "Working with Change Management" on page 113.
- It is assumed that you have started SAS Data Integration Studio and have opened the appropriate metadata profile.

The first task is to check out any existing metadata that must be updated for the current job.

Check Out Existing Metadata That Must Be Updated

You do not have to check out the metadata for a table in order to add it as a source or a target in a job. However, the metadata for the CUSTOMER_SCD table must be checked out because (a) we assume that the metadata for this table was created and checked in earlier, and (b) the metadata for the table must be updated for the current job.

Follow these steps to check out existing metadata:

1. On the SAS Data Integration Studio desktop, select the **Inventory** tab.
2. In the Inventory tree, open the **Tables** folder.
3. Select the table that must be updated for the current job: CUSTOMER_SCD.
4. Select **Project ▶ Check Out** from the menu bar. The metadata for this table will be checked out and will appear in the Project tree.

The next task is to create and populate the job.

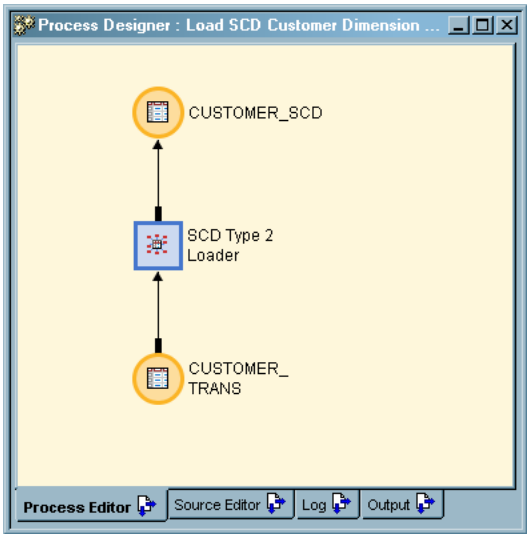
Create and Populate the Job

Follow these steps to populate the job **Load the Customer Dimension**:

1. In SAS Data Integration Studio, in the **Shortcuts** pane, click **Process Designer** to start the New Job Wizard.
2. In the **New Job Wizard**, type the job name **Load the Organization Dimension** and click **Finish**. An empty Process Designer window is displayed.
3. In the tree view, click the **Process Library** tab, then expand the **Data Transforms** folder.
4. In the **Data Transforms** folder, click and drag **SCD Type 2 Loader** into the Process Designer window. Release the mouse button to display the SCD Type 2 Loader transformation template in the Process Designer window for the new job. The template displays with drop zones for a source and a target.
5. In the tree view, select the **Inventory** tab.
6. In the Inventory tree, open the **Tables** folder.
7. In the **Tables** folder, click and drag the **CUSTOMER_TRANS** table into the source drop area of the SCD Type 2 Loader.
8. In the tree view, click the **Project** tab.

9 In the Project tree, click and drag **CUSTOMER_SCD** into the target drop area of the SCD Type 2 Loader transformation. The job is now fully populated with tables and transformations, as shown in the following display.

Display 12.5 Fully Populated Job for Loading the Dimension Table CUSTOMER_SCD



The next step is to add tracking columns and a generated key column to the CUSTOMER_SCD table.

Add SCD Columns to the Dimension Table

Follow these steps to add tracking columns and a generated key column to the CUSTOMER_SCD table:

1 In the Process Designer window, double-click the icon for the CUSTOMER_SCD table to display its properties window.

2 In the properties window, click the **Columns** tab. Initially, the CUSTOMER_SCD dimension table was created with the same columns as the source table CUSTOMER_TRANS. (These columns are shown in Display 12.4 on page 204.) To implement slowly changing dimensions, the dimension table needs three new columns.

3 To add the first new column, click **CUSTOMER_ID**, then click **New**. A new untitled column appears beneath CUSTOMER_ID.

4 Replace the default name of the new column with the name **VALID_FROM_DTTM**. This column will contain the date and time that each row was physically loaded into the table. When the job is run, data will be provided for this column by the SCD Type 2 Loader.

5 In the row for **VALID_FROM_DTTM**, double-click the **Type** column and select **Numeric**.

6 In the row for **VALID_FROM_DTTM**, double-click the **Format** column and type **Datetime20.**.

7 In the row for **VALID_FROM_DTTM**, click **New**. A new untitled column appears beneath **VALID_FROM_DTTM**.

8 Replace the default name of the second new column with the name **VALID_TO_DTTM**. This column will contain the date and time that each row was

superseded by a new current row. When the job is run, data will be provided for this column by the SCD Type 2 Loader.

9 In the row for VALID_TO_DTTM, specify the **Numeric** type and the **Datetime20.** format.

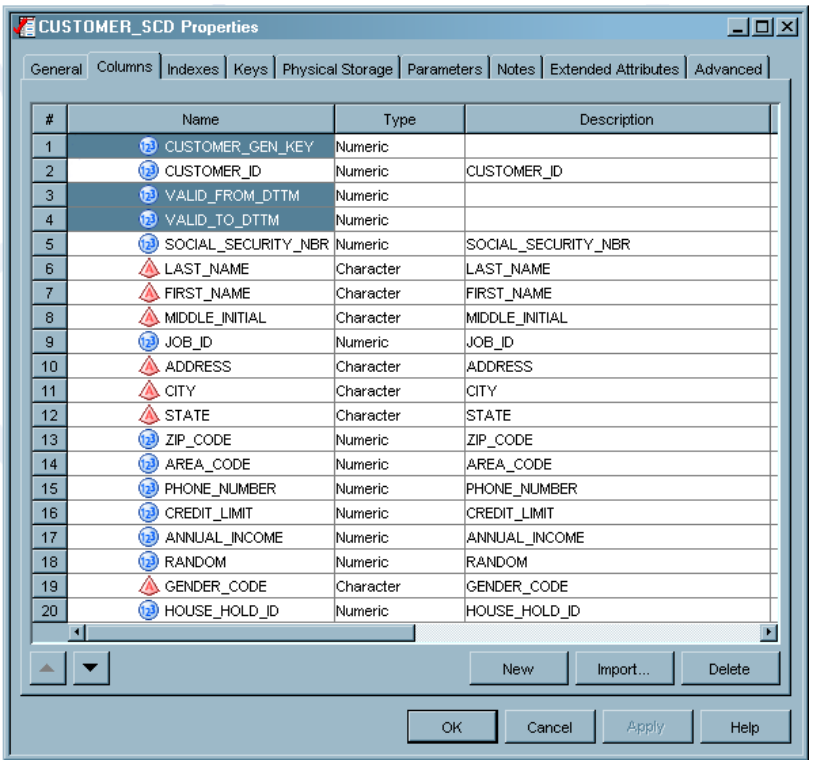
10 To add the third new column, click **New** and replace the default column name with the name **CUSTOMER_GEN_KEY**. This column will provide unique key values for all of the rows in the table. The new key values will be generated by the SCD Type 2 Loader.

11 Press the TAB key twice, then double click and select the **Numeric** data type.

12 Press the TAB key once, then click and type **12.**, which is a numeric format.

13 Click the row number on the far left for the row CUSTOMER_GEN_KEY and drag the column up to position number one. The columns of the dimension table are now configured for slowly changing dimensions, as shown in the following display.

Display 12.6 Columns in the Dimension Table



14 Click **Apply** to save your changes.

The next step is to designate the generated column as the primary key for the dimension table.

Specify the Primary Key for the Dimension Table

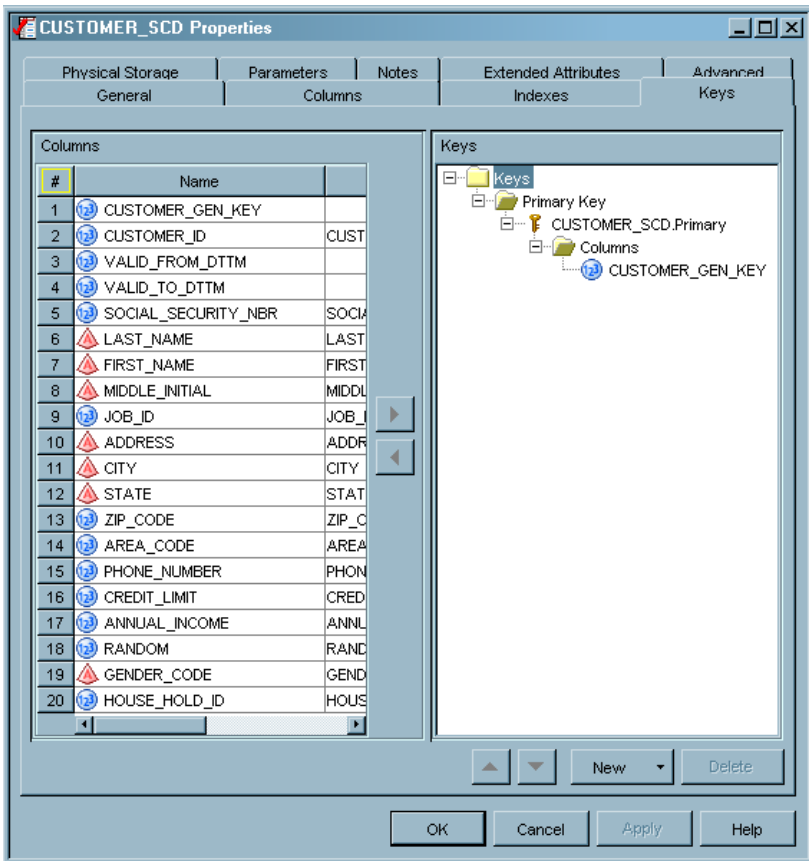
Follow these steps to designate the generated column as the primary key for the dimension table:

1 In the properties window for the CUSTOMER_SCD table, click the **Keys** tab. Locate the **New** button in the left pane of the tab.

2 Click the down arrow to the right of the **New** button and select **Primary Key** from the pull-down menu. Default metadata for a primary key appears.

3 In the right pane of the tab, select the CUSTOMER_GEN_KEY column, then click the right arrow. The CUSTOMER_GEN_KEY column is now specified as the primary key of the dimension table, as shown in the following display.

Display 12.7 Primary Key for the Dimension Table



4 Click **OK** to save your changes and close the properties window for the dimension table.

You have now configured the metadata for the dimension table. The next step is to configure keys and change tracking for the SCD loader.

Specify the Business Key for the SCD Loader

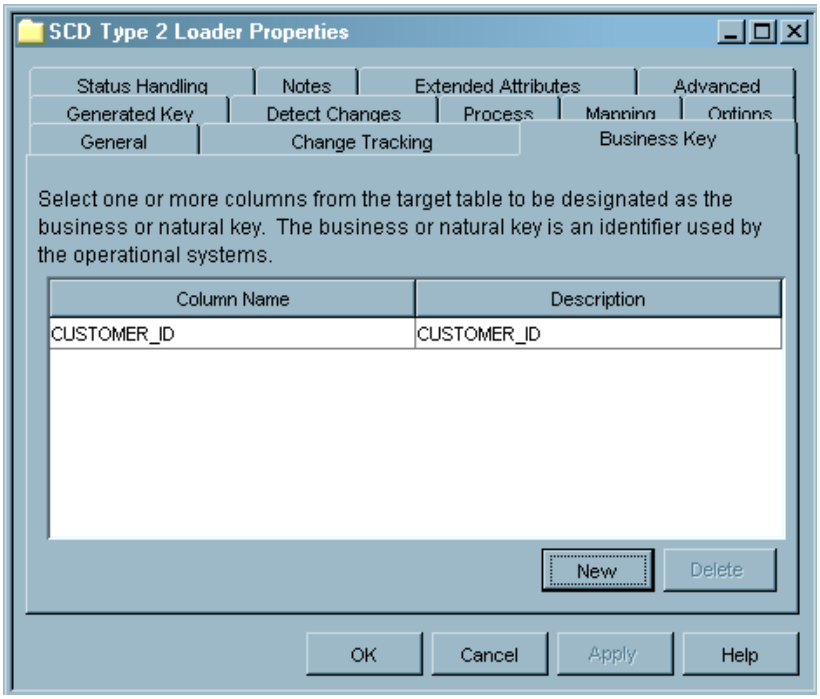
The business key is the primary key of a source table that is used to load the dimension table. In this example, the business key would be the CUSTOMER_ID column in the CUSTOMER_TRANS table. Follow these steps to specify the business key for the SCD loader:

1 In the Process Designer window, double-click the icon for the SCD Type 2 Loader to display its properties window.

2 In the properties window, click the **Business Key** tab. Click **New** to display the column selection window.

3 In the column selection window, select the CUSTOMER_ID column and click **OK**. The CUSTOMER_ID column displays as the business key on the **Business Key** tab, as shown in the following display.

Display 12.8 Business Key Specified for the SCD Type 2 Loader



4 Click **Apply** to save your changes.

The next step is to specify a generated key for the SCD loader.

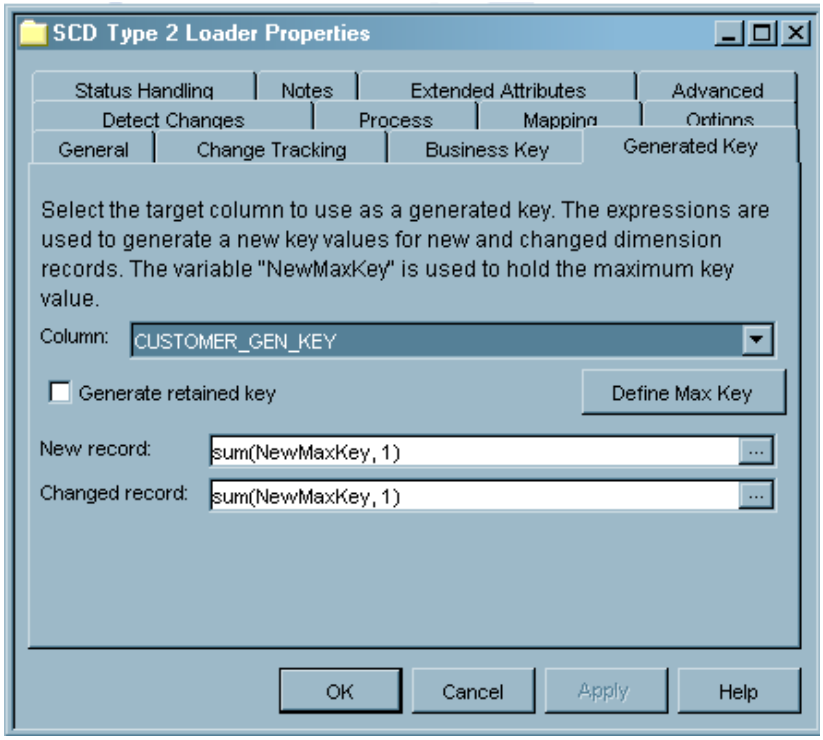
Specify the Generated Key for the SCD Loader

Follow these steps to specify the generated key for the SCD loader:

1 In the properties window for the SCD Type 2 Loader, click the **Generated Keys** tab.

2 Click the down arrow to the right of the **Column** field and select the generated column that you specified earlier for the dimension table (CUSTOMER_GEN_KEY). Assume that the default values shown in the following display are correct, and no further updates are needed on this tab.

Display 12.9 Generated Key for the SCD Loader



For more information about using this tab, click **Help**.

3 Click **Apply** to save your changes.

The next step is to set up change tracking.

Set Up Change Tracking in the SCD Loader

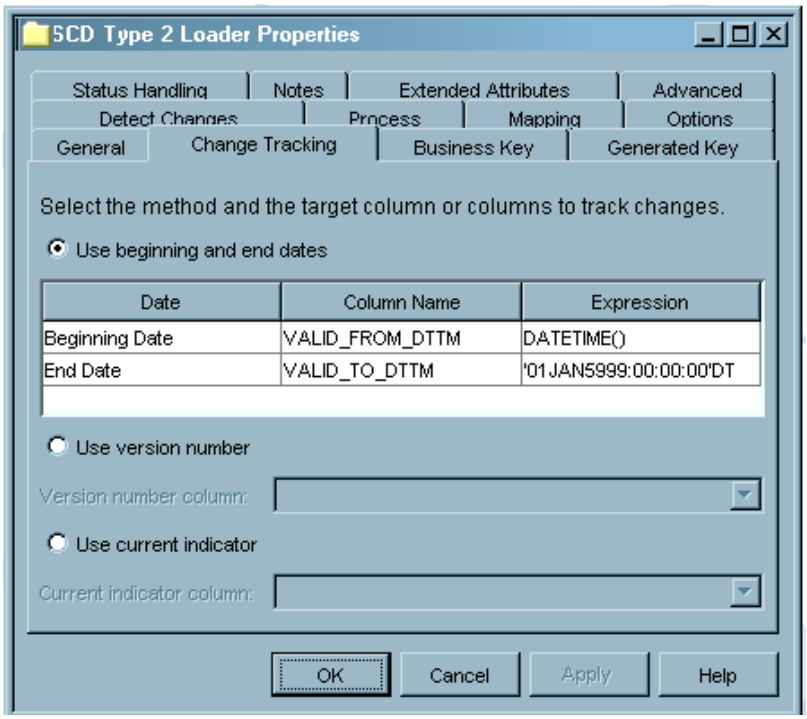
Follow these steps to configure the automatic insertion of date and time information in the dimension table:

1 In the properties window for the SCD Type 2 Loader, click the **Change Tracking** tab.

2 In the **Beginning Date** row, in the **Column Name** column, double-click to display the pull-down menu. Then select the beginning date column in the dimension table (VALID_FROM_DTTM).

3 In the **`End Date`** row, in the **`Column Name`** column, double-click to display the pull-down menu. Then select the end date column in the dimension table (VALID_TO_DTTM). The **`Change Tracking`** tab will look similar to the following display.

Display 12.10 Change Tracking Tab



4 Click **`Apply`** to save your changes.

The last configuration step is to set up change detection.

Set Up Change Detection in the SCD Loader

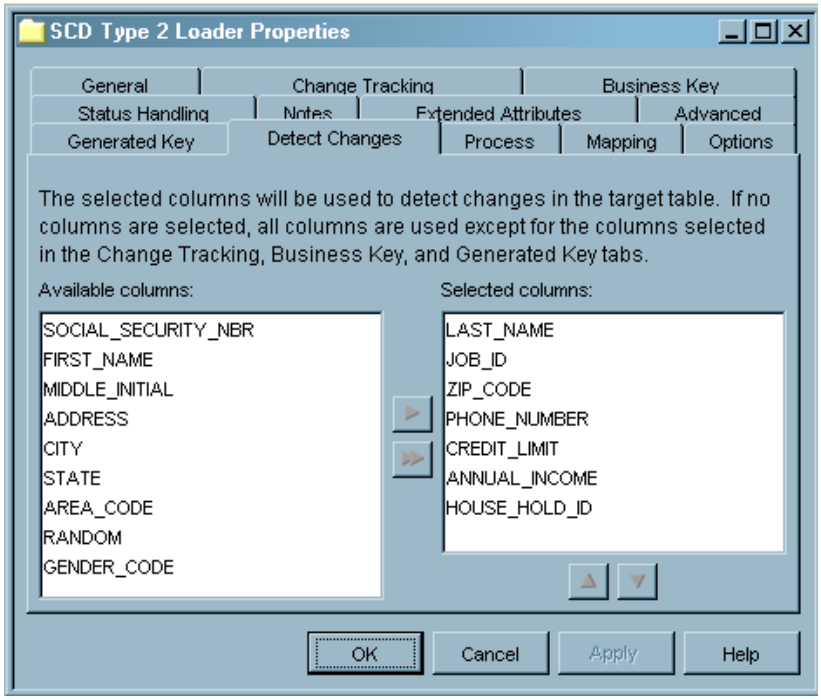
Follow these steps to configure change detection in the SCD loader:

1 In the properties window for the SCD Type 2 Loader, click the **`Detect Changes`** tab.

2 Press and hold the CTRL key, then click the columns for which you will detect changes. In this example, select LAST_NAME, JOB_ID, ZIP_CODE, PHONE_NUMBER, CREDIT_LIMIT, ANNUAL_INCOME, and HOUSE_HOLD_ID.

3 Click the right arrow to move the selected columns into the right pane. The **`Detect Changes`** tab will look similar to the following display.

Display 12.11 Detect Changes Tab



4 Click **`Apply`** to save your changes.

You have now configured the process flow shown in Display 12.5 on page 206. The next step is to run the job.

Run the Job and View the Results

The slowly changing dimension job is now ready to run. In the Process Designer, right-click and select **`Save`**. Then right-click and select **`Submit`**.

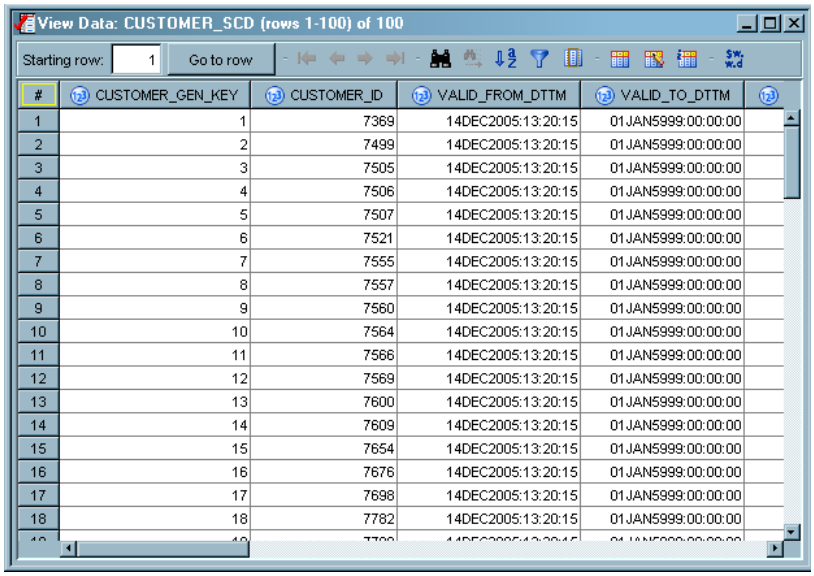
If job execution terminates due to errors, click the **`Log`** tab, locate the error, resolve the error in the Process Editor, and submit the job again.

To view the results of the job, click the **`Process Editor`** tab, right-click **`ORGANIZATION_DIM`**, and select **`View Data`**. The View Data window shows typical results for initial loads of new slowly changing dimension tables, including the following:

- sequential generated key numbers for CUSTOMER_GEN_KEY
- the current date and time in VALID_FROM_DTTM
- fictitious place-holding date/time values in VALID_TO_DTTM

The fictitious date/time values in VALID_TO_DTTM will be replaced in subsequent loads as new rows supersede existing rows.

Display 12.12 Data in Dimension Table CUSTOMER_SCD


View Data: CUSTOMER_SCD (rows 1-100) of 100

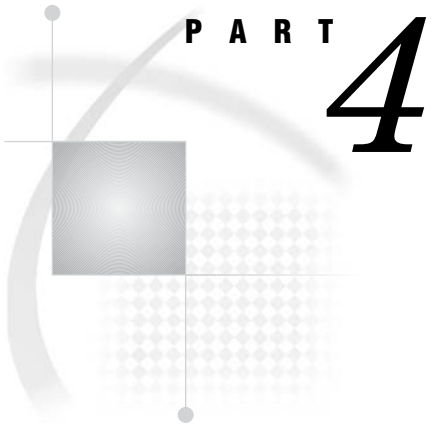
Starting row: 1 Go to row

#	CUSTOMER_GEN_KEY	CUSTOMER_ID	VALID_FROM_DTTM	VALID_TO_DTTM
1	1	7369	14DEC2005:13:20:15	01JAN5999:00:00:00
2	2	7499	14DEC2005:13:20:15	01JAN5999:00:00:00
3	3	7505	14DEC2005:13:20:15	01JAN5999:00:00:00
4	4	7506	14DEC2005:13:20:15	01JAN5999:00:00:00
5	5	7507	14DEC2005:13:20:15	01JAN5999:00:00:00
6	6	7521	14DEC2005:13:20:15	01JAN5999:00:00:00
7	7	7555	14DEC2005:13:20:15	01JAN5999:00:00:00
8	8	7557	14DEC2005:13:20:15	01JAN5999:00:00:00
9	9	7560	14DEC2005:13:20:15	01JAN5999:00:00:00
10	10	7564	14DEC2005:13:20:15	01JAN5999:00:00:00
11	11	7566	14DEC2005:13:20:15	01JAN5999:00:00:00
12	12	7569	14DEC2005:13:20:15	01JAN5999:00:00:00
13	13	7600	14DEC2005:13:20:15	01JAN5999:00:00:00
14	14	7609	14DEC2005:13:20:15	01JAN5999:00:00:00
15	15	7654	14DEC2005:13:20:15	01JAN5999:00:00:00
16	16	7676	14DEC2005:13:20:15	01JAN5999:00:00:00
17	17	7698	14DEC2005:13:20:15	01JAN5999:00:00:00
18	18	7782	14DEC2005:13:20:15	01JAN5999:00:00:00

Check In the Metadata

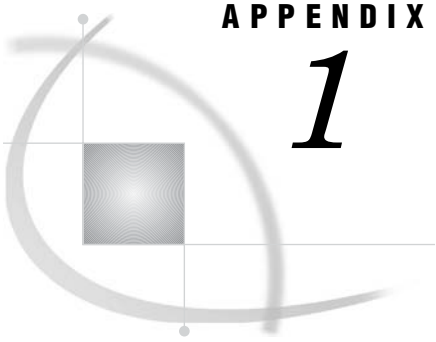
Check in all metadata in the Project tree as follows:

1. In the Project tree, select the repository icon.
2. From the SAS Data Integration Studio menu bar, select **Project ▸ Check In Repository**. All of the objects in the project repository are checked into the change-managed repository.



PART 4

Appendixes



APPENDIX

1

Standard Transformations in the Process Library

About the Process Library

Overview of the Process Library

The Process Library tree is one of the hierarchical lists in the tree view on the SAS Data Integration Studio desktop. It displays a collection of transformation templates. A transformation template is a process flow diagram that includes drop zones for metadata that the user must supply. A template typically consists of a transformation object and one or more drop zones for sources, targets, or both.

The Process Library organizes transformations into a set of folders, such as the Access folder for transformations that you can use to read or write specific kinds of data, and the Analysis folder for transformations that you can use to analyze data in various ways.

Access Folder

The following table describes the transformations in the Access folder in the Process Library.

Table A1.1 Access Folder Transformations

Name	**Description**
File Reader	reads an external file and writes to a target. Added automatically to a process flow when an external file is specified as a source. Executes on the host where the external file resides, as specified in the metadata for the external file.
File Writer	reads a source and writes to an external file. Added automatically to a process flow when an external file is specified as a target. Executes on the host where the external file resides, as specified in the metadata for the external file.
Library Contents	creates a control table to use with an iterative job, a job with a control loop in which one or more processes are executed multiple times.
SPD Server Table Loader	reads a source and writes to a SAS SPD Server target. It is automatically added to a process flow when a SAS SPD Server table is specified as a source or as a target. Enables you to specify options that are specific to SAS SPD Server tables.
Table Loader	reads a source table and writes to a target table. Added automatically to a process flow when a table is specified as a source or a target.

Analysis Folder

The following table describes the transformations in the Analysis folder in the Process Library.

Table A1.2 Analysis Folder Transformations

Name	**Description**
Correlations	creates an output table that contains correlation statistics.
Correlations - Report	creates an HTML report that contains summary correlation statistics.
Distribution Analysis	creates an output table that contains a distribution analysis.
Distribution Analysis - Report	creates an HTML report that contains a distribution analysis.
Forecasting	generates forecasts for sets of time-series or transactional data.
Frequency	
Frequency - Report	creates an output table that contains frequency information.
Summary Statistics	creates an output table that contains summary statistics.

Name	**Description**
Summary Statistics - Report	creates an HTML report that contains summary statistics.
Summary Tables - Report	creates an HTML report that contains summary tables.

Control Folder

The following table describes the transformations in the Control folder in the Process Library.

Table A1.3 Control Folder Transformations

Name	**Description**
Loop	marks the beginning of the iterative processing sequence in an iterative job.
Loop End	marks the end of the iterative processing sequence in an iterative job.
Return Code Check	provides status-handling logic at a desired point in the process flow diagram for a job. Can be inserted between existing transformations and removed later without affecting the mappings in the original process flow.

Data Transforms Folder

The following table describes the transformations in the Data Transforms folder in the Process Library.

Table A1.4 Data Transforms Folder Transformations

Name	**Description**
Append	creates a single target table by combining data from several source tables.
Apply Lookup Standardization	applies scheme data sets to transform source columns.
Create Match Code	establishes relationships between source rows using match code analysis or cluster analysis.
Data Transfer	moves data directly from one machine to another. Direct data transfer is more efficient than the default transfer mechanism.
Data Validation	cleanses data before it is added to a data warehouse or data mart.
Extract	selects multiple sets of rows from a source and writes those rows to a target. Typically used to create one subset from a source. Can also be used to create columns in a target that are derived from columns in a source.
Fact Table Lookup	loads source data into a fact table and translates business keys into generated keys.

Name	**Description**
Key Effective Date	enables change tracking in intersection tables.
Lookup	loads a target with columns taken from a source and from several lookup tables.
Mining Results	integrates a SAS Enterprise Miner model into a SAS Data Integration Studio data warehouse. Typically used to create target tables from a SAS Enterprise Miner model.
SAS Rank	ranks one or more numeric column variables in the source and stores the ranks in the target.
SAS Sort	reads data from a source, sorts it, and writes the sorted data to a target.
SAS Splitter	selects multiple sets of rows from one source and writes each set of rows to a different target. Typically used to create two or more subsets of a source. Can also be used to create two or more copies of a source.
SCD Type 2 Loader	loads source data into a dimension table, detects changes between source and target rows, updates change tracking columns, and applies generated key values. This transformation implements slowly changing dimensions.
SQL Join	selects multiple sets of rows from one or more sources and writes each set of rows to a single target. Typically used to merge two or more sources into one target. Can also be used to merge two or more copies of a single source.
Standardize	creates an output table that contains data standardized to a particular number.
Surrogate Key Generator	loads a target, adds generated whole number values to a surrogate key column and sorts and saves the source based on the values in the business key column(s).
Transpose	creates an output table that contains transposed data.
User Written Code	retrieves a user-written transformation. Can be inserted between existing transformations and removed later without affecting the mappings in the original process flow. Can also be used to document the process flow for the transformation so that you can view and analyze the metadata for a user-written transformation, similarly to how you can for other transformations.

Output Folder

The following table describes the transformations in the Output folder in the Process Library.

Table A1.5 Output Folder Transformations

Name	**Description**
List Data	creates an HTML report that contains selected columns from a source table. See also “Adding a List Data Transformation to the Process Flow” on page 193.

Publish Folder

The following table describes the transformations in the Publish folder in the Process Library.

Table A1.6 Publish Folder Transformations

Name	**Description**
Publish to Archive	creates an HTML report and an archive of the report.
Publish to Email	creates an HTML report and e-mails it to a designated address.
Publish to Queue	creates an HTML report and publishes it to a queue using MQSeries.

Additional Information About Process Library Transformations

To display examples that illustrate how the Process Library transformations can be used in SAS Data Integration Studio jobs, follow these steps:

1. From the SAS Data Integration Studio menu bar, select **Help ▸ Contents**. The Help window displays.
2. In the left pane of the Help window, select **Examples ▸ Process Library Examples**.

APPENDIX

2 Customizing or Replacing Generated Code in SAS Data Integration Studio

Methods of Customizing or Replacing Generated Code

There are several ways to customize or replace the code that SAS Data Integration Studio generates for a job and the transformations within a job.

Table A2.1 Methods of Customizing or Replacing Generated Code

Method	Description
Modify configuration files or SAS Start commands	You can modify the config.sas file, the autoexec.sas file, or the SAS start command for the SAS Workspace Server or servers that are to execute SAS Data Integration Studio jobs. Use this method to specify SAS system options or startup options for all jobs that are executed on a particular SAS Workspace server.
Specify options on the **Code Generation** tab	In SAS Data Integration Studio 3.3, you can specify options on the **Code Generation** tab on the general Options window. Use this method to affect the code that is generated for all new jobs. Many of these settings are used for parallel processing and grid computing.
Add SAS code to the **Pre and Post Processing** tab in the properties window for a job	Use this method to execute SAS code at the beginning or end of a job.
Specify options on the **Code Generation** tab	You can specify SAS system options, SAS statement options, or transformation-specific options on the **Options** tab or other tabs in the properties window for many transformations. Use this method to take advantage of these options when a particular transformation executes.

Method	Description
Replace the generated code for a transformation with user-written code	For example, you can edit the code that is generated for a transformation, save it, and then set the transformation so that the customized code runs whenever the transformation is executed.
Add a User-Written Code transformation to the process flow for a job	The User-Written Code transformation is one of the transformations in the Process Library. Use this transformation to perform custom tasks in for a specific process flow—tasks that are not supported by the standard transformations and are beyond the scope that can be managed by making a few changes to the generated code for a single transformation.User-Written Code transformations do not support multiple inputs or outputs.
Add a generated transformation to the Process Library	The Transformation Generator wizard guides you through the steps of specifying SAS code for a generated transformation. After the transformation is saved, it is displayed in the Process Library tree, where it is available for use in any job. Use a generated transformation when you want to create reusable, custom code that you can insert into the flow for any SAS Data Integration Studio job. Unlike User-Written Code transformations, generated transformations support multiple inputs or outputs.

Modifying Configuration Files or SAS Start Commands

To specify SAS system options or startup options for all jobs that are executed on a particular SAS Workspace Server, modify the config.sas file, the autoexec.sas file, or the SAS start command for that server. Examples of these options include UTILLOC or NOWORKINIT. For more information about SAS configuration files and SAS invocation options, see the topics that are releated to "configuration" and "invoking SAS" in your SAS online documentation.

One server option that is supported by SAS Data Integration Studio 3.3 or later is the ETLS_DEBUG option. By default, SAS Data Integration Studio specifies the MPRINT option at the start of the code that is generated for all jobs. The MPRINT option adds messages to the log that are useful for debugging in a development environment. However, MPRINT can make the log very large, which might not be desirable in a production environment.

The ETLS_DEBUG option provides a way to turn off the MPRINT option for all jobs that are executed on a particular SAS Workspace Server, without having to turn off this option for each job. If the ETLS_DEBUG option is specified for a SAS Workspace Server, and a job is executed on that server, the default MPRINT option is ignored.

Here is the syntax for the ETLS_DEBUG option:

```
%let etls_debug=0;
```

Note: Setting ETLS_DEBUG to any value other than 0 has one effect: MPRINT is turned on. This might change in future releases. We suggest that you add this assignment with a value equal to 0; otherwise, leave the statement out. △

CAUTION:

We strongly recommend that you do not turn off MPRINT in a development environment. △

See also “Setting SAS Options for Jobs and Transformations” on page 189.

Specifying Options in the Code Generation Tab

In SAS Data Integration Studio 3.3, you can specify options on the **Code Generation** tab in the general Options window. Use this method to affect the code that is generated for all new jobs. Many of these settings are used for parallel processing and grid computing.

To display the general Options window from the SAS Data Integration Studio desktop, select **Tools** ▶ **Options** from the menu bar. For a description of the available options, see the Help for the tab.

Adding SAS Code to the Pre and Post Processing Tab

You can add SAS code on the **Pre and Post Processing** tab in the properties window for a job. To display the properties window for a job, right-click the icon for the job in the Inventory tree and select **Properties** from the pop-up window. Use this method to execute SAS code at the beginning or end of a job. For details about the steps for adding code, see the Help for the tab. See also "(Optional) Reduce the Amount of Data Processed by the Job" on page 153.

To redirect the SAS log and SAS output to specific locations, you could add this code on the **Pre and Post Processing** tab:

```
PROC PRINTTO LOG=''
PRINT='';
RUN;
```

You can also specify option statements for options such as FULLSTIMER, MSGLEVEL, SYMBOLGEN, UBUFNO, WORKTERM, BLKSIZE and BUFSIZE if you want these options to apply to an entire job.

By default, SAS Data Integration Studio specifies the MPRINT option at the start of the code that is generated for all jobs (before any pre-processing code). The MPRINT option adds messages to the log that are useful for debugging in a development environment. To turn off the default MPRINT option for a particular job, add the following code to the **Pre and Post Processing** tab:

```
OPTIONS NOMPRINT;
```

See also "Redirecting SAS Data Integration Studio's Log to a File" on page 191.

Specifying Options for Transformations

You can specify SAS system options, SAS statement options, or transformation-specific options on the **Options** tab or other tabs in the properties window for many transformations. Use this method to take advantage of these options when a particular transformation executes.

Follow these steps to display the properties window for a transformation in a job:

1. Open the job to display its process flow.
2. Right-click the transformation and select **Properties** from the pop-up menu.

For a description of the available options for a particular transformation, see the Help for the **Options** tab or other tabs that enable you to specify options.

If the **Options** tab includes a system options field, you can specify options such as UBUFNO for the current transformation. Some transformations enable you to specify

options that are specific to that transformation. For example, the **Options** tab for the Sort transformation has specific fields for sort size and sort sequence. It also has a **PROC SORT Options** field where you can specify sort-related options that are not otherwise surfaced in the interface. For example, you could use these fields to specify the options that are described in the "Sorting" section.

In addition to the **Options** tab, some transformations have other tabs that enable you to specify options that affect performance. For example, the properties window for the SAS Scalable Performance Server Loader transformation has an **Other SPD Server Settings** tab, which enables you to specify several SAS Scalable Performance Server options.

Replacing the Generated Code for a Transformation with User-Written Code

You can replace the generated code for a transformation with user-written code. For example, you can edit the code that is generated for a transformation, save it, and then set the transformation so that the customized code runs whenever the transformation is executed. You can switch back to have SAS regenerate the step's code at any time. (The edited version persists, but it is ignored.) Column mappings between the edited step and its predecessor and successor steps remain intact.

You can use this method when you find that a transformation does not yet provide all choices that you need in the point-and-click interface and the provided option fields. For example, you could use this method to change the default join order that is generated for the SQL Join transformation, as described in the section "Join Order and Performance."

The following steps assume that a process flow has already been created, that you know what modifications you want to make in the generated code for a particular transformation in the flow, and that the flow displays in the Process Editor.

1. In the Process Editor, right-click the desired transformation and then select **Process ► View Step Code**. Code is generated for the transformation and displays in the Source Editor window.
2. In the Source Editor window, edit the generated code.
3. When finished, click the **X** in the upper right corner of the Source Editor.
4. Select **Yes** to save your changes. A file selection window displays.
5. From the file selection window, specify a path name for the edited code, and click **Save**. The edited code is saved to a file.

After you have save the customized code, the next task is to set the transformation so that the customized code runs whenever the transformation is executed. Follow these steps:

1. In the Process Designer window, select the transformation and then select **File ► Properties** from the menu bar. The properties window for the transformation displays.
2. Click the **Process** tab.
3. On the **Process** tab, select the **User Written** button. The **Type** field and related fields become active.
4. In the **Type** field, select **File**.
5. Typically, you should accept the default host in the **Host** field.

6 In the **Path** field, specify a path to the file that contains the edited code. The server in the **Host** field must be able to resolve this path. You can enter this path or use the **Browse** button to display a file-selection window.

7 After specifying the path, click **OK** to save your changes.

The specified user-written code is retrieved whenever code for this transformation is generated. From that point forward, the edited code is used in the job code generation (until you re-select the **Automatically create source code** option on the **Process** tab). For more details about specifying user-written code for a transformation, see the topic "Specifying User-Written Source Code for a Transformation Within a Job" in the Help for SAS Data Integration Studio.

Adding a User-Written Code Transformation to the Process Flow for a Job

To create a process flow for a job, you can drag and drop transformations from the Process Library tree into the Process Editor. If the predefined transformations in the Process Library tree do not meet your needs for a step in a specific job, you can write a SAS program that performs the desired task, add a User-Written Code transformation to the process flow, and then specify the location of the new code in the metadata for the User-Written Code transformation.

After the transformation has been inserted, you update its metadata so that it specifies the location of your user-written code. When the job is executed, the user-written code is retrieved and executed as part of the job.

You can base your SAS program on code that is generated for one or more standard transformations in SAS Data Integration Studio, or you can use some other SAS code-generation tool such as SAS Enterprise Guide to help build the initial code.

The following steps assume that you have already created a process flow, that you have already written and tested the desired SAS program, that you are ready to add the User-Written Code transformation to the flow, and that the flow displays in the Process Editor.

1 Drag a User-Written Code transformation from the Process Library and drop it into the appropriate location in the process flow.

2 Right-click the icon for the User-Written Code transformation and select **Properties** from the pop-up menu.

3 On the **Source** tab, either paste the revised code as metadata or specify the path to the saved code (if you plan to maintain user-written code in an external location).

4 Make any final edits of the code; be sure the input and/or output table names are correct. Use &SYSLAST if the input is to refer to the output from the predecessor step, and specify &_OUTPUT as your output table name if your output is a single work dataset being used to feed the successor step.

5 Manually define columns on the **Mapping** tab if other steps are to follow. Use the **Quick Propagate** option on the **Mapping** tab and apply further edits to define the output columns correctly.

When SAS Data Integration Studio generates all code for a job, it can automatically generate the metadata for column mappings between sources and targets. However, when you specify user-written code for part of a job, you might have to manually define the column metadata for that part of the job that the user-written code handles. SAS Data Integration Studio needs this metadata to generate the code for the part of the job that comes after the user-written code step.

Note: In a process flow, a User-Written Code transformation can have only one input and one output. If you need additional inputs or outputs, consider adding a generated transformation, as described in the next section. △

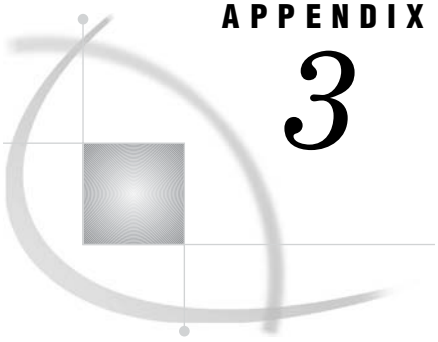
For more details about using a User-Written Code transformation in a job, see "Example: Include a User-Written Code Template in a Job" in the Help for SAS Data Integration Studio.

Adding a Generated Transformation to the Process Library

If have custom code that you would want to use in more than one place, or if there are multiple inputs or outputs for your custom code segment, use the Transformation Generator wizard to create a generated transformation.

To display the Transformation Generator wizard from the SAS Data Integration Studio desk top, select **Tools ► Transformation Generator** from the menu bar. In the wizard, specify the number of inputs and outputs, any custom options, then attach your user-written SAS code, just as you would for the User-Written transformation described in the previous section. The newly created transformation appears in the Process Library tree, where it is available for use in any SAS Data Integration Studio job.

Drop the generated transformation into the process flow for a job and set any options. When the job is executed, the user-written code is retrieved and executed as part of the job. For more information, see "Maintaining Generated Transformations" on page 75.



APPENDIX

3

Recommended Reading

Recommended Reading

Here is the recommended reading list for this title:

- *Cody's Data Cleaning Techniques Using SAS Software*
- *Communications Access Methods for SAS/CONNECT and SAS/SHARE*
- *Moving and Accessing SAS Files*
- *PROC SQL: Beyond the Basics Using SAS*
- *SAS Intelligence Platform: Administration Guide*
- *SAS Intelligence Platform: Installation Guide*
- *SAS Intelligence Platform: Overview*
- *SAS Intelligence Platform: Security Administration Guide*
- *SAS Management Console: User's Guide*
- *SAS OLAP Server: Administrator's Guide*
- *SAS SQL Procedure User's Guide*

For a complete list of SAS publications, see the current *SAS Publishing Catalog*. To order the most current publications or to receive a free copy of the catalog, contact a SAS representative at

SAS Publishing Sales
SAS Campus Drive
Cary, NC 27513
Telephone: (800) 727-3228*
Fax: (919) 677-8166
E-mail: **sasbook@sas.com**
Web address: **support.sas.com/pubs**
* For other SAS Institute business, call (919) 677-8000.

Customers outside the United States should contact their local SAS office.

Glossary

administrator
the person who is responsible for maintaining the technical attributes of an object such as a table or a library. For example, an administrator might specify where a table is stored and who can access the table. See also owner.

alternate key
another term for unique key. See unique key.

analysis data set
in SAS data quality, a SAS output data set that provides information on the degree of divergence in specified character values.

business key
one or more columns in a dimension table that comprise the primary key in a source table in an operational system.

change analysis
the process of comparing one set of metadata to another set of metadata and identifying the differences between the two sets of metadata. For example, in SAS Data Integration Studio, you have the option of performing change analysis on imported metadata. Imported metadata is compared to existing metadata. You can view any changes in the Differences window and choose which changes to apply. To help you understand the impact of a given change, you can run impact analysis or reverse impact analysis on tables and columns in the Differences window.

change management
in the SAS Open Metadata Architecture, a facility for metadata source control, metadata promotion, and metadata replication.

change-managed repository
in the SAS Open Metadata Architecture, a metadata repository that is under metadata source control.

cluster
in SAS data quality, a set of character values that have the same match code.

comparison result
the output of change analysis. For example, in SAS Data Integration Studio, the metadata for a comparison result can be selected, and the results of that comparison can be viewed in a Differences window and applied to a metadata repository. See also change analysis.

cross-reference table
a table that contains only the current rows of a larger dimension table. Columns generally include all business key columns and a digest column. The business key column is used to determine if source rows are new dimensions or updates to existing dimensions. The digest column is used to detect changes in source rows that might update an existing dimension. During updates of the fact table that is associated with the dimension table, the cross-reference table can provide generated keys that replace the business key in new fact table rows.

custom repository
in the SAS Open Metadata Architecture, a metadata repository that must be dependent on a foundation repository or custom repository, thus allowing access to metadata definitions in the repository or repositories on which it depends. A custom repository is used to specify resources that are unique to a particular data collection. For example, a custom repository could define sources and targets that are unique to a particular data warehouse. The custom repository would access user definitions, group definitions, and most server metadata from the foundation repository. See also foundation repository, project repository.

data analysis
in SAS data quality, the process of evaluating input data sets in order to determine whether data cleansing is needed.

data cleansing
the process of eliminating inaccuracies, irregularities, and discrepancies from character data.

data lineage
a search that seeks to identify the tables, columns, and transformations that have an impact on a selected table or column. See also impact analysis, reverse impact analysis, transformation.

data transformation
in SAS data quality, a cleansing process that applies a scheme to a specified character variable. The scheme creates match codes internally to create clusters. All values in each cluster are then transformed to the standardization value that is specified in the scheme for each cluster.

database library
a collection of one or more database management system files that are recognized by SAS and that are referenced and stored as a unit. Each file is a member of the library.

database server
a server that provides relational database services to a client. Oracle, DB/2 and Teradata are examples of relational databases.

delimiter
a character that separates words or phrases in a text string.

derived mapping
a mapping between a source column and a target column in which the value of the target column is a function of the value of the source column. For example, if two tables contain a Price column, the value of the target table's Price column might be equal to the value of the source table's Price column multiplied by 0.8.

digest column
a column in a cross-reference table that contains a concatenation of encrypted values for specified columns in a target table. If a source row has a digest value that differs from the digest value for that dimension, then changes are detected and the source

row becomes the new current row in the target. The old target row is closed out and receives a new value in the end date/time column.

dimension
a category of contextual data or detail data that is implemented in a data model such as a star schema. For example, in a star schema, a dimension named Customers might associate customer data with transaction identifiers and transaction amounts in a fact table.

dimension table
in a star schema or snowflake schema, a table that contains data about a particular dimension. A primary key connects a dimension table to a related fact table. For example, if a dimension table named Customers has a primary key column named Customer ID, then a fact table named Customer Sales might specify the Customer ID column as a foreign key.

fact table
the central table in a star schema or snowflake schema. A fact table typically contains numerical measurements or amounts and is supplemented by contextual information in dimension tables. For example, a fact table might include transaction identifiers and transaction amounts. Dimension tables could add contextual information about customers, products, and salespersons. Fact tables are associated with dimension tables via key columns. Foreign key columns in the fact table contain the same values as the primary key columns in the dimension tables.

foreign key
one or more columns that are associated with a primary key or unique key in another table. A table can have one or more foreign keys. A foreign key is dependent upon its associated primary or unique key. In other words, a foreign key cannot exist without that primary or unique key.

foundation repository
in the SAS Open Metadata Architecture, a metadata repository that is used to specify metadata for global resources that can be shared by other repositories. For example, a foundation repository is used to store metadata that defines users and groups on the metadata server. Only one foundation repository should be defined on a metadata server. See also custom repository, project repository.

generated key
a column in a dimension table that contains values that are sequentially generated using a specified expression. Generated keys are used to implement surrogate keys and retained keys.

global resource
an object, such as a server or a library, that is shared on a network.

impact analysis
a search that seeks to identify the tables, columns, and transformations that would be affected by a change in a selected table or column. See also transformation, data lineage.

intersection table
a table that describes the relationships between two or more tables. For example, an intersection table could describe the many-to-many relationships between a table of users and a table of groups.

iterative job
a job with a control loop in which one or more processes are executed multiple times. Iterative jobs can be executed in parallel. See also job.

iterative processing
a method of processing in which a control loop executes one or more processes multiple times.

job
a metadata object that specifies processes that create output.

locale
a value that reflects the language, local conventions, and culture for a geographic region. Local conventions can include specific formatting rules for dates, times, and numbers, and a currency symbol for the country or region. Collating sequences, paper sizes, and conventions for postal addresses and telephone numbers are also typically specified for each locale. Some examples of locale values are French_Canada, Portuguese_Brazil, and Chinese_Singapore.

lookup standardization
a process that applies a scheme to a data set for the purpose of data analysis or data cleansing.

match code
an encoded version of a character value that is created as a basis for data analysis and data cleansing. Match codes are used to cluster and compare character values. See also sensitivity.

metadata administrator
a person who defines the metadata for servers, metadata repositories, users, and other global resources.

metadata model
a definition of the metadata for a set of objects. The model describes the attributes for each object, as well as the relationships between objects within the model.

metadata object
a set of attributes that describe a table, a server, a user, or another resource on a network. The specific attributes that a metadata object includes vary depending on which metadata model is being used.

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

metadata server
a server that provides metadata management services to one or more client applications. A SAS Metadata Server is an example.

metadata source control
in the SAS Open Metadata Architecture, a feature that enables multiple users to work with the same metadata repository at the same time without overwriting each other's changes. See also change management.

operational data
data as it exists in the operational system, which is used as source data for a data warehouse.

operational system
one or more programs (frequently relational databases) that provide source data for a data warehouse.

owner
the person who is responsible for the contents of an object such as a table or a library. See also administrator.

parameterized job
a job that specifies its inputs and outputs as parameters. See also job.

parameterized table
a table whose metadata specifies some attributes as variables rather than as literal values. For example, the input to an iterative job could be a parameterized table whose metadata specifies its physical pathname as a variable. See also iterative job.

primary key
one or more columns that are used to uniquely identify a row in a table. A table can have only one primary key. The column(s) in a primary key cannot contain null values. See also unique key, foreign key.

process flow diagram
a diagram in the Process Editor that specifies the sequence of each source, target, and process in a job. In the diagram, each source, target, and process has its own metadata object. Each process in the diagram is specified by a metadata object called a transformation.

project repository
a repository that must be dependent on a foundation repository or custom repository that will be managed by the Change Management Facility. A project repository is used to isolate changes from a foundation repository or from a custom repository. The project repository enables metadata programmers to check out metadata from a foundation repository or custom repository so that the metadata can be modified and tested in a separate area. Project repositories provide a development/testing environment for customers who want to implement a formal change management scheme. See also custom repository, foundation repository.

Quality Knowledge Base
a collection of locales and other information that is referenced during data analysis and data cleansing. For example, to create match codes for a data set that contains street addresses in Great Britain, you would reference the ADDRESS match definition in the ENGBR locale in the Quality Knowledge Base.

register
to save metadata about an object to a metadata repository. For example, if you register a table, you save metadata about that table to a metadata repository.

retained key
a numeric column in a dimension table that is combined with a begin-date column to make up the primary key. During the update of a dimensional target table, source rows that contain a new business key are added to the target. A key value is generated and added to the retained key column and a date is added to the begin-date column. When a source row has the same business key as a row in the target, the source row is added to the target, including a new begin-date value. The retained key of the new column is copied from the target row.

reverse impact analysis
See data lineage.

SAS application server
a server that provides SAS services to a client. In the SAS Open Metadata Architecture, the metadata for a SAS application server specifies one or more server components that provide SAS services to a client.

SAS Management Console
a Java application that provides a single user interface for performing SAS administrative tasks.

SAS OLAP Server
a SAS server that provides access to multidimensional data. The data is queried using the multidimensional expressions (MDX) language.

SAS Open Metadata application
a client application that connects to the SAS Metadata Server and uses metadata from one or more SAS Metadata Repositories.

SAS Open Metadata Architecture
a general-purpose metadata management facility that provides metadata services to SAS applications. The SAS Open Metadata Architecture enables applications to exchange metadata, which makes it easier for these applications to work together.

SAS/CONNECT server
a server that provides SAS/CONNECT services to a client. When SAS Data Integration Studio generates code for a job, it uses SAS/CONNECT software to submit code to remote computers. SAS Data Integration Studio can also use SAS/CONNECT software for interactive access to remote libraries.

SAS/SHARE library
a SAS library for which input and output requests are controlled and executed by a SAS/SHARE server.

SAS/SHARE server
the result of an execution of the SERVER procedure, which is part of SAS/SHARE software. A server runs in a separate SAS session that services users' SAS sessions by controlling and executing input and output requests to one or more libraries.

scheme
a data set that is created from a character variable or column and which is applied to that same character data for the purpose of transformation or analysis.

sensitivity
in SAS data quality, a value that specifies the degree of complexity in newly created match codes. A higher sensitivity value results in greater match code complexity, which in turn results in a larger number of clusters, with fewer members in each cluster.

server administrator
a person who installs and maintains server hardware or software. See also metadata administrator.

server component
in SAS Management Console, a metadata object that specifies information about how to connect to a particular kind of SAS server on a particular computer.

slowly changing dimensions
a technique for tracking changes to dimension table values in order to analyze trends. For example, a dimension table named Customers might have columns for Customer ID, Home Address, Age, and Income. Each time the address or income changes for a customer, a new row could created for that customer in the dimension table, and the old row could be retained. This historical record of changes could be combined with purchasing information to forecast buying trends and direct customer marketing campaigns.

snowflake schema
tables in a database in which a single fact table is connected to multiple dimension tables. The dimension tables are structured to minimize update anomalies and to address single themes. This structure is visually represented in a snowflake pattern. See also star schema.

source
an input to an operation.

star schema
tables in a database in which a single fact table is connected to multiple dimension tables. This is visually represented in a star pattern. SAS OLAP cubes can be created from a star schema.

surrogate key
a numeric column in a dimension table that is the primary key of that table. The surrogate key column contains unique integer values that are generated sequentially when rows are added and updated. In the associated fact table, the surrogate key is included as a foreign key in order to connect to specific dimensions.

target
an output of an operation.

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

transformation template
a process flow diagram that consists of a transformation object and one or more drop zones for sources, targets, or both.

unique key
one or more columns that can be used to uniquely identify a row in a table. A table can have one or more unique keys. Unlike a primary key, a unique key can contain null values. See also primary key, foreign key.

Web service
a programming interface that enables distributed applications to communicate even if the applications are written in different programming languages or are running on different operating systems.

Index

M

N

O

P

Q

R

S

T

U

V

W

X

Your Turn

If you have comments or suggestions about the *SAS Data Integration Studio 3.3: User's Guide*, please send them to us on a photocopy of this page, or send us electronic mail.

For comments about this book, please return the photocopy to

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

For suggestions about the software, please return the photocopy to

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