



SAS Publishing



SAS[®] 9.1

ETL Studio: User's Guide

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2004. *SAS® 9.1 ETL Studio: User's Guide*. Cary, NC: SAS Institute Inc.

SAS® 9.1 ETL Studio: User's Guide

Copyright © 2004, SAS Institute Inc., Cary, NC, USA

ISBN 1-59047-461-9

All rights reserved. Produced in the United States of America. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

U.S. Government Restricted Rights Notice. Use, duplication, or disclosure of this software and related documentation by the U.S. government is subject to the Agreement with SAS Institute and the restrictions set forth in FAR 52.227–19 Commercial Computer Software-Restricted Rights (June 1987).

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

1st printing, December 2003

2nd printing, February 2004

3rd printing, March 2004

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.

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.

Contents

PART 1 Introduction 1

Chapter 1 △ Using This Manual 3

Purpose 3

Intended Audience 3

SAS ETL Studio Online Help 4

Chapter 2 △ Introduction to SAS ETL Studio 5

Data Warehouses and Data Marts 6

The SAS Intelligence Value Chain 6

The SAS Intelligence Architecture 7

SAS ETL Studio 10

Managing with Metadata 10

Features 10

Windows 13

Wizards 19

Main Tasks 21

Usage Notes 23

PART 2 Planning 25

Chapter 3 △ Designing a Data Warehouse 27

Overview of Warehouse Design 27

Data Warehousing with SAS ETL Studio 28

Planning a Data Warehouse 29

Planning Security for a Data Warehouse 30

Chapter 4 △ Example Data Warehouse for Orion Star Sports & Outdoors 31

Orion Star Sports & Outdoors 31

Asking the Right Questions 32

Which Sales Person Is Making the Most Sales? 32

What Are the Time and Place Dependencies of Product Sales? 36

The Next Step 39

PART 3 Installation and Setup 41

Chapter 5 △ Setup Tasks for Administrators 43

Overview 43

Reviewing Project Plans for the Data Warehouse 44

Installing SAS ETL Studio and Related Software 44

Setting Up Change-Managed Metadata Repositories 45

Entering Metadata for Users and Groups	45
Entering Metadata for Servers	46
Entering Metadata for Libraries	48
Supporting Case and Special Characters in Table and Column Names	51
Prerequisites for SAS Data Quality	53
Prerequisites for Metadata Import and Export	54
Additional Information about Prerequisites	54

Chapter 6 △ Preliminary Tasks for Users 55

Overview	55
Start SAS ETL Studio	55
Create a Metadata Profile	56
Open a Metadata Profile	58
Select a Default SAS Application Server	58

PART 4 Using SAS ETL Studio 61

Chapter 7 △ Defining Sources for a Data Warehouse or Data Mart 63

Overview	64
General Tasks for Sources	64
Example: Specifying Metadata for Source Tables in SAS Format	68
Example: Extracting Information from a Flat File	74
Additional Information about Sources	84

Chapter 8 △ Defining Targets for a Data Warehouse or Data Mart 85

Overview	85
General Tasks for Targets	86
Example: Specifying Metadata for a Target Table in SAS Format	89
Additional Information about Targets	97

Chapter 9 △ Introduction to SAS ETL Studio Jobs 99

Overview	100
Main Windows for Jobs	102
General Tasks for Jobs	114
Example: Creating a SAS Code Transformation Template	122
General Tasks for SAS Code Transformation Templates	130
Additional Information about Jobs	132

Chapter 10 △ Loading Targets in a Data Warehouse or Data Mart 135

Overview	135
Example: Creating a Job That Joins Two Tables and Generates a Report	135
Example: Using a SAS Code Transformation Template in a Job	150

Chapter 11 △ Creating Cubes 159

Overview of Cubes	159
General Tasks for Cubes	160
Example: Building a Cube from a Star Schema	162

Example: Using the Source Editor to Submit User-Written Code for a Cube	169
Additional Information about Cubes	173

PART 5 Appendices 175

Appendix 1 △ Usage Notes 177

Migrating from SAS/Warehouse Administrator to SAS ETL Studio	177
ODBC Informix Library	177
SQL Join Transformation	178
Signon Scripts for SAS/CONNECT Servers	179
Preassigned Libraries in SAS ETL Studio	179
Separate Login for Each Authentication Domain	179
Access to Tables Using ODBC DB2 z/OS Pass-Through	179
SAS Source Designer Does Not Import Keys	180
Truncating Tables Does Not Free Physical Storage	180
DBMS Tables Might Be Unusable after Dropping or Re-Creating	180
Saving Metadata Changes to the Corresponding Physical Table	180
New Schema Names Must Match the Names in the DBMS	181
Usage Notes for Metadata Import/Export	181
Case and Special Characters in SAS Names	181

Appendix 2 △ Building Java Plug-ins for SAS ETL Studio 183

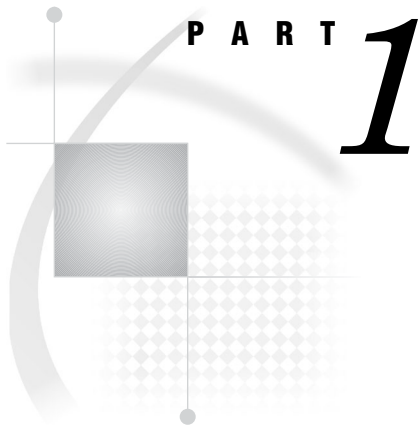
Overview	183
Shortcut Plug-ins	183
Installing a Shortcut Plug-in	184
Example: Building a Source Designer Plug-in	185

Appendix 3 △ Recommended Reading 199

Recommended Reading	199
---------------------	-----

Glossary 201

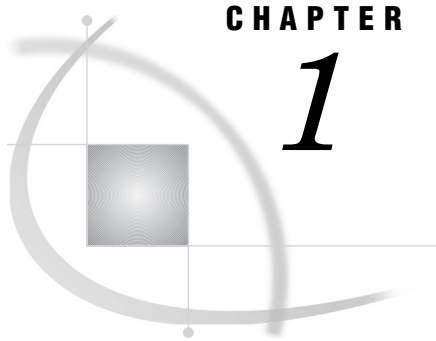
Index 205



Introduction

Chapter 1.....**Using This Manual** 3

Chapter 2.....**Introduction to SAS ETL Studio** 5



CHAPTER

1

Using This Manual

Purpose 3
Intended Audience 3
SAS ETL Studio Online Help 4

Purpose

SAS ETL Studio is an application that enables you to manage *ETL process flows*—sequences of steps for the extraction, transformation, and loading of data. This manual explains how to use SAS ETL Studio to do the following tasks:

- specify metadata for sources, such as tables in an operational system
- specify metadata for targets—the tables and other data stores 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 ETL Studio requires. A data warehouse for a fictional company, Orion Star Sports & Outdoors, is used to illustrate these tasks.

Intended Audience

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

- SAS ETL Studio user—a person who uses SAS ETL Studio to extract, transform, and load information into a data warehouse or data mart.
- SAS ETL 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 ETL Studio.

This manual is not intended for server administrators—people who install and maintain server hardware or software. Server administrators are mentioned in the manual, however, because some SAS ETL Studio tasks depend on tasks that the server administrator performs.

A common scenario for SAS ETL Studio projects is as follows:

- SAS ETL Studio users are simply told which servers to use.
- 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 *SAS Intelligence Architecture: Planning and Administration Guide*.

- A metadata administrator uses SAS Management Console to define metadata for the servers. For details about maintaining server metadata, see the online help for the Server Manager in SAS Management Console. See also the *SAS Management Console: User's Guide* and the *SAS Intelligence Architecture: Planning and Administration Guide*.

SAS ETL Studio Online Help

This manual is a companion to the online help for SAS ETL Studio. The online help describes all of the windows in SAS ETL Studio, and it summarizes the main tasks that you can perform with the software. The help includes examples for the source designer wizards, the target designer wizards, and the transformation templates in the Process Library tree.

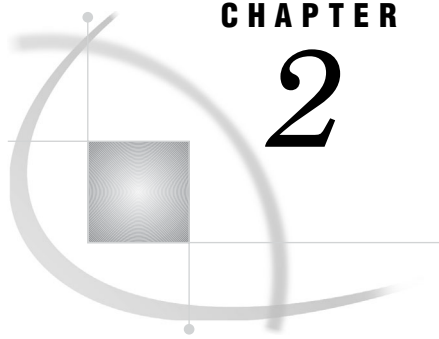
Perform the following steps to display the main help window for SAS ETL Studio.

- 1 Start SAS ETL Studio as described in “Start SAS ETL Studio” on page 55.
- 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.



CHAPTER 2

Introduction to SAS ETL Studio

<i>Data Warehouses and Data Marts</i>	6
<i>The SAS Intelligence Value Chain</i>	6
<i>The SAS Intelligence Architecture</i>	7
<i>Client Tier</i>	8
<i>Mid-Tier</i>	9
<i>Server Tier</i>	9
<i>Data Tier</i>	9
<i>SAS ETL Studio</i>	10
<i>Managing with Metadata</i>	10
<i>Features</i>	10
<i>Metadata Import and Export</i>	11
<i>Change Management Facility</i>	11
<i>Multi-Tier Support</i>	12
<i>Integrated SAS Data Quality Software</i>	12
<i>User-Written Components</i>	12
<i>Job Scheduling</i>	13
<i>Windows</i>	13
<i>Online Help for Windows</i>	14
<i>Open a Metadata Profile Window</i>	14
<i>Desktop</i>	15
<i>Menu Bar</i>	15
<i>Toolbar</i>	15
<i>Shortcut Bar</i>	16
<i>Tree View</i>	16
<i>Trees</i>	16
<i>Status Line</i>	16
<i>Message Window</i>	16
<i>Process Designer Window</i>	16
<i>Source Editor Window</i>	17
<i>Options Window</i>	18
<i>Wizards</i>	19
<i>Main Tasks</i>	21
<i>Starting SAS ETL Studio</i>	21
<i>Creating and Opening a Metadata Profile</i>	21
<i>Working with Change Management in SAS ETL Studio</i>	21
<i>Specifying Metadata for Sources</i>	22
<i>Specifying Metadata for Targets</i>	22
<i>Importing a Data Model for a Set of Sources or Targets</i>	23
<i>Creating a Job</i>	23
<i>Running the Job</i>	23
<i>Verifying a Job's Output</i>	23

Data Warehouses and Data Marts

A *data warehouse* is a collection of data that is extracted from one or more sources, is loaded into intelligent storage, and is optimized for the purpose of business intelligence and analytical intelligence queries and analyses. For example, a company might create a data warehouse that integrates sales, product, and customer information from various sources. This collection would help the company analyze how sales are affected by geography, by different promotions, by the gender and age of the customer, and by other factors.

A *data mart* is a collection of data that is optimized for a specialized set of users who have a finite set of questions and reports. For example, a shipping department could use a data mart to analyze how shipping costs are affected by geography and time of year.

Data warehouses and data marts are one link in the SAS Intelligence Value Chain.

The SAS Intelligence Value Chain

The SAS Intelligence Value Chain represents the links required to build an end-to-end business intelligence system. Each link in the chain except the planning link corresponds to a set of SAS products. This means that you can create an entire intelligence solution using SAS software. In this way, you deal with a single vendor, and most important, you can run all of your applications within a single framework.

There are five key components in the SAS Intelligence Value Chain, as shown in the following figure.



SAS ETL Studio is the main product that is associated with the *ETL^q* link, the stage in which you create a data warehouse or data mart by integrating data from existing operational data sources, such as SAS data sets, DBMS tables, and enterprise application tables. The software components in this link enable you to do the following tasks.

- *Extract* data from the data sources mentioned previously, regardless of the platform on which the data sources reside or the format of the data.
- *Transform* the data before writing it to the final target tables. For example, you might change the structure of your data by joining the contents of several tables into one table or change data values by applying expressions to existing data.
- *Load* the transformed data into target tables in a data warehouse or data mart.

The *ETL^q* stage also involves ensuring the *quality* of the data that is to be loaded into the warehouse; that is, data records must be accurate, up-to-date, and consistently represented.

SAS ETL Studio enables you to perform all of the tasks in the *ETL^q* link of the SAS Intelligence Value Chain: the extraction of data from operational data stores, the transformation of this data, and the loading of the extracted data into your data

warehouse or data mart. This application extends into the Intelligent Storage link because it enables you to design the flow of data into SAS data sets, OLAP cubes, and/or third-party relational database tables.

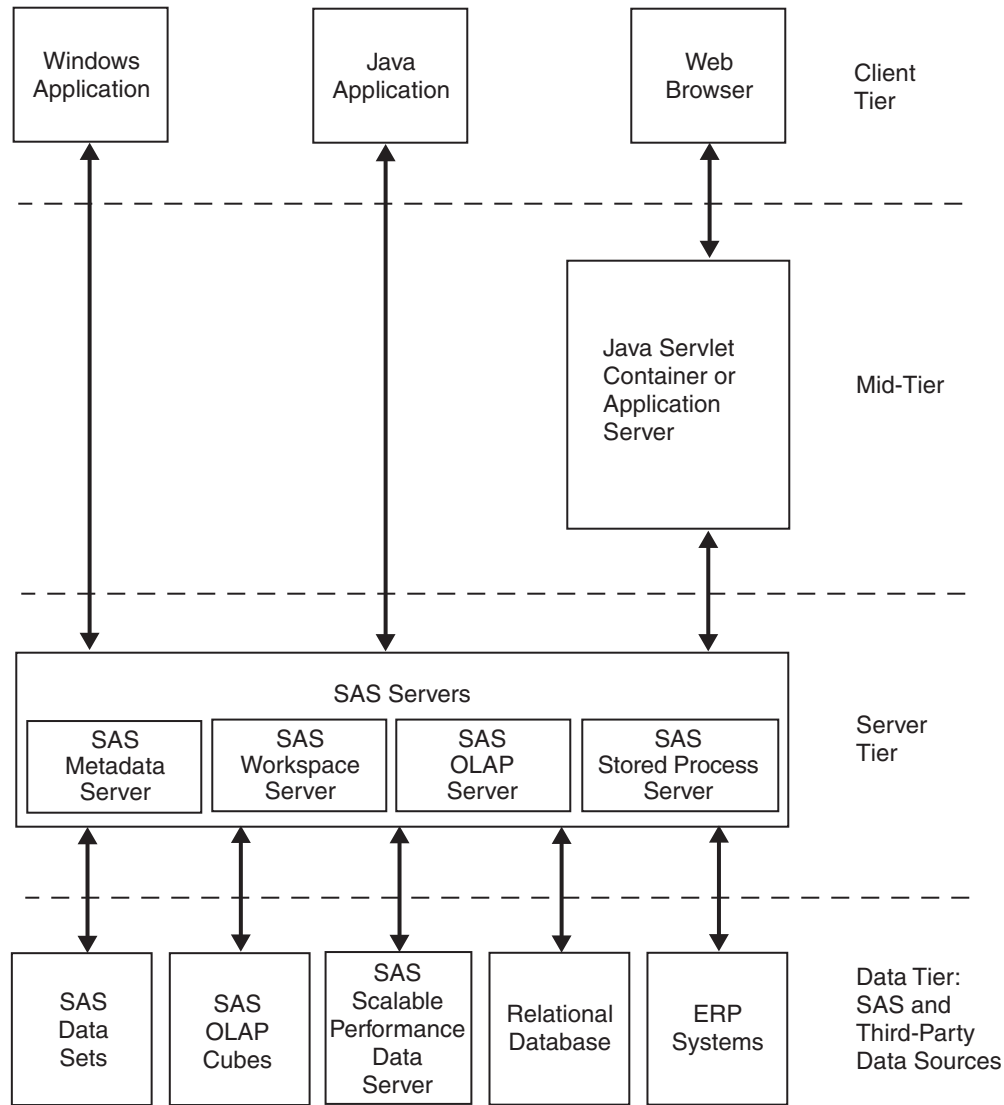
A number of products augment the capabilities of SAS ETL Studio. For example, the SAS/ACCESS interfaces to relational databases enable you to read, write, and update data regardless of its native database and platform. The SAS Data Surveyors enable you to build SAS ETL Studio jobs that help you read and write data from enterprise applications from SAP, Siebel, Oracle, and other vendors.

There are also several components that enable you to improve the quality of your data. For instance, the SAS Data Quality Server allows you to analyze, cleanse, and standardize your data. This product is often used in conjunction with products such as dfPower Studio from DataFlux Corporation, which enables you to customize the quality knowledge base that the SAS Data Quality Server uses to store its data-cleansing guidelines.

For more about the SAS Intelligence Value Chain, see the *SAS Intelligence Architecture: Planning and Administration Guide*.

The SAS Intelligence Architecture

The SAS Intelligence Architecture is one implementation of the SAS Intelligence Value Chain. The architecture can be thought of as a set of applications and data that are arranged in tiers, as shown in the following figure.

Figure 2.1 Tiers in the SAS Intelligence Architecture

The tiers in the architecture are described in the following sections.

Client Tier

Applications in the client tier provide user interfaces to the servers and data in other tiers of the architecture. As illustrated in the previous figure, there are several different kinds of clients.

The following client must be installed on Microsoft Windows systems:

- SAS Enterprise Guide

The Java applications include the following:

- SAS Management Console
- SAS ETL Studio
- SAS OLAP Cube Studio
- SAS Enterprise Miner

- other SAS Analytic Intelligence solutions.

You can run SAS Management Console on Windows systems and several UNIX platforms. The remaining applications are supported on Windows systems.

For the following applications, you only install a Web browser on each client machine:

- SAS Information Delivery Portal
- Webdoc.

These products actually run in a J2EE servlet container on the mid-tier. They communicate with the user by sending data to and receiving data from the user's Web browser. For example, an application of this type displays its user interface by sending an HTML document to the user's browser. The user can submit input to the application by sending it an HTTP response—usually by clicking a link or submitting an HTML form.

Mid-Tier

The mid-tier provides an environment where the business intelligence Web applications, such as the SAS Information Delivery Portal, can execute. The mid-tier is primarily of interest to administrators. For details about the mid-tier, see the SAS Intelligence Architecture chapter in the *SAS Intelligence Architecture: Planning and Administration Guide*.

Server Tier

The server tier consists of a set of SAS servers. Each server exposes a different set of Integrated Object Model (IOM) interfaces and has a different purpose. The servers are listed as follows:

- The SAS Metadata Server controls access to a central repository of metadata shared by all of the applications in the system. This repository contains metadata that represents SAS servers, users, libraries, tables and other resources.
- The SAS Workspace Server executes any type of SAS program.
- The SAS OLAP Server handles multidimensional expression (MDX) queries.
- The SAS Stored Process Server executes stored processes, which support input parameters.

Data Tier

SAS provides all of the products that you need to build your data tier. As mentioned earlier, SAS provides for intelligent storage by supporting the following:

- SAS data sets, which are analogous to relational database tables
- SAS SPD Engine tables, which can be read or written by multiple threads
- online analytical processing (OLAP) cubes.

SAS also provides products that enable you to access data in existing third-party database management systems (DBMSs) and Enterprise Resource Planning (ERP) systems. The SAS/ACCESS interfaces provide direct access to data in many different formats, including:

- DB2
- Informix
- MS SQL Server

- Oracle
- Sybase
- ERP systems from SAP, PeopleSoft, and Baan.

SAS ETL Studio

SAS ETL Studio is an application that enables you to manage ETL process flows—sequences of steps for the extraction, transformation, and loading of data. SAS ETL Studio enables you to do the following:

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

Managing with Metadata

Metadata is a definition or description of data. The pathname for a file is one example of metadata, and a note that explains how a particular column in a report is derived is another example.

SAS ETL Studio enables you to manage metadata for sources, targets, and the transformations that connect them. It uses metadata to generate or retrieve SAS code that reads sources and creates targets on a file system. This metadata-driven approach enables you to do the following:

- manage complex extractions, transformations and loads
- create repeatable processes
- enforce metadata source control, for team-based development of ETL process flows
- create separate environments for development, testing, and production
- share metadata with other SAS applications and with third-party software
- access metadata from enterprise applications, such as Siebel or PeopleSoft applications.

Features

SAS ETL Studio supports a number of features that enable you to manage large data warehousing projects.

Table 2.1 SAS ETL Studio Features

Feature	Related Documentation
A metadata architecture that complies with the Common Warehouse Metamodel (CWM). Enables SAS ETL Studio to share metadata with other applications.	See the SAS Open Metadata Architecture documentation on the SAS online documentation CD or on the support.sas.com web site.
Import and export of metadata in Common Warehouse Metamodel (CWM) format. Optional bridges are available for other formats. Enables SAS ETL Studio to import and export metadata about sources and targets.	See “Metadata Import and Export” on page 11.

Feature	Related Documentation
Optional Data Surveyor wizards that provide access to the metadata in enterprise applications such as PeopleSoft, SAP R/3, Siebel, and Oracle Applications.	See the online help for the Data Surveyor wizards in SAS ETL Studio, if installed.
Source control for metadata. Supports team-based development of ETL process flows.	See “Working with Change Management in SAS ETL Studio” on page 21.
Metadata access control by user and group.	See the <i>SAS Intelligence Architecture: Planning and Administration Guide</i> .
Metadata backup.	See the <i>SAS Metadata Server: Setup Guide</i> .
Multi-tier support for processes that flow across multiple servers.	See “Multi-Tier Support” on page 12.
Optional, integrated SAS data quality software for data cleansing and data analysis.	See “Integrated SAS Data Quality Software” on page 12.
Support for OLAP data stores.	See Chapter 11, “Creating Cubes,” on page 159.
Support for user-written components.	See “User-Written Components” on page 12.
Job scheduling.	See “Job Scheduling” on page 13.
Multiple work environments—separate environments for development, testing, and production, for example.	See metadata promotion and metadata replication in the <i>SAS Management Console: User’s Guide</i> .

Metadata Import and Export

The SAS Metadata Server enables you to import metadata from a variety of sources and to export it in a variety of formats. The server supports the Object Management Group’s Common Warehouse Metamodel/XML Metadata Interchange (CWM/XMI) interchange format, the industry standard for data warehouse metadata integration.

For example, you could use a data modeling tool to create a model for a set of tables, save the model in CWM format, then use the Metadata Importer wizard to import the model into a metadata repository. In SAS ETL Studio, you could view the properties of each table and verify that the appropriate metadata was imported. The tables could then be used in SAS ETL Studio jobs.

For details about the metadata import and export wizards, see the *SAS Management Console: User’s Guide*.

To import or export metadata in formats other than the CWM/XMI format, additional software from Meta Integration Technology, Inc. must be installed. Meta Integration is a SAS software partner. For information about obtaining and installing their software, see www.metaintegration.net/Products/MIMB/Description.html. You can also request an evaluation license key from this location.

Change Management Facility

SAS ETL 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. In SAS ETL Studio, the change management facility enables

multiple SAS ETL Studio users to work with the same metadata repository at the same time—without overwriting each other’s changes. For details, see “Working with Change Management in SAS ETL Studio” on page 21.

Multi-Tier Support

SAS ETL Studio provides N-tier support for processes that flow across multiple servers, as described in “The SAS Intelligence Architecture” on page 7. SAS ETL Studio uses a combination of SAS/Integration Technologies software and SAS/CONNECT software to access SAS servers.

SAS servers provide two critical services:

- Data services—access to data using SAS software, including the SAS/ACCESS products for access to DBMS data.
- Compute services—submission of SAS code to other machines running SAS and retrieval of the results.

SAS ETL Studio’s multi-tier support includes support for *implicit* data transfers using the UPLOAD/DOWNLOAD procedures, *explicit* data transfers using a Data Transfer transformation template, or both. (A Data Transfer transformation template is one of the templates that is provided in the Process Library. For an overview of the Process Library, see “Process Library Tree” on page 108.)

Support is included for scripted signon for SAS/CONNECT. SAS ETL Studio will generate script assignments in its generated code.

Integrated SAS Data Quality Software

The Process Library in SAS ETL Studio contains two data quality transformation templates: Create Match Code and Apply Lookup Standardization. These templates enable you to increase the value of your data through *data analysis* and *data cleansing*.

The prerequisites for these templates are described in “Prerequisites for SAS Data Quality” on page 53. After the prerequisites have been met, you can drag and drop the templates into process flow diagrams. “Example: Creating a Job That Joins Two Tables and Generates a Report” on page 135 illustrates the general steps for dragging and dropping transformation templates into a process flow diagram.

User-Written Components

SAS ETL Studio enables you to do the following:

- Specify user-written code for an entire job or a transformation within a job. For a summary of this task, see “Creating Jobs That Retrieve User-Written Code” on page 118.
- Drag a User-Written transformation template from the Process Library and drop it into the process flow diagram for a job. You can then update the default metadata for the transformation so that it specifies the location of user-written program. For an overview of the Process Library and its transformation templates, see “Process Library Tree” on page 108.
- Use the Transformation Generator wizard to create your own SAS code transformation templates and add them to the Process Library. After a transformation template has been added to the Process Library, you can drag and drop it into any job. For a description of this wizard, see “Transformation Generator Wizard” on page 113.

The online help for SAS ETL Studio provides additional information about working with user-written components. To display the relevant help topics, do the following:

- 1 From the SAS ETL Studio menu bar, select

Help

▶
Contents

The online help window displays.

- 2 In the left pane of the help window, select

Working with User-Written Components

▶ User-Written Components and SAS ETL Studio

You can also do the following:

- Use the Java programming language to create your own plug-ins for SAS ETL Studio. You can create Java-based transformation templates, source designer wizards, target designer wizards, and new object wizards. For details about creating your own Java plug-ins, see Appendix 2, “Building Java Plug-ins for SAS ETL Studio,” on page 183.

Job Scheduling

After you define a job in SAS ETL Studio, you can submit the job for immediate execution or deploy it for scheduling. After a job is deployed, an administrator can use SAS Management Console to schedule the job to run at specified date and time or when a specified event occurs. For details about deploying jobs and scheduling jobs, see “Jobs Can Be Scheduled” on page 102.

Windows

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

Table 2.2 SAS ETL Studio Interface

Components	Description
“Open a Metadata Profile Window” on page 14	Displays in front of the SAS ETL Studio desktop. Use to open or maintain metadata profiles. You use metadata profiles to connect to various metadata servers.
“Desktop” on page 15	Use to begin working with the metadata in the current repositories.
“Process Designer Window” on page 16	Use to create process flow diagrams, to generate and submit code for jobs, and to perform related tasks.

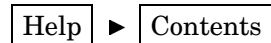
Components	Description
“Source Editor Window” on page 17	A general-purpose SAS code editor.
“Options Window” on page 18	Use to specify options for SAS ETL Studio.

Online Help for Windows

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

You can also use the table of contents to access help topics for the main windows. To display the relevant help topics, do the following:

- 1 From the SAS ETL Studio desktop, select



from the menu bar. The online help window displays.

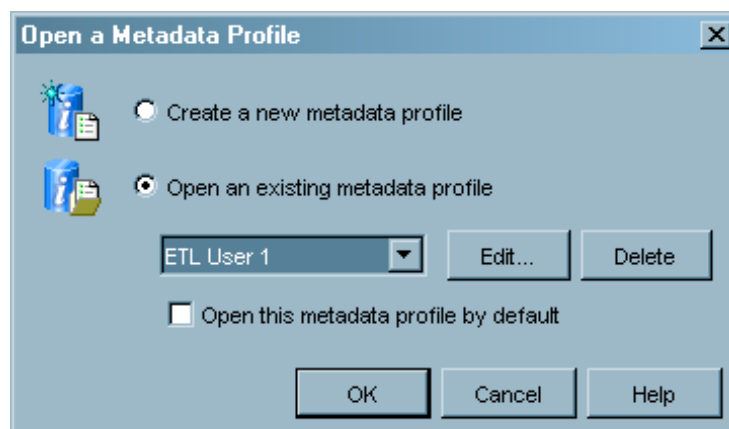
- 2 In the left pane of the help window, select the **SAS ETL Studio Desktop** folder or the **Other Main Windows** folder.
- 3 In the folder, select the desired topic.

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 ETL Studio, the Open a Metadata Profile window displays in front of the SAS ETL Studio desktop. Display 2.1 on page 14 shows an example of this window.

Display 2.1 Open a Metadata Profile Window

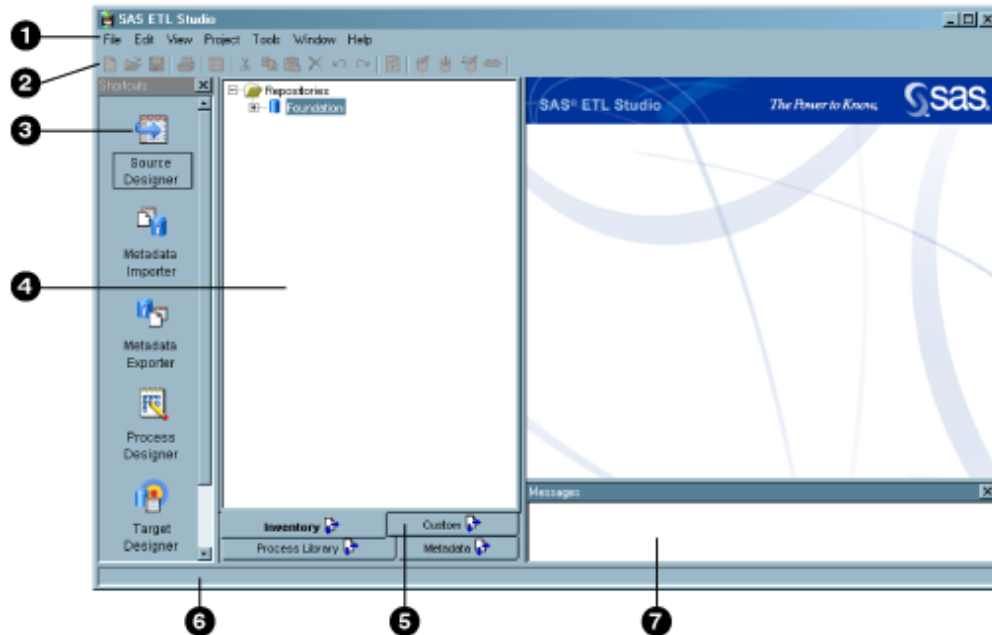


Use the Open a Metadata Profile window to open an existing metadata profile, edit an existing metadata profile, or add a new metadata profile. You must open a metadata profile before you can do any work in SAS ETL Studio. See “Creating and Opening a Metadata Profile” on page 21.

Desktop

After you open a metadata profile, the SAS ETL Studio desktop displays, as shown in Display 2.2 on page 15.

Display 2.2 SAS ETL Studio Desktop



The SAS ETL Studio desktop consists of the following components:

- 1 Menu bar
- 2 Toolbar
- 3 Shortcut bar
- 4 Tree view
- 5 Trees
- 6 Status line
- 7 Message window

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 is selected. 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 is selected. Inactive options are disabled or hidden.

Shortcut Bar

The shortcut bar displays a pane of task icons on the left side of the SAS ETL 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 the metadata that is associated with a current metadata repository. Use the tabs at the bottom of this pane, such as **Inventory** and **Custom**, to display different views or "trees" of a current repository.

Trees

Most trees display the contents of a current metadata repository in various ways. The Process Library tree can be used to drag and drop transformation templates into the process flow diagram for a job.

Status Line

The status line at the bottom of the SAS ETL Studio desktop displays error messages or other information.

Message Window

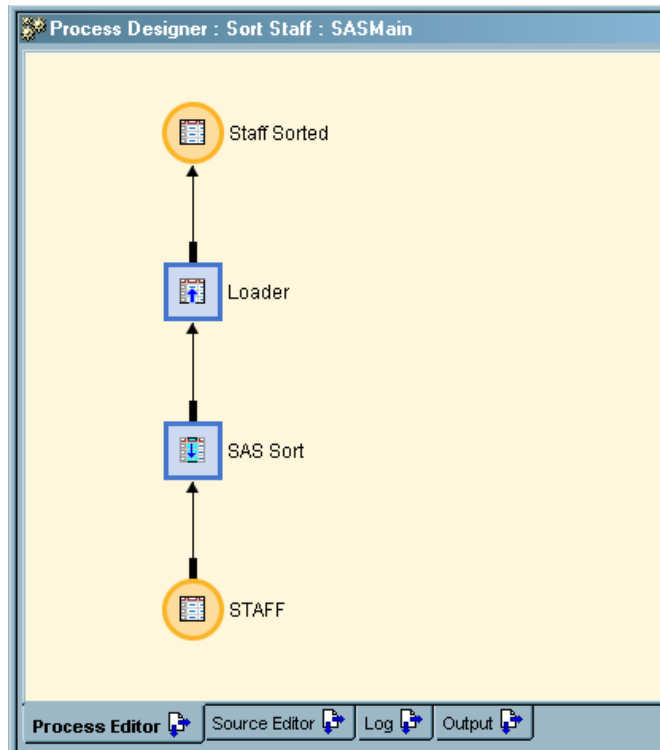
The message window display various messages. To display it, select

View ► Message Window

from the menu bar.

Process Designer Window

The Process Designer window is used to create a *process flow diagram* for the selected job, to generate and submit code for the selected job, and to perform related tasks. Display 2.3 on page 17 shows an example of this window and a process flow diagram.

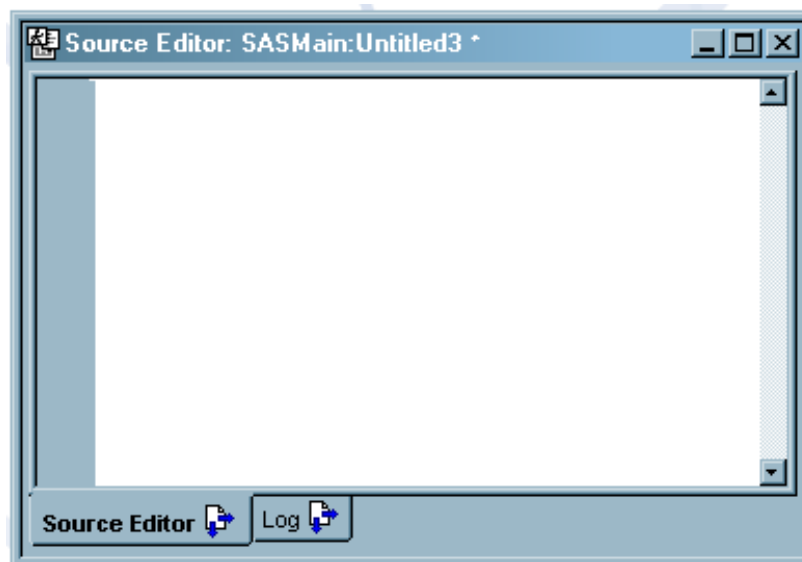
Display 2.3 Process Designer Window

For an introduction to jobs and related windows, see Chapter 9, “Introduction to SAS ETL Studio Jobs,” on page 99.

Source Editor Window

As shown in Display 2.3 on page 17, the Process Designer window includes a Source Editor tab. The Source Editor tab enables you to view and update the SAS code for a selected job.

SAS ETL Studio also provides a separate Source Editor window that you can use as a general-purpose SAS code editor. Display 2.4 on page 18 shows an example of this window.

Display 2.4 Source Editor Window

To display the Source Editor window, from the SAS ETL Studio desktop, select

Tools ► Source Editor

To submit code from the Source Editor, from the SAS ETL Studio desktop, select

Editor ► Submit

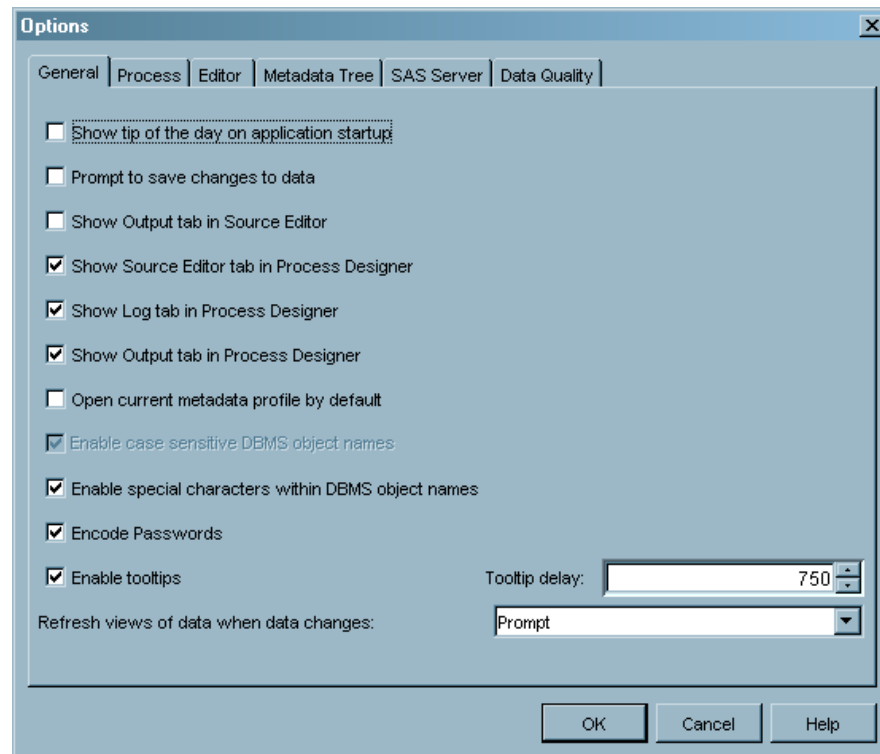
To display help for this window, press the **F1** key.

Options Window

Use the Options window to specify options for SAS ETL Studio such as:

- ☐ the default support for case and/or special characters in DBMS names
- ☐ the default SAS application server
- ☐ the default display options for the Process Designer window.

Display 2.5 on page 19 shows an example of this window.

Display 2.5 Options Window

The following steps describe one way to view or update the options on the Options window.

- 1 From the SAS ETL Studio desktop, select



to display the **Options** window.

- 2 Select the tab that contains the options that you want to view or update.

Wizards

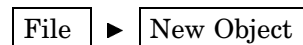
The following wizards are available from the **Shortcut Bar** or the **Tools** item on the menu bar on the SAS ETL Studio desktop:

Table 2.3 SAS ETL Studio Wizards

Wizard	Description
Source Designer wizards	Enable you to define the metadata for sources. See “Specifying Metadata for Sources” on page 22.
Target Designer wizards	Enable you to define new metadata for targets, such as tables and OLAP cubes. See “Specifying Metadata for Targets” on page 22.

Wizard	Description
Cube Designer wizard	Target designer that enables you to create a cube, a data store that supports Online Analytical Processing. See Chapter 11, “Creating Cubes,” on page 159.
New Job wizard	Enables you to select one or more tables as the target(s) (outputs) for a new job. Generates metadata for the new job. See “New Job Wizard” on page 103.
Data Surveyor wizards (optional)	If installed, provide access to the metadata in enterprise applications such as PeopleSoft, SAP R/3, Siebel, and Oracle Applications.
Metadata Importer wizard	Enables you to import metadata from other applications that support Common Warehouse Metamodel (CWM) format. Optional bridges are available for other formats. See “Metadata Import and Export” on page 11.
Metadata Exporter wizard	Enables you to export metadata to other applications that support CWM format. Optional bridges are available for other formats. See “Metadata Import and Export” on page 11.
Transformation Generator wizard	Enables you to create a user-written, SAS code transformation and make it available in the Process Library tree. One of the easiest ways to customize SAS ETL Studio. See “Transformation Generator Wizard” on page 113.

The following wizards are available from the New Object wizard selection window, which is displayed by selecting



from the SAS ETL Studio desktop. Some of these wizards are also available from the properties windows for some objects:

Table 2.4 Wizards Accessible from New Object Wizard Selection Window

Wizard	Description
Cube Designer	Enables you to define a cube for Online Analytical Processing (OLAP).
New Library wizard	Enables you to define a SAS library for SAS data or for DBMS data. See “Entering Metadata for Libraries” on page 48.
New Document wizard	Enables you to define a document that you can associate with one or more objects in a metadata repository.

Wizard	Description
New Note wizard	Enables you to define a note that you can associate with one or more objects in a metadata repository.
New Group wizard	Enables you to add a user-defined group to the Custom tree on the SAS ETL Studio desktop.

Main Tasks

After meeting the prerequisites for SAS ETL Studio as described in Chapter 6, “Preliminary Tasks for Users,” on page 55, you will be ready to create ETL process flows—sequences of steps for the extraction, transformation, and loading of data.

Starting SAS ETL Studio

See “Start SAS ETL Studio” on page 55.

Creating and Opening 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. You must open a metadata profile in order to do any work in SAS ETL Studio. For details, see “Create a Metadata Profile” on page 56 and “Open a Metadata Profile” on page 58.

Working with Change Management in SAS ETL Studio

In SAS ETL Studio, the change management facility enables multiple SAS ETL Studio users to work with the same metadata repository at the same time—without overwriting each other’s changes.

Unless your user profile includes administrative privileges, you will be working under change-management control. The following is a general description of what it is like to work under change-management control in SAS ETL Studio.

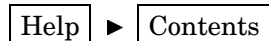
- When you open a metadata profile whose default repository is change-managed, metadata in the change-managed repository is displayed in the Inventory tree and the Custom tree. Metadata in the Project repository is displayed in the Project tree. The Project tree contains any metadata that has been checked out of the change-managed repository and any new metadata objects that have been added.
- Typically, most SAS ETL Studio users will not have the appropriate privilege to directly add or update metadata in a change-managed repository. They must check metadata objects out of and into the change-managed repository.
- To update an existing metadata object that is under metadata source control, SAS ETL Studio users will use the Inventory tree or the Custom tree to check out the object from the change-managed repository. The object will appear in the Project tree, where you can update the object’s metadata.
- After an object has been checked out by one person, it is locked so that it cannot be updated by another person until the object has been checked back in.

- You do not have to check out a library in order to add metadata about a table in that library.
- If two or more parent objects share a common object such as a table, a primary key, a note, or a document, and you check out one of these parent objects, only you will be able to check out the other parent objects that share the common object. (Other users will not be able to access the common object that you have checked out, and the shared object is required in order to check out a parent object that uses that shared object.) For example, suppose that two jobs use the same source table. If you check out Job 1, only you will be able to check out Job 2.
- When you add a new metadata object, it goes directly into the Project repository. The object will appear in the Project tree, where you can update the object's default metadata.
- Use the Fetch option to get a copy of a metadata object for testing purposes. The copied object is not checked out, so the original object is not locked. The copied object can be modified, but it cannot be checked in. Fetched items will remain in the Project repository until they are deleted.
- When you are finished working with all objects in the Project repository (and the Project tree), use the Check In feature to remove the objects from the Project repository and store them in the change-managed repository. Remember that a check-in operation checks in all of the metadata objects that are in the Project repository. You cannot check in some objects and leave other objects in the Project repository. Accordingly, you might find it convenient to work with small sets of related objects in the Project repository. For details, see *Checking In Metadata Objects*.
- To remove a metadata object from the Project repository, use the Delete option or the Undo Check Out option. To remove a metadata object from both the Project repository and the change-managed repository, use the Destroy option.

For details, about setting up the change-management facility, administrators should see “Setting Up Change-Managed Metadata Repositories” on page 45.

The online help for SAS ETL Studio provides more details about change-management. To display the relevant help topics, do the following:

- 1 From the SAS ETL Studio menu bar, select



The online help window displays.

- 2 In the left pane of the help window, select **Working with Change Management** to view the topics.

Specifying Metadata for Sources

After you have opened the appropriate metadata profile, one of your first tasks will be to specify metadata for sources. A source is a table, a view, or a file from which you will extract information. The metadata for a source typically specifies an input to a SAS ETL Studio job. For details, see Chapter 7, “Defining Sources for a Data Warehouse or Data Mart,” on page 63.

Specifying Metadata for Targets

After you have specified metadata for sources, you are ready to enter metadata for targets. A target is a table, a view, or a file that contains information that has been extracted from a source. The metadata for a target specifies an output from a SAS ETL

Studio job. For details, see Chapter 8, “Defining Targets for a Data Warehouse or Data Mart,” on page 85.

Importing a Data Model for a Set of Sources or Targets

It is possible to import a data model for a set of sources or targets using the Metadata Importer wizard from the **Shortcut Bar** or the **Tools** menu on the SAS ETL Studio desktop. The metadata for the sources or targets is added to the default metadata repository. To import or export metadata, administrators must install the appropriate software to support the metadata import and export wizards. For details, see “Prerequisites for Metadata Import and Export” on page 54.

Creating a Job

After you have specified metadata for sources and targets, you are ready to define jobs that will connect them. A job is a metadata object that specifies processes that create output. In SAS ETL Studio, each process in a job is represented by a transformation. For details, see Chapter 9, “Introduction to SAS ETL Studio Jobs,” on page 99 and Chapter 10, “Loading Targets in a Data Warehouse or Data Mart,” on page 135.

Running the Job

After you have defined the metadata for a job, you can submit the job for execution. Until you do that, the targets might not exist on the file system. For details, see “Run and Troubleshoot the Job” on page 116.

Verifying a Job’s Output

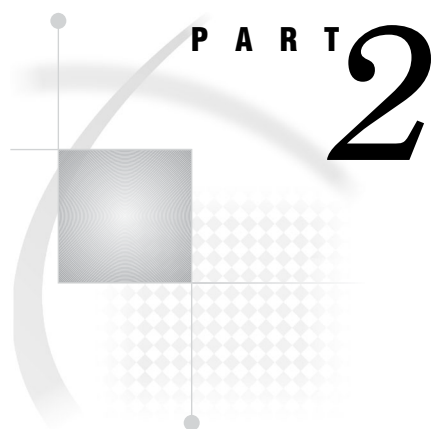
After a job has run successfully, you should verify that the output is what you were expecting. For details, see “Viewing the Data for a Source or a Target in a Job” on page 119.

Deploying a Job for Scheduling

If the appropriate software has been installed, you can deploy a SAS ETL Studio job for scheduling. After a job is deployed, an administrator can use SAS Management Console to schedule the job to run at a specified date and time or when a specified event occurs. For details, see “Jobs Can Be Scheduled” on page 102.

Usage Notes

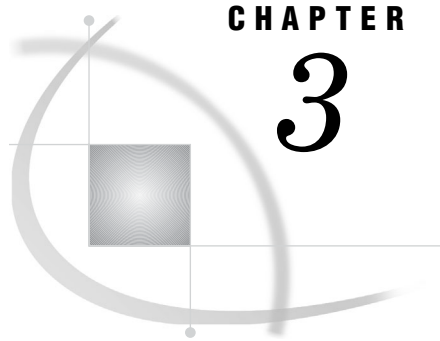
See Appendix 1, “Usage Notes,” on page 177 for usage notes that affect the current release of SAS ETL Studio.



Planning

Chapter 3.....**Designing a Data Warehouse** 27

Chapter 4.....**Example Data Warehouse for Orion Star Sports & Outdoors** 31



CHAPTER

3

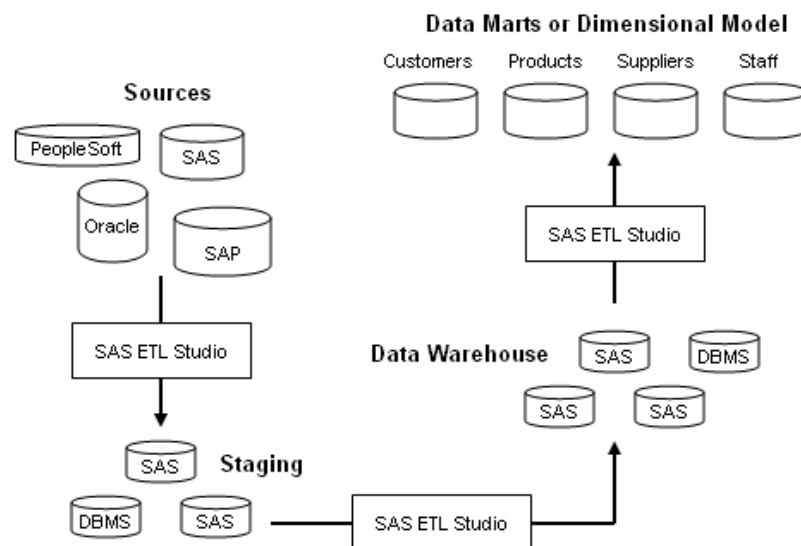
Designing a Data Warehouse

<i>Overview of Warehouse Design</i>	27
<i>Data Warehousing with SAS ETL Studio</i>	28
<i>Step 1: Extract and Denormalize Source Data</i>	28
<i>Step 2: Cleanse, Validate, and Load</i>	28
<i>Step 3: Create Data Marts or Dimensional Data</i>	29
<i>Planning a Data Warehouse</i>	29
<i>Planning Security for a Data Warehouse</i>	30

Overview of Warehouse Design

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

Figure 3.1 Best Practice Data Warehouse



In this model, SAS ETL 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 ETL 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 ETL Studio

SAS ETL 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 28).
- Cleanse extracted data and populate a central data warehouse (see “Step 2: Cleanse, Validate, and Load” on page 28).
- Create dimensional data that reflects important business needs (see “Step 3: Create Data Marts or Dimensional Data” on page 29).

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 ETL 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 ETL 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

After loading the staging area, a second set of SAS ETL 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 on 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, user, and foreign keys are correctly assigned.

When the data in the staging area is valid, SAS ETL 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 ETL 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 ETL 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 ETL 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 ETL 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 ETL Studio jobs for data cleansing in the staging area:
 - SAS ETL 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 ETL Studio's Expression Builder.

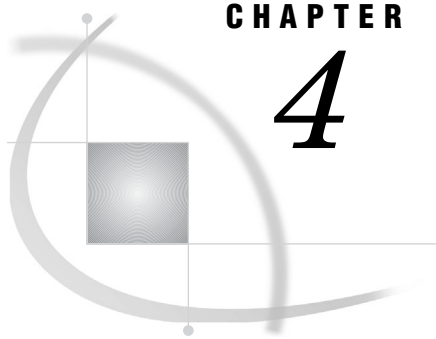
- 7 Plan and specify SAS ETL Studio jobs for data validation and load:
 - Ensure that the extracted data meets the data mode of the data warehouse before the data is loaded into the data warehouse.
 - Load data into the data warehouse at a time that is compatible with the extraction jobs that populate the data marts.
- 8 Plan and specify SAS ETL Studio jobs that populate data marts or a dimensional model out of the central data warehouse.
- 9 Plan and specify SAS ETL Studio jobs that generate reports out of the data marts or dimensional model. These jobs and all SAS ETL 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 ETL 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 ETL 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 chapters in the *SAS Intelligence Architecture: Planning and Administration Guide*.



CHAPTER

4

Example Data Warehouse for Orion Star Sports & Outdoors

<i>Orion Star Sports & Outdoors</i>	31
<i>Asking the Right Questions</i>	32
<i>Initial Questions to Be Answered</i>	32
<i>Which Sales Person Is Making the Most Sales?</i>	32
<i>Identifying Relevant Information</i>	32
<i>Identifying Sources</i>	33
<i>Source for Staff Information</i>	33
<i>Source for Organization Information</i>	33
<i>Source for Order Information</i>	34
<i>Source for Order Item Information</i>	34
<i>Source for Customer Information</i>	35
<i>Identifying Targets</i>	36
<i>Target That Combines Order Information</i>	36
<i>Target That Combines Organization Information</i>	36
<i>Target That Lists Total Sales by Employee</i>	36
<i>What Are the Time and Place Dependencies of Product Sales?</i>	36
<i>Identifying Relevant Information</i>	36
<i>Identifying Sources</i>	37
<i>Sources Related to Customers</i>	37
<i>Sources Related to Geography</i>	37
<i>Sources Related to Organization</i>	38
<i>Sources Related to Time</i>	38
<i>Identifying Targets</i>	38
<i>Target to Support OLAP</i>	38
<i>Target to Provide Input for the Cube</i>	38
<i>Target That Combines Customer Information</i>	38
<i>Target That Combines Geographic Information</i>	38
<i>Target That Combines Organization Information</i>	39
<i>Target That Combines Time Information</i>	39
<i>The Next Step</i>	39

Orion Star Sports & Outdoors

A set of data for a fictitious company, Orion Star Sports & Outdoors, is assumed to be the input for the data warehouse that is described in this manual.

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

Suppose that the executives at Orion Star Sports & Outdoors want to be more 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 questions:

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?

Initial Questions to Be Answered

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

- ☐ Which sales person 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 Sales Person Is Making the Most Sales?

Identifying Relevant Information

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

Display 4.1 Total Sales by Employee (mockup)

	NAME	SALES	ID	TITLE	COMPANY	GROUP
1	Internet/Catalog Sales	230,000	99999		Orion HQ	Internet/Catalog Sales
2	John Smith	52,000	12345	Sales Rep II	Orion HQ	Clothing
3	Jane Reynolds	125,900	33445	Sales Rep IV	Orion HQ	Equipment
5	Maria Angeles	43,780	22456	Sales Rep III	Orion Spain	Shoes
6	Jean Claude Dubois	54,020	74859	Sales Rep III	Orion France	Equipment

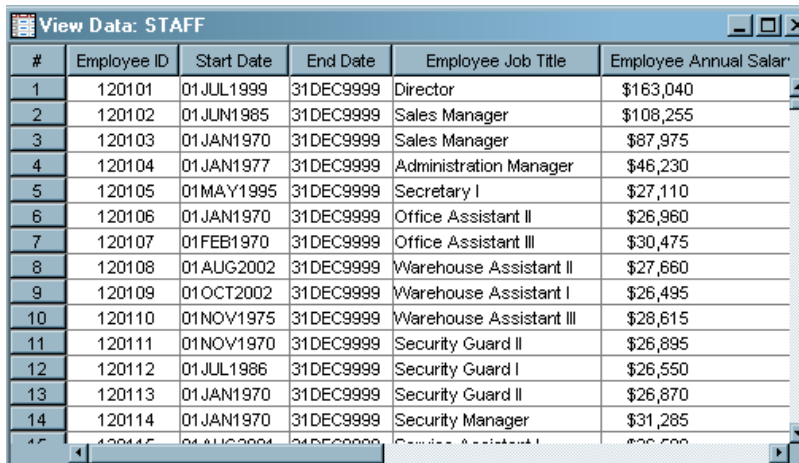
The next step is to identify how such a report could be created.

Identifying Sources

The data warehouse team examined existing tables to determine if they could be used to create the report shown in Display 4.1 on page 33. They identified a number of tables that could be used. These tables are described in the following sections.

Source for Staff Information

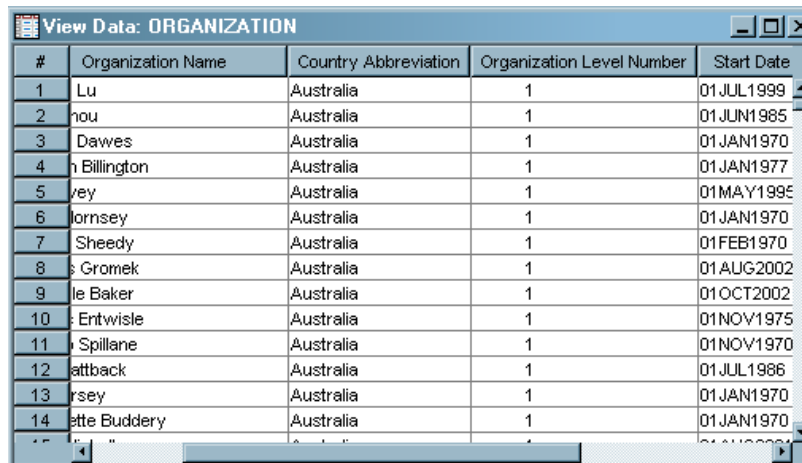
The STAFF table contains information about employees, such as name, ID, department, supervisor, and salary, as shown in the following display.

Display 4.2 The STAFF Table


#	Employee ID	Start Date	End Date	Employee Job Title	Employee Annual Salary
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
15	120115	01AUG2004	31DEC9999	Security Assistant I	\$26,500

Source for Organization Information

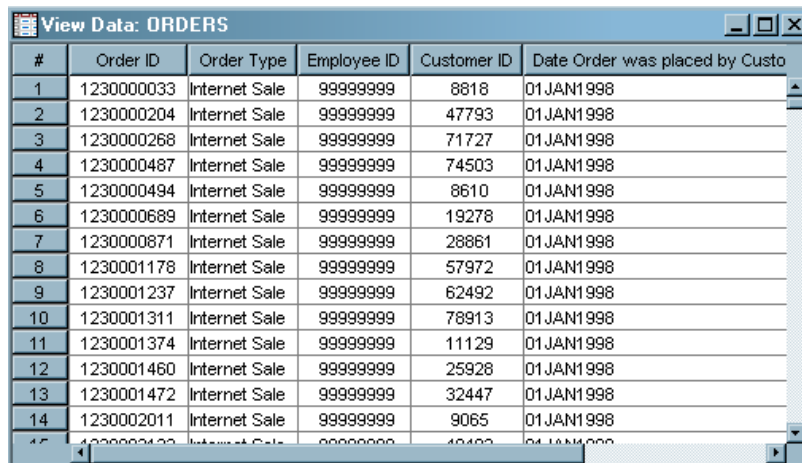
The ORGANIZATION table identifies the organization to which an employee belongs.

Display 4.3 The ORGANIZATION Table


#	Organization Name	Country Abbreviation	Organization Level Number	Start Date
1	Lu	Australia	1	01 JUL 1999
2	hou	Australia	1	01 JUN 1985
3	Dawes	Australia	1	01 JAN 1970
4	h Billington	Australia	1	01 JAN 1977
5	vey	Australia	1	01 MAY 1995
6	ornsey	Australia	1	01 JAN 1970
7	Sheedy	Australia	1	01 FEB 1970
8	s Gromek	Australia	1	01 AUG 2002
9	le Baker	Australia	1	01 OCT 2002
10	Entwisle	Australia	1	01 NOV 1975
11	Spillane	Australia	1	01 NOV 1970
12	atback	Australia	1	01 JUL 1986
13	rsey	Australia	1	01 JAN 1970
14	ette Buddery	Australia	1	01 JAN 1970

Source for Order Information

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

Display 4.4 The ORDERS Table


#	Order ID	Order Type	Employee ID	Customer ID	Date Order was placed by Custo
1	1230000033	Internet Sale	99999999	8818	01 JAN 1998
2	1230000204	Internet Sale	99999999	47793	01 JAN 1998
3	1230000268	Internet Sale	99999999	71727	01 JAN 1998
4	1230000487	Internet Sale	99999999	74503	01 JAN 1998
5	1230000494	Internet Sale	99999999	8610	01 JAN 1998
6	1230000689	Internet Sale	99999999	19278	01 JAN 1998
7	1230000871	Internet Sale	99999999	28861	01 JAN 1998
8	1230001178	Internet Sale	99999999	57972	01 JAN 1998
9	1230001237	Internet Sale	99999999	62492	01 JAN 1998
10	1230001311	Internet Sale	99999999	78913	01 JAN 1998
11	1230001374	Internet Sale	99999999	11129	01 JAN 1998
12	1230001460	Internet Sale	99999999	25928	01 JAN 1998
13	1230001472	Internet Sale	99999999	32447	01 JAN 1998
14	1230002011	Internet Sale	99999999	9065	01 JAN 1998

Source for Order Item Information

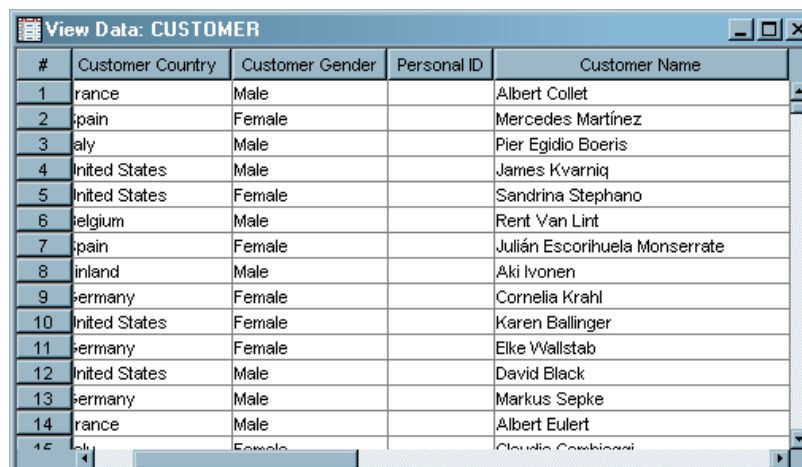
The ORDER_ITEM table contains information about orders placed with the company, and includes product ID, amount ordered, price of items, and other data.

Display 4.5 The ORDER_ITEM Table


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

Display 4.6 The CUSTOMER Table


#	Customer Country	Customer Gender	Personal ID	Customer Name
1	France	Male		Albert Collet
2	Spain	Female		Mercedes Martinez
3	Italy	Male		Pier Egidio Boeris
4	United States	Male		James Kvarniq
5	United States	Female		Sandrina Stephano
6	Belgium	Male		Rent Van Lint
7	Spain	Female		Julián Escorihuela Monserrate
8	Finland	Male		Aki Ikonen
9	Germany	Female		Cornelia Krahl
10	United States	Female		Karen Ballinger
11	Germany	Female		Elke Wallstab
12	United States	Male		David Black
13	Germany	Male		Markus Sepke
14	France	Male		Albert Eulert
15	Italy	Female		Claudia Combicani

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 sales person 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 ETL 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.
- A new table will be created that joins the STAFF and ORGANIZATION tables.
- 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.

By combining tables, the warehouse team can easily answer the specified question, as well 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. The new table will include employee name, total revenue, employee ID, job title, company, and department. It will be used to produce the report shown in Display 4.1 on page 33.

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 reported sales across a time dimension and a geographic dimension. The following display shows an example of such a report.

Display 4.7 Time and Place Dependencies for Sales

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	113751

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 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 if they could be used as inputs to an OLAP data store that would produce reports similar to the one shown in Display 4.7 on page 37. They identified a number of 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, a number of new data stores must be created, as described in the following section.

Identifying Targets

In order to support the OLAP reports such as the one shown in Display 4.7 on page 37, 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 4.7 on page 37.

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 36 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 37. 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 37. In this example, GEOGRAPHY_DIM is one dimension in a star schema.

Target That Combines Organization Information

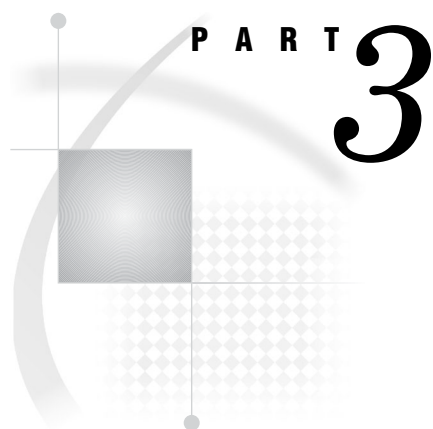
This dimension table is the same as “Target That Combines Organization Information” on page 36. In this example, ORGANIZATION_DIM is one dimension in a star schema.

Target That Combines Time Information

The TIME_DIM table will be created by joining the tables described in “Sources Related to Time” on page 38. In this example, TIME_DIM is one dimension in a star schema.

The Next Step

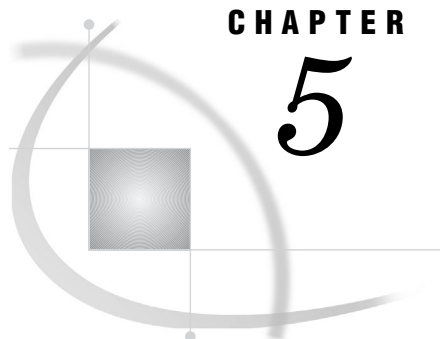
After the questions, desired outputs, sources, and targets have been specified, you can begin setting up the servers, libraries and other resources that SAS ETL Studio requires.



Installation and Setup

Chapter 5.....**Setup Tasks for Administrators** 43

Chapter 6.....**Preliminary Tasks for Users** 55



CHAPTER

5

Setup Tasks for Administrators

<i>Overview</i>	43
<i>Reviewing Project Plans for the Data Warehouse</i>	44
<i>Installing SAS ETL Studio and Related Software</i>	44
<i>Required Servers</i>	44
<i>SAS/CONNECT Server</i>	44
<i>Setting Up Change-Managed Metadata Repositories</i>	45
<i>Entering Metadata for Users and Groups</i>	45
<i>Entering Metadata for Servers</i>	46
<i>Default SAS Application Server</i>	46
<i>SAS/CONNECT Component</i>	47
<i>Impact of the Default SAS Application Server</i>	47
<i>Code Generation</i>	47
<i>Interactive Access to Data</i>	47
<i>Entering Metadata for Libraries</i>	48
<i>Which Libraries Are Needed?</i>	48
<i>Accessing SAS Tables</i>	48
<i>Accessing Custom SAS Formats</i>	49
<i>Accessing Microsoft Excel Files</i>	49
<i>Accessing External Files</i>	49
<i>Accessing DBMS Tables</i>	49
<i>Accessing Enterprise Applications</i>	50
<i>Enter Metadata for a Library</i>	50
<i>Libraries for the Example Warehouse</i>	51
<i>Additional Information about Libraries</i>	51
<i>Supporting Case and Special Characters in Table and Column Names</i>	51
<i>Case and Special Characters in SAS Table and Column Names</i>	51
<i>Case and Special Characters in DBMS Table and Column Names</i>	51
<i>Enabling DBMS Name Options for a New Database Library</i>	52
<i>Enabling DBMS Name Options for an Existing Database Library</i>	52
<i>Setting Default Name Options for Tables and Columns</i>	53
<i>Prerequisites for SAS Data Quality</i>	53
<i>Prerequisites for Metadata Import and Export</i>	54
<i>Additional Information about Prerequisites</i>	54

Overview

This chapter describes the installation and setup that must be done before users can begin work in SAS ETL Studio. These tasks are typically done by administrators rather than SAS ETL Studio users.

Reviewing Project Plans for the Data Warehouse

Review the project plans for your data warehouse. For an overview of warehouse project plans, see “Planning a Data Warehouse” on page 29 and “Planning Security for a Data Warehouse” on page 30.

Installing SAS ETL Studio and Related Software

Your data warehouse project plan should identify the SAS software that is required for your data warehouse. For example, to answer the business questions that are described in Chapter 4, “Example Data Warehouse for Orion Star Sports & Outdoors,” on page 31, the software that is listed in the following table must be installed.

Table 5.1 Software Required to Support the Example Data Warehouse

Software	Required in Order to Perform These Tasks
SAS Management Console	Administer SAS software.
SAS ETL Studio	Build and maintain ETL process flows.
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: All of the data in the example data warehouse is assumed to be local, and it is assumed to be in SAS format. Additional software would be required to read and write data that is remote or is in a format other than SAS. △

You can use the SAS Software Navigator to install SAS ETL Studio and related software. Detailed installation instructions are available from the configuration wizard that is associated with the navigator. For a detailed overview of installation and configuration, see the SAS ETL Studio chapter in the *SAS Intelligence Architecture: Planning and Administration Guide*.

Required Servers

As a client, SAS ETL Studio must connect to a SAS Metadata Server to read or write metadata. It must connect to other servers to run SAS code, to connect to a third-party database management system, or to perform other tasks. Your data warehouse project plan should identify the servers that are required for your data warehouse. Table 5.1 on page 44 lists the servers that must be installed for the example data warehouse.

SAS/CONNECT Server

SAS ETL Studio uses a SAS/CONNECT server to submit generated SAS code to machines that are remote from the default SAS application server. A SAS/CONNECT

server can also be used for interactive access to remote libraries. If your work in SAS ETL Studio requires either of these services, you should install SAS/CONNECT software.

Setting Up Change-Managed Metadata Repositories

SAS ETL 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, 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. In the example data warehouse for Orion Star Sports & Outdoors, the main metadata repository will be under *change management*. Change management enables multiple SAS ETL 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 ETL 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 ETL Studio, metadata administrators should see the SAS ETL Studio chapter in the *SAS Intelligence Architecture: Planning and 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 ETL 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.

Entering Metadata for Users and Groups

The metadata for users and groups provides the following information:

- the user and group identities on which change management and other access control methods depend
- a user ID and password when connecting to a remote machine with SAS/CONNECT
- a user ID and password when connecting to a database management system (DBMS) with SAS/ACCESS software.

Also, SAS ETL 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:

- a metadata administrator with the generic name *Metadata Admin*
- a number of SAS ETL Studio users with generic names such as *etlUser1* and *etlUser2*
- a group for SAS ETL Studio users called *ETL User Group*.

Administrators use SAS Management Console to enter metadata for persons and groups. The metadata for each person or group specifies certain privileges. For example, the metadata for *Metadata Admin* specifies administrative privileges. 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.

For details about entering metadata for users in a change-management context, see the SAS ETL Studio chapter of the *SAS Intelligence Architecture: Planning and Administration Guide*.

Entering Metadata for Servers

Metadata must be entered for all of the servers that were installed to support SAS ETL Studio. For the example data warehouse, metadata must be entered for all of the servers that are described in Table 5.1 on page 44.

Detailed instructions for entering metadata about servers are available from the configuration wizard that is associated with the SAS Software Navigator. For a detailed overview of the installation and configuration of these servers, see the SAS ETL Studio chapter in *SAS Intelligence Architecture: Planning and Administration Guide*.

Default SAS Application Server

SAS ETL Studio enables users to select a *default SAS application server*. The default SAS application server enables SAS ETL 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 a number of 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 ETL Studio. To enter metadata for SAS application servers, follow the instructions that are provided by the configuration wizard that is associated with the SAS Software Navigator.

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

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

Note that a SAS Open Metadata Server is also required for the example data warehouse. It is specified as part of the metadata profile, as described in “Create a Metadata Profile” on page 56.

SAS/CONNECT Component

SAS ETL Studio uses a SAS/CONNECT server to submit generated SAS code to machines that are remote from the default SAS application server. A SAS/CONNECT server can also be used for interactive access to remote libraries.

If your work in SAS ETL Studio requires either of these services, you might want to add a SAS/CONNECT component to your default SAS application server.

Impact of the Default SAS Application Server

In a client/server environment, the terms *local* and *remote* are relative. The same resource can be local in one context and remote in another.

Code Generation

When SAS ETL Studio generates code for a job, a resource is local or remote relative to the default SAS application server that is specified on the SAS Server tab of the Options window.

To submit a job for execution on a machine that is local to the default SAS application server, the default SAS application server must have a SAS Workspace Server component.

To submit a job to a machine that is remote from the default SAS application server, one of the following conditions must hold:

- the default SAS application server must have a SAS Workspace Server component, and
- the default SAS application server must have access to a SAS/CONNECT server on the remote machine where the job is to be executed.

Interactive Access to Data

When SAS ETL 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 unless another SAS application server is specified in the metadata for the resource.

For example, in the source designer wizard for external files, the *Host* field on the External File Selection 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. To display the contents of the table, the default SAS application server in SAS ETL Studio, or a SAS application server that is specified in the 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 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 whose metadata contains a SAS/CONNECT server component, and the SAS/CONNECT server component is accessible to the default SAS application server.

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

Entering Metadata for Libraries

In SAS software, a library is a collection of one or more files that are recognized by SAS and that are referenced and stored as a unit. SAS ETL Studio uses libraries to access the sources and targets that are referenced in SAS ETL Studio jobs. Accordingly, one of your first tasks might be to specify metadata for the libraries that contain sources, targets, or other resources.

Both SAS Management Console and SAS ETL 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 ETL 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 the need arose.

Keep in mind that 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 Chapter 7, "Defining Sources for a Data Warehouse or Data Mart," on page 63.

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, database management system (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 kind of library metadata that you need to enter.

Accessing SAS Tables

If any sources or targets are stored in SAS format, a SAS library must be defined for these tables, and metadata for each library must be added to the appropriate metadata repository.

If SAS/SHARE software will be used to provide concurrent update access to these tables, metadata for a SAS/SHARE server and a SAS/SHARE library must be added to the appropriate metadata repository.

Accessing 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 have 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 installation and configuration chapter in *SAS Intelligence Architecture: Planning and Administration Guide*.

Accessing Microsoft Excel Files

See “Accessing External Files” on page 49.

Accessing 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 an example.

External files can be accessed in at least two ways.

- External File source designer. The External File source designer is a wizard that guides you through the steps of creating and executing a SAS ETL Studio job. The job extracts information from an external file and writes it to a SAS table. Typically, the SAS table is used as a source table in another SAS ETL Studio job. The current External File source designer can extract information from flat files in fixed or delimited format. Supported file types are *.txt, *.dat, and *.csv. See “Example: Extracting Information from a Flat File” on page 74.
- ODBC library. ODBC libraries are used to read and write ODBC-compliant files as if they were SAS files. They provide interactive access to ODBC-compliant files. ODBC libraries are useful for applications that comply with ODBC, such as Microsoft Excel. The general steps for entering library metadata are described in “Enter Metadata for a Library” on page 50. In the first window of the New Library wizard, select the ODBC icon.

After an ODBC library has been created, SAS ETL Studio users can display the ODBC source designer and use it to enter metadata about the tables within the library. The general steps for using source designers are described in “Using Source Designers to Specify Metadata for Tables” on page 65.

Accessing DBMS Tables

If any sources or targets are stored in a database management system (DBMS), a library must be defined for these tables, and metadata for each library must be added to the appropriate metadata repository.

The New Library wizard is used to enter metadata for libraries, as described in “Enter Metadata for a Library” on page 50. The first window in the wizard enables you to select the following kinds of libraries that can be used to access DBMS tables:

- Database libraries. A database library uses a SAS/ACCESS interface for a specific DBMS. Database libraries can provide better performance than ODBC libraries, OLE libraries, or generic libraries.

- ODBC libraries and OLE libraries. An ODBC library uses the SAS/ACCESS interface for Open Database Connectivity (ODBC). An OLE library uses the SAS/ACCESS interface for OLE DB providers. Use these libraries when an interface for a specific DBMS is not available, and the DBMS complies with ODBC or OLE DB.
- Generic libraries. Use a generic library when no other library is suitable.

See also “Case and Special Characters in DBMS Table and Column Names” on page 51.

Accessing Enterprise Applications

Optional Data Surveyor wizards can be installed in SAS ETL Studio that provide access to the metadata in enterprise applications such as PeopleSoft, SAP R/3, Siebel, and Oracle Applications. See the documentation for these wizards for details about any libraries that must be defined to support the wizards.

Enter Metadata for a Library

The New Library wizard in SAS Management Console and SAS ETL 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 ETL 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 ETL Studio as described in “Start SAS ETL Studio” on page 55.
- 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 “Open a Metadata Profile” on page 58.
- 3 In SAS ETL 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 kind 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 kind of library for which you want to enter metadata, such as **SAS Libraries**.
- 8 Select the particular kind 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 ETL 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.

Libraries for the Example Warehouse

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

- one or more libraries for data sources in SAS format
- one or more libraries for data targets in SAS format.

You do not have to enter metadata for a library that contains SAS formats, but this library must be set up properly. See “Accessing Custom SAS Formats” on page 49.

Additional Information about Libraries

The online help for SAS ETL Studio contains additional information about libraries. To display the relevant help topics, perform the following steps:

- 1 From the SAS ETL Studio menu bar, select

Help ► Contents

The online help window displays.

- 2 In the left pane of the help window, select

Prerequisites ► Specifying Metadata for Libraries

Supporting Case and Special Characters in Table and Column Names

SAS ETL 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 ETL 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.

The source and target chapters describe one way that SAS ETL Studio users can set name options in the metadata for individual tables. For description of this task, see “Setting Name Options for Individual Tables” on page 68.

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 ETL Studio. For details, see “Setting Default Name Options for Tables and Columns” on page 53.

Case and Special Characters in DBMS Table and Column Names

SAS ETL 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 topic. Administrators could then let SAS ETL Studio users know which DBMS name options to specify in the metadata for tables in that library. The source and target chapters describe one way that SAS ETL Studio users can set name options in the metadata for DBMS tables. For description of this task, see “Setting Name Options for Individual Tables” on page 68.

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 ETL Studio. For details, see “Setting Default Name Options for Tables and Columns” on page 53.

Enabling DBMS Name Options for a New Database Library

The steps in this topic 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 general instructions in “Enter Metadata for a Library” on page 50. In the first window of 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 rest of 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 ETL Studio as described in “Start SAS ETL Studio” on page 55.
- 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 “Open a Metadata Profile” on page 58.
- 3 In SAS ETL 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 ETL Studio. These defaults apply to tables in SAS format or in database management system (DBMS) format.

Note: For details about these defaults as they relate to SAS tables, see “Case and Special Characters in SAS Names” on page 181. △

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 ETL Studio.

- 1 Start SAS ETL Studio as described in “Start SAS ETL Studio” on page 55.
- 2 Open the metadata profile that specifies the repository where metadata for the tables is stored. The steps for opening a metadata profile are described in “Open a Metadata Profile” on page 58.
- 3 From the SAS ETL 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.

Prerequisites for SAS Data Quality

The Process Library in SAS ETL Studio contains two data quality transformation templates: Create Match Code and Apply Lookup Standardization. To support these

templates, an administrator must install SAS Data Quality Server software and configure a SAS application server to access a quality knowledge base. Optional prerequisites include setting options for data quality, downloading new and updated locales, and creating schemes. For details about the SAS Data Quality Server software and the metadata for that software, administrators should see the SAS ETL Studio chapter of the *SAS Intelligence Architecture: Planning and Administration Guide*.

Prerequisites for Metadata Import and Export

SAS ETL 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 “Metadata Import and Export” on page 11.

Additional Information about Prerequisites

The online help for SAS ETL Studio provides additional information about prerequisite tasks, such as installation and setup. To display the relevant help topics, perform the following steps:

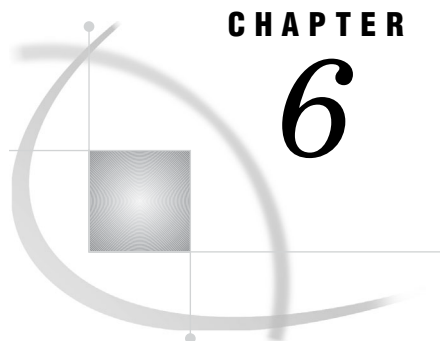
- 1 From the SAS ETL Studio menu bar, select

Help ► Contents

The online help window displays.

- 2 In the left pane of the help window, select

Prerequisites



CHAPTER

6

Preliminary Tasks for Users

<i>Overview</i>	55
<i>Start SAS ETL Studio</i>	55
<i>Specifying Java Options</i>	55
<i>Specifying the Plug-in Location</i>	56
<i>Specifying the Error Log Location</i>	56
<i>Specifying Message Logging</i>	56
<i>Create a Metadata Profile</i>	56
<i>Preparation</i>	57
<i>Default Metadata Repository</i>	57
<i>Task Summary</i>	57
<i>Open a Metadata Profile</i>	58
<i>Select a Default SAS Application Server</i>	58

Overview

After administrators complete the tasks that are described in Chapter 5, “Setup Tasks for Administrators,” on page 43, you are ready to perform some preliminary tasks that must be done before you can begin work in SAS ETL Studio.

Start SAS ETL Studio

Start SAS ETL Studio as you would any other SAS application on a given platform. For example, under Microsoft Windows, you can select

Start ► Programs ► SAS ► SAS ETL Studio

You can also start the application from a command line. Navigate to the SAS ETL Studio installation directory and issue the *etlstudio.exe* command.

If you do not specify any options, SAS ETL 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 ETL Studio, use the **--javaopts** option and enclose the Java options in single quotation marks. For example, the following command starts SAS ETL 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 ETL Studio looks for plug-ins in a **plugins** directory under the directory in which the application was installed. If you are starting SAS ETL 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 ETL Studio writes error information to a file named `errorlog.txt` in the working directory. Because each SAS ETL 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 ETL 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

Create 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 a user's login information and instructions for connecting to the metadata server automatically.

You must open a metadata profile before you can do any work in SAS ETL Studio. This topic 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 ETL 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 ETL Studio users what 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 ETL 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 ETL 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

Perform these steps to create a metadata profile:

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

Open a Metadata Profile

After a metadata profile has been created, you can open the profile in SAS ETL Studio. You must open a metadata profile in order to do any work in SAS ETL Studio. Perform these steps to open a metadata profile:

- 1 Start SAS ETL Studio. A window displays that has various options for maintaining a metadata profile.
- 2 Select **Open an existing metadata profile**. The selected profile is opened in SAS ETL Studio.

Another way to open a metadata profile is to start SAS ETL 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 in SAS ETL Studio” on page 21. 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 ETL Studio desktop.
- You can only add 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.

Select a Default SAS Application Server

One of the first tasks that users will perform in SAS ETL 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 ETL 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 46.

Perform these steps to select a default SAS application server:

- 1 From the SAS ETL Studio menu bar, select

File ► Options

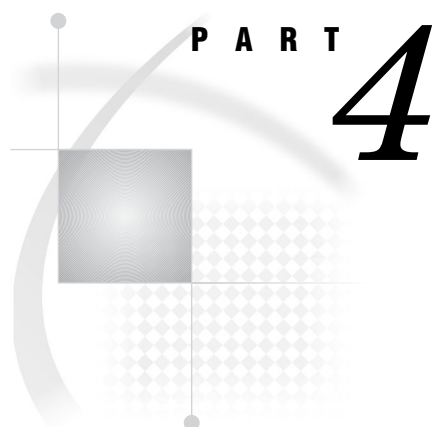
to display the Options window.

- 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.

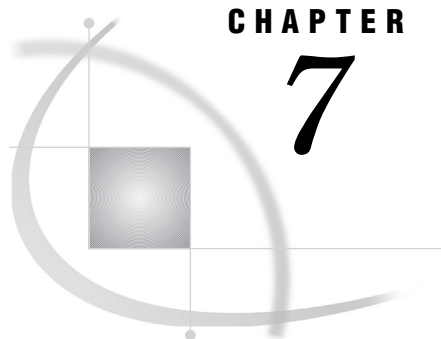
- 5 After you have verified the connection to the default SAS application server, click OK to save any changes. The server that specified in the **Server** field is now the default SAS application server.

You might want to be aware of the impact of selecting a default SAS application server. For details, see “Impact of the Default SAS Application Server” on page 47.



Using SAS ETL Studio

<i>Chapter 7</i>	Defining Sources for a Data Warehouse or Data Mart	63
<i>Chapter 8</i>	Defining Targets for a Data Warehouse or Data Mart	85
<i>Chapter 9</i>	Introduction to SAS ETL Studio Jobs	99
<i>Chapter 10</i>	Loading Targets in a Data Warehouse or Data Mart	135
<i>Chapter 11</i>	Creating Cubes	159



CHAPTER

7

Defining Sources for a Data Warehouse or Data Mart

<i>Overview</i>	64
<i>What Is a Source?</i>	64
<i>General Tasks for Sources</i>	64
<i>Prerequisite Tasks</i>	64
<i>Working Under Change Management Control</i>	64
<i>Importing a Data Model for a Set of Sources</i>	65
<i>Using Source Designers to Specify Metadata for Tables</i>	65
<i>Specifying Metadata for DBMS Tables With Keys</i>	65
<i>Extracting Information from External Files</i>	66
<i>Viewing the Data in a Source</i>	66
<i>Viewing the Metadata for a Source</i>	66
<i>Updating the Metadata for a Source</i>	67
<i>Impact of Updating a Table's Metadata</i>	67
<i>Updating Column and Mapping Metadata</i>	67
<i>Setting Name Options for Individual Tables</i>	68
<i>Prerequisites</i>	68
<i>Task Summary</i>	68
<i>Example: Specifying Metadata for Source Tables in SAS Format</i>	68
<i>Preparation</i>	68
<i>Start SAS ETL Studio and Open the Appropriate Metadata Profile</i>	69
<i>Select the Appropriate Source Designer</i>	69
<i>Select the Library That Contains the Source Tables</i>	70
<i>Select the Source Tables</i>	71
<i>Save the Source Metadata</i>	72
<i>Check In the Source Metadata</i>	73
<i>Example: Extracting Information from a Flat File</i>	74
<i>Overview</i>	75
<i>Preparation</i>	75
<i>Start SAS ETL Studio and Open the Appropriate Metadata Profile</i>	75
<i>Display the External File Source Designer</i>	76
<i>Specify How the External File Will Be Accessed</i>	77
<i>Specify How Information Should Be Imported</i>	77
<i>Specify the Width of Columns in the Target</i>	78
<i>Specify Column Variables for the Target</i>	79
<i>Specify the Location and Format of the Target</i>	80
<i>Specify a Descriptive Name for the Target</i>	81
<i>Validate the DATA Step That Will Create the Target</i>	81
<i>Create the Target</i>	82
<i>Check In the Job for the Target</i>	82
<i>Additional Information about Sources</i>	84

Overview

After you complete the tasks that are described in Chapter 6, “Preliminary Tasks for Users,” on page 55, you can specify the sources for your data warehouse or data mart.

What Is a Source?

In SAS ETL Studio, a source is a table, a view, or a file from which you will extract information. Sources can be in any format that SAS can access, on any supported hardware platform. SAS ETL Studio cannot access a source unless the metadata for that source has been defined in a repository that is available through the current metadata profile.

SAS ETL Studio has a number of source designer wizards for specific data formats. These wizards guide you through the task of specifying metadata for a source. You can then include this metadata in a job that extracts information from one or more sources and writes it to one or more targets.

A project plan should identify the sources that are required for your data warehouse or data mart. For example, the sources that are required for the Orion Star Sports & Outdoors data warehouse are listed under each business question. See the Identifying Sources section under each business question in Chapter 4, “Example Data Warehouse for Orion Star Sports & Outdoors,” on page 31.

General Tasks for Sources

Prerequisite Tasks

Defining metadata for sources will be easier if the following prerequisites have been met:

- ☐ administrators have performed the tasks that are described in Chapter 5, “Setup Tasks for Administrators,” on page 43
- ☐ metadata has been entered for any required libraries, such as any libraries that contain sources, as described in “Entering Metadata for Libraries” on page 48.

Working Under Change Management Control

Unless your user profile includes administrative privileges, you will be working under change management control. For a general description of how change management affects user tasks in SAS ETL Studio, see “Working with Change Management in SAS ETL Studio” on page 21.

When working with sources, the main impacts of change management are as follows:

- ☐ To update an existing source table, you must check out the source table.
- ☐ Metadata for new sources is added to the Project tree. At some point, you must check in new objects to the change-managed repository.

You do not have to check out a library in order to add metadata about source tables or target tables in that library.







Importing a Data Model for a Set of Sources

You can use the Metadata Import wizard to import a data model for a set of sources. You can display the wizard from the Shortcuts Bar or the Tools menu on the SAS ETL Studio desktop.

If the model to be imported is not in Common Warehouse Metamodel (CWM) format, administrators must install optional bridge software from Meta Integration Technology, Inc. For details, see “Metadata Import and Export” on page 11.

Using Source Designers to Specify Metadata for Tables

A source designer is a type of wizard that enables you to enter metadata for one or more sources. The general steps for using a source designer are as follows:

- 1 If the source will be accessed with library, verify that metadata for the library is available. This step is the same whether the source is in SAS format or in most other formats. At some sites, administrators might define all libraries and simply tell SAS ETL Studio users which libraries to use. For details, see “Entering Metadata for Libraries” on page 48.
- 2 From the SAS ETL Studio desktop, select
 ► 
from the menu bar. The Source Designer selection window is displayed.
- 3 Select the appropriate source designer and click .
- 4 Select the library that contains the source tables and click .
- 5 Select one or more source tables and click .
- 6 Review the metadata and click .

For an example of how a source designer can be used, see “Example: Specifying Metadata for Source Tables in SAS Format” on page 68. For details about writing your own source designer, see Appendix 2, “Building Java Plug-ins for SAS ETL Studio,” on page 183.


Specifying Metadata for 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 of these defined. 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.

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.

Extracting Information from 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 an example.

External files can be accessed in at least two ways:

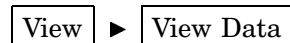
- External File source designer. The External File source designer is a wizard that guides you through the steps of creating and executing a SAS ETL Studio job. The job extracts information from an external file and writes it to a SAS table. Typically, the SAS table is used as the input in another SAS ETL Studio job. The current External File source designer can extract information from flat files in fixed or delimited format. Supported file types are *.txt, *.dat, and *.csv. See “Example: Extracting Information from a Flat File” on page 74.
- ODBC library . ODBC libraries are used to read and write ODBC-compliant files as if they were SAS files. They provide interactive access to ODBC-compliant files. ODBC libraries are useful for applications that comply with ODBC, such as Microsoft Excel. At some sites, administrators might define all libraries and simply tell SAS ETL Studio users which libraries to use. The general steps for entering library metadata are described in “Enter Metadata for a Library” on page 50. In the first window of the New Library wizard, select the ODBC icon.

After an ODBC library has been created, SAS ETL Studio users can display the ODBC source designer and use it to enter metadata about the tables within the library. The general steps for using source designers are described in “Using Source Designers to Specify Metadata for Tables” on page 65.

Viewing the Data in a Source

After the metadata for a source table has been entered, you might want to verify that the corresponding physical table contains the data that you were expecting. Perform the following steps to view the data that corresponds to the metadata for a source:

- 1 On the SAS ETL Studio desktop, select the Inventory tree.
- 2 In the Inventory tree, open the folder that contains the metadata for the source, such as the **Tables** folder.
- 3 Select the source table, then select



from the menu bar. The View Data window displays the column headings, row numbers, and the rows of data in the source. If the column headings are ordered and named as expected, then the metadata for the source is correct.

Viewing the Metadata for a Source

To view the metadata for a source, perform the following steps:

- 1 On the SAS ETL Studio desktop, select the Inventory tree.

- 2 In the Inventory tree, open the **Tables** folder.
- 3 Select the metadata for the source, then select

File ► **Properties**

from the menu bar. The properties window for the table is displayed.

- 4 Use the tabs in this window to view metadata for the table. Each tab has its own Help button.

Updating the Metadata for a Source

Perform the following steps to update the metadata for a source that is under change management.

- 1 On the SAS ETL Studio desktop, select the Inventory tree.
- 2 In the Inventory tree, open the folder that contains the metadata for the source table, such as the **Tables** folder or the **External Tables** folder.
- 3 Select the source, then select

Project ► **Check Out**

The metadata for the source is checked out. A check mark is displayed next to the source in the Inventory tree. An icon indicating a checked-out table appears in the Project tree.

- 4 Display the Project tree, select the source, and select

File ► **Properties**

from the menu bar. The properties window for the source is displayed.

Note that you must display the source from the Project tree in order to update metadata. Displaying the source from the Inventory tree enables browsing only.

- 5 Use the tabs in this window to make changes to the metadata for the source. Each tab has its own Help button. Column metadata is a special case. For details, see “Updating Column and Mapping Metadata” on page 67.
- 6 When you are finished updating the metadata, you can check in your changes. In the Project tree, select the repository icon.
- 7 From the menu bar on the SAS ETL Studio desktop, select

Project ► **Check In Repository**

Impact of Updating a Table's Metadata

Keep in mind that a table, such as a source table, can be used in multiple jobs. A table can also be used in multiple places in the same job. Accordingly, when you update the metadata for a table, make sure that the updates are appropriate in all contexts where the metadata is used. For example, if you update the columns for Table 1 in one job, the updates would also have to be appropriate for Table 1 in the context of another job.

Updating Column and Mapping Metadata

If the metadata for a source has not yet been added to a job, you can update its column metadata as previously described. If the metadata for a source has been added

to a job, the job might have one or more targets and transformations that depend on the current column metadata for the source. In that case, use the steps that are described in “Updating Column and Mapping Metadata” on page 121.

Setting Name Options for Individual Tables

SAS ETL 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 database management system (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 “Case and Special Characters in DBMS Table and Column Names” on page 51. Name options do not have 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 Start SAS ETL Studio, open the appropriate metadata profile, and check out the table(s) whose metadata must be updated. For details about these steps, see “Updating the Metadata for a Source” on page 67.
- 2 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.
- 3 In the properties window, click the **Physical Storage** tab.
- 4 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 Names” on page 181.
- 5 Click **OK** to save your changes.
- 6 Check in the changed metadata, as described in “Updating the Metadata for a Source” on page 67.

Example: Specifying Metadata for Source Tables in SAS Format

This example demonstrates how to use a source designer to enter metadata for several source tables in SAS format. This example is based on some source tables that are needed for the example data warehouse, as described in “Which Sales Person Is Making the Most Sales?” on page 32.

Preparation

For the current example, assume that the following statements are true.

- ❑ The CUSTOMER table, ORDERS table, and ORDER_ITEM table are currently existing operational tables. They contain information that is needed for a data warehousing project.
- ❑ All of the tables are in SAS format and are stored in a SAS library called Ordetail.
- ❑ Metadata for the Ordetail library has been added to the main metadata repository for the example data warehouse. For details about libraries, see “Entering Metadata for Libraries” on page 48.
- ❑ The main metadata repository is under change-management control. For details about change management, see “Working with Change Management in SAS ETL Studio” on page 21.
- ❑ You have selected a default SAS application server for SAS ETL Studio, as described in “Select a Default SAS Application Server” on page 58.

Start SAS ETL Studio and Open the Appropriate Metadata Profile

Perform the following steps to begin work in SAS ETL Studio:

- 1 Start SAS ETL Studio as described in “Start SAS ETL Studio” on page 55.
- 2 Open the appropriate metadata profile as described in “Open a Metadata Profile” on page 58. For this example, the appropriate metadata profile would specify the project repository that will enable you to access metadata about the Ordetail library.

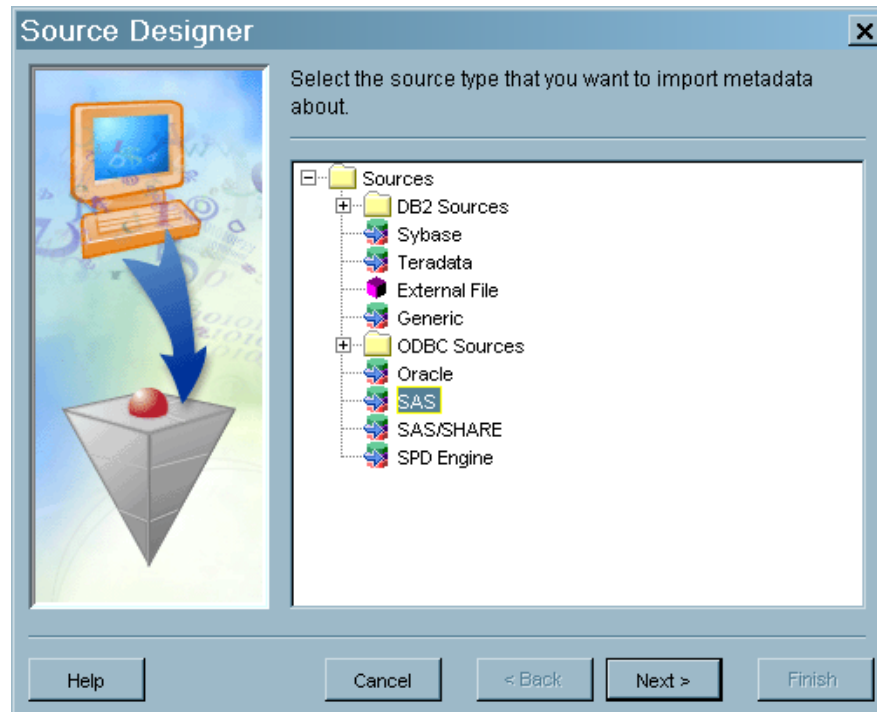
You do not have to check out a library in order to add metadata about source tables or target tables in that library. Accordingly, the next task is to select the appropriate source designer.

Select the Appropriate Source Designer

To select the wizard that enables you to enter metadata for a SAS table, from the menu bar on the SAS ETL Studio desktop, select

Tools ► Source Designer

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

Display 7.1 Source Designer Selection Window

The list of available wizards on your machine might be somewhat different from the list shown in Display 7.1 on page 70. Only those source types for which source designer wizards have been installed are available. From this window, take the following actions:

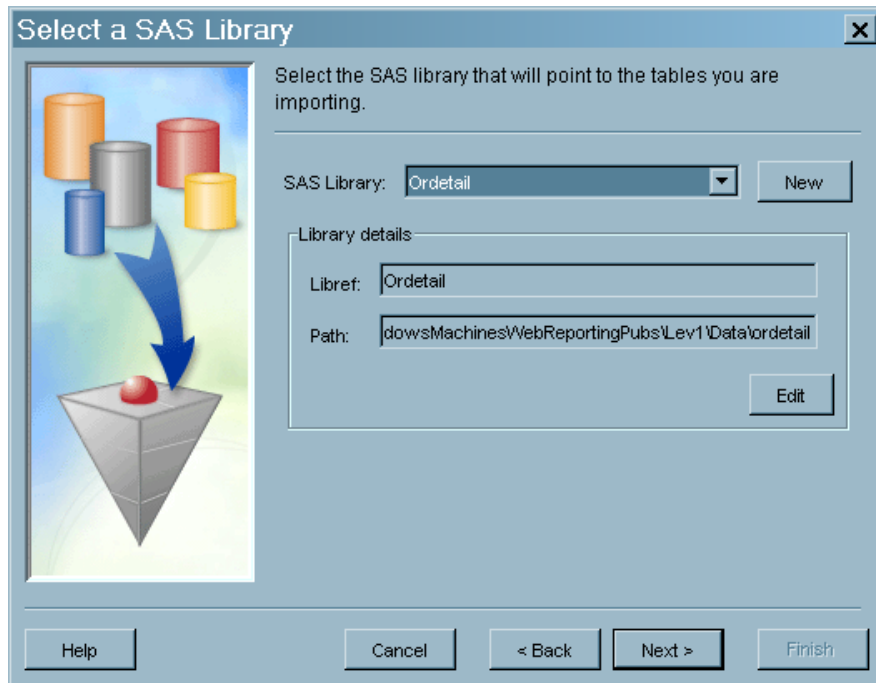
- 1 Click the **SAS** icon.
- 2 Click Next.

The wizard will attempt 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 source tables.

Select the Library That Contains the Source 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 desired source table(s). For the current example, you would select the Ordetail library, as shown in the following display.

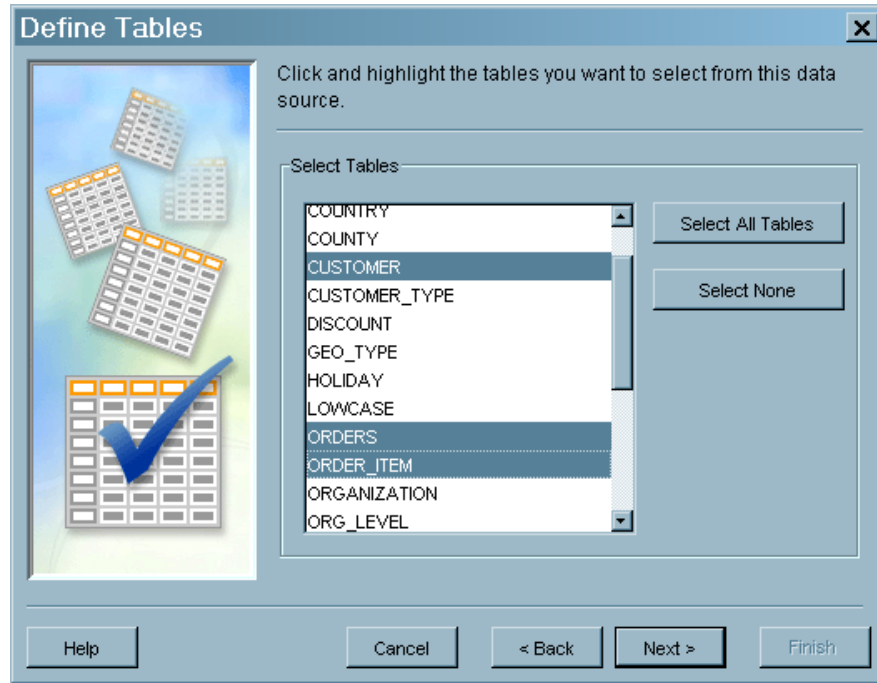
Display 7.2 Select a SAS Library Window

After selecting the appropriate library, click **Next**. The SAS application server is used to access the library, and the Define Tables window is displayed.

The next task is to select the source tables.

Select the Source Tables

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

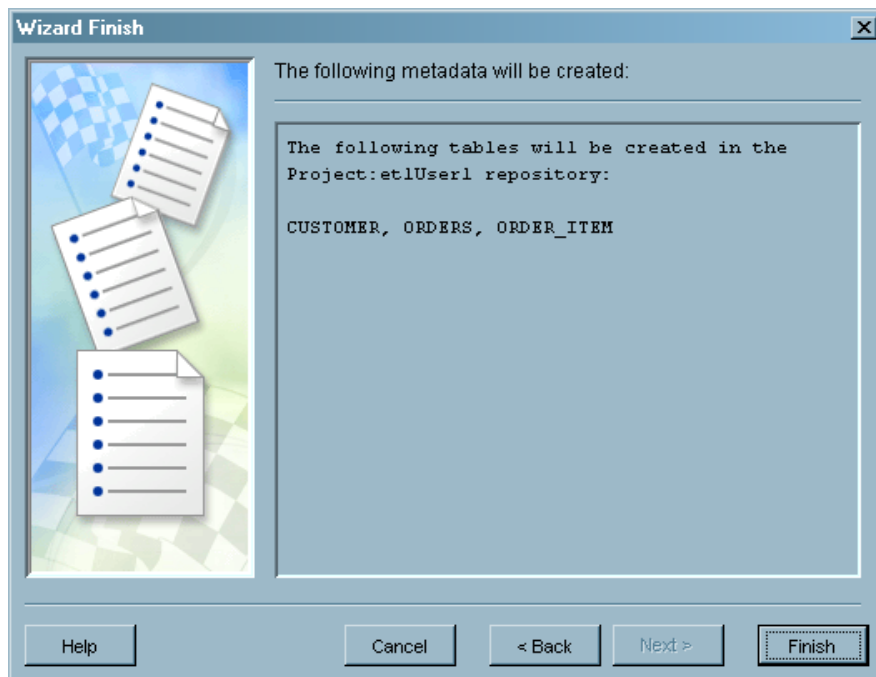
Display 7.3 Define Tables Window

In this example, we want to create metadata objects for CUSTOMER, ORDERS, and ORDER_ITEM. Accordingly, select these tables and click **[Next]**. The Wizard Finish window is displayed.

The next task is to review and save the metadata for source tables.

Save the Source Metadata

After you select the source table(s), use the Wizard Finish window to review the metadata that you have entered.

Display 7.4 Wizard Finish Window

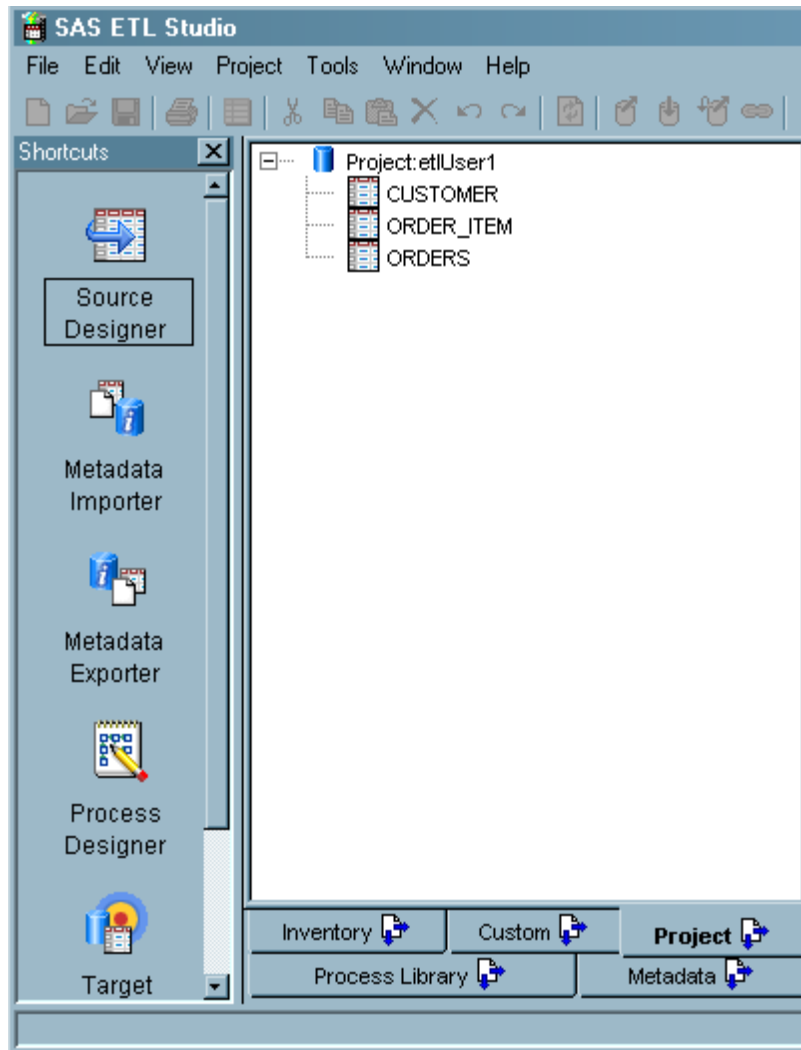
Review the text in the metadata pane on the Wizard Finish window. The text shown in the previous display means that the metadata for three tables called CUSTOMER, ORDERS, and ORDER_ITEM will be saved to the project repository *Project: etlUser1*.

When you are satisfied that the metadata is correct, click **Finish**. The metadata for CUSTOMER, ORDERS, and ORDER_ITEM will be saved to the project repository (visible in the Project tree on the SAS ETL Studio desktop).

The next task is to check in the metadata for source tables.

Check In the Source Metadata

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

Display 7.5 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: etlUser1*).
- 2 From the menu bar on the SAS ETL Studio desktop, select

Project ► Check In Repository

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

Example: Extracting Information from a Flat File

This example demonstrates how to use the External File wizard to extract information from a flat file.

Overview

As noted in “Extracting Information from External Files” on page 66, the External File source designer is a wizard that guides you through the steps of creating and executing a SAS ETL Studio job. The job extracts information from an external file and writes it to a SAS table. Typically, the SAS table is used as a source table in another SAS ETL Studio job.

The External File source designer enables you to do the following tasks:

- extract information from flat files in fixed or delimited format. Supported file types are *.txt, *.dat, and *.csv.
- import column-aligned data or data that is not column-aligned. Data that is not column-aligned can be imported with single or multiple delimiters separating the values.
- import variable length records and fixed-length records.
- import character, numeric and nonstandard numeric data (such as currency data or signed numbers).
- specify how missing values should be treated.
- read 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 in the imported data; arrange the order of the columns, change attributes of any column, add new columns.

For column-aligned data, the External File source designer uses 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.

Preparation

For the current example, assume that the following statements are true:

- A data warehouse project plan specified a report that requires information from an external file. The external file is a flat file that is called `employeeFlatFile.dat`.
- Information will be extracted from `employeeFlatFile.dat` into a SAS table called `EmployeeSAS`.
- `EmployeeSAS` will be stored in a SAS library called `Efiout`. Assume that metadata for `Efiout` has been added to the main metadata repository for the example data warehouse. For details about libraries, see “Entering Metadata for Libraries” on page 48.
- The main metadata repository is under change management control. For details about change management, see “Working with Change Management in SAS ETL Studio” on page 21.
- You have selected a default SAS application server for SAS ETL Studio, as described in “Select a Default SAS Application Server” on page 58.

Start SAS ETL Studio and Open the Appropriate Metadata Profile

Perform the following steps to begin work in SAS ETL Studio:

- 1 Start SAS ETL Studio as described in “Start SAS ETL Studio” on page 55.

- 2 Open the appropriate metadata profile as described in “Open a Metadata Profile” on page 58. For this example, the appropriate metadata profile would specify the project repository that will enable you to access metadata for the Efiout library.

You do not have to check out a library in order to add metadata about source tables or target tables in that library. Accordingly, the next task is to display the External File source designer.

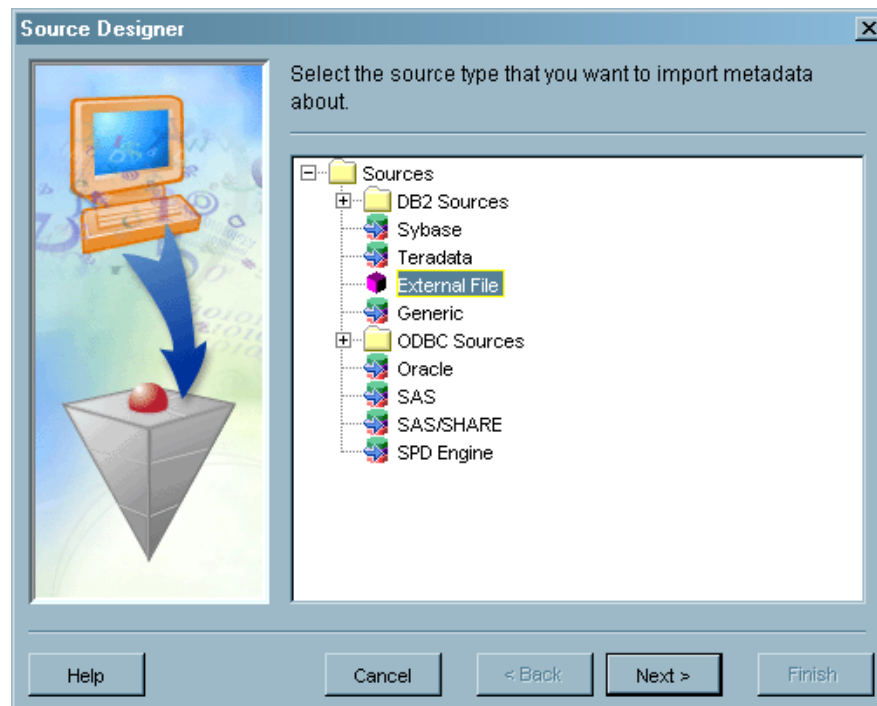
Display the External File Source Designer

To display the External File source designer, from the menu bar on the SAS ETL Studio desktop, select

Tools ► Source Designer

The Source Designer selection window is displayed, as shown in the following display:

Display 7.6 Source Designer Selection Window



From this window, take the following actions:

- 1 Click the **External File** icon.
- 2 Click Next.

The wizard will attempt 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, the first window of the External File source designer is displayed: the External File Selection window.

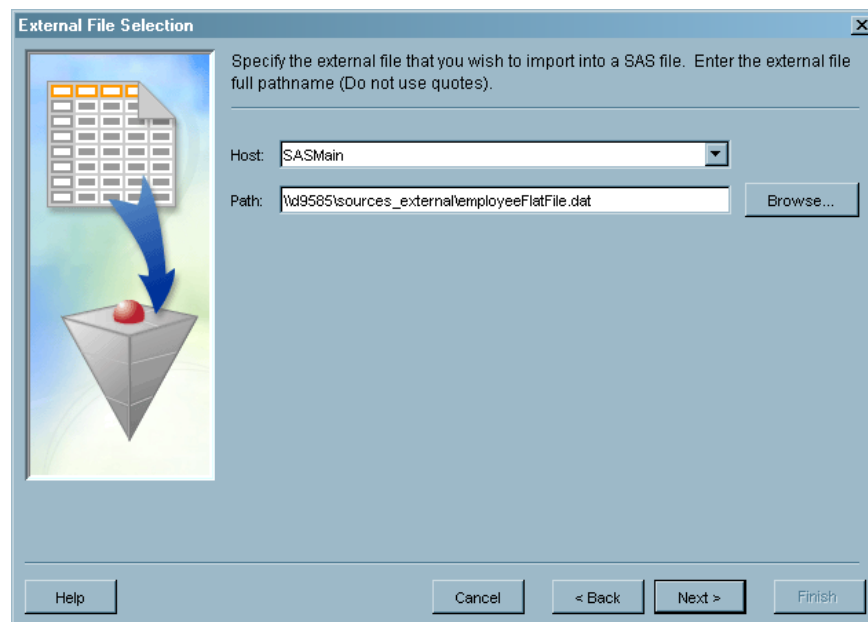
Specify How the External File Will Be Accessed

Perform the following steps to specify how the external file will be accessed:

- 1 In the External File Selection window, select the SAS application server that will be used to access the external file, then specify a physical path to the external file.

The external file is probably remote from the SAS application server, so you will probably have to enter a remote path in the **Path** field, such as `\\d9585\sources_external\employeeFlatFile.dat`. The following display shows an External File Selection window with values that are appropriate for the current example.

Display 7.7 External File Selection Window

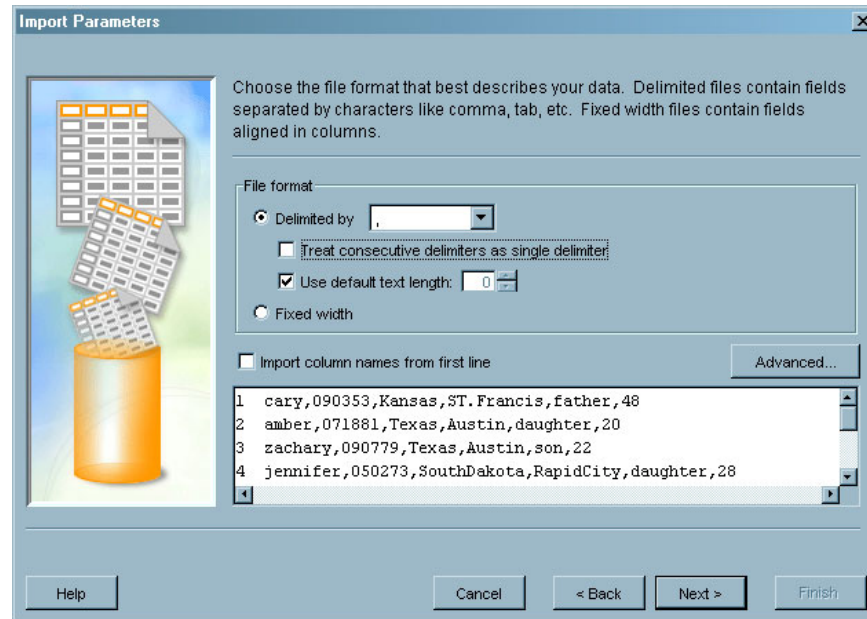


- 2 When the appropriate server and path have been specified, click **Next**. The wizard reads the source file and tries to determine whether the source contains fixed-width data or delimited data. The Import Parameters window is displayed with some estimated parameters.

Specify How Information Should Be Imported

Perform the following steps to specify how information should be imported from the external file:

- 1 Review the estimated parameters and sample data that are displayed in the Import Parameters window. Update as needed. The following display shows an Import Parameters window with values that are appropriate for the current example.

Display 7.8 Import Parameters Window

- When the import parameters are correct, click **Next**. The wizard reads the source file and derives default metadata for columns in the target (the SAS table), based on columns in the source (the external file).

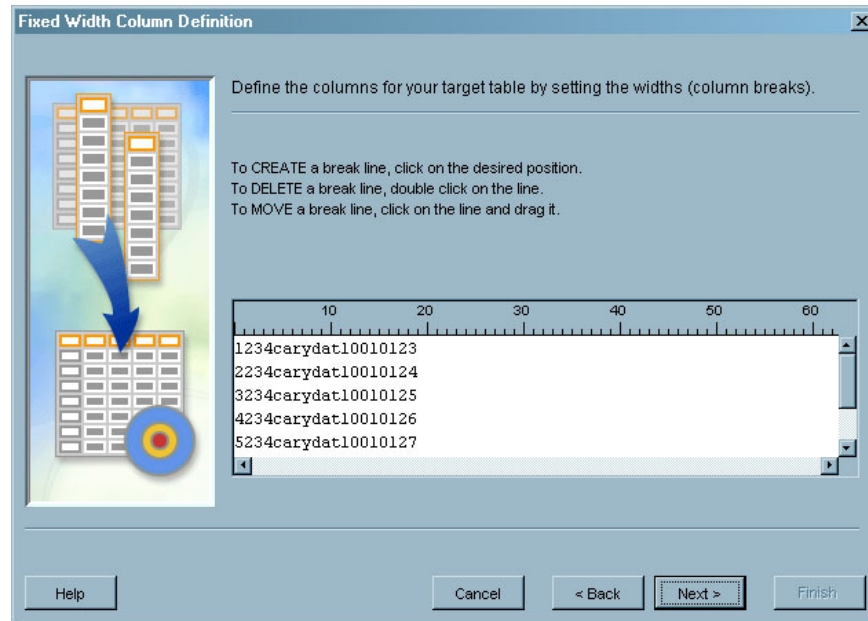
For the current example, assume that the data in `employeeFlatFile.dat` is arranged in columns, and the Set Column Definitions window is displayed. The next task is described in “Specify Column Variables for the Target” on page 79.

However, if the data in the external file is not arranged in columns, the Fixed Width Column Definition window is displayed. In this scenario, the next task is described in “Specify the Width of Columns in the Target” on page 78.

Specify the Width of Columns in the Target

If the data in the external file is not arranged in columns, use the Fixed Width Column Definition window to view the data in the source (external file) and specify the width of the columns in the target (SAS table).

- To specify the width of a column, study the example data, decide where the columns should be, then click the location where the column should be. An arrow is added at each column location, as shown in the following display.

Display 7.9 Fixed Width Column Definition Window

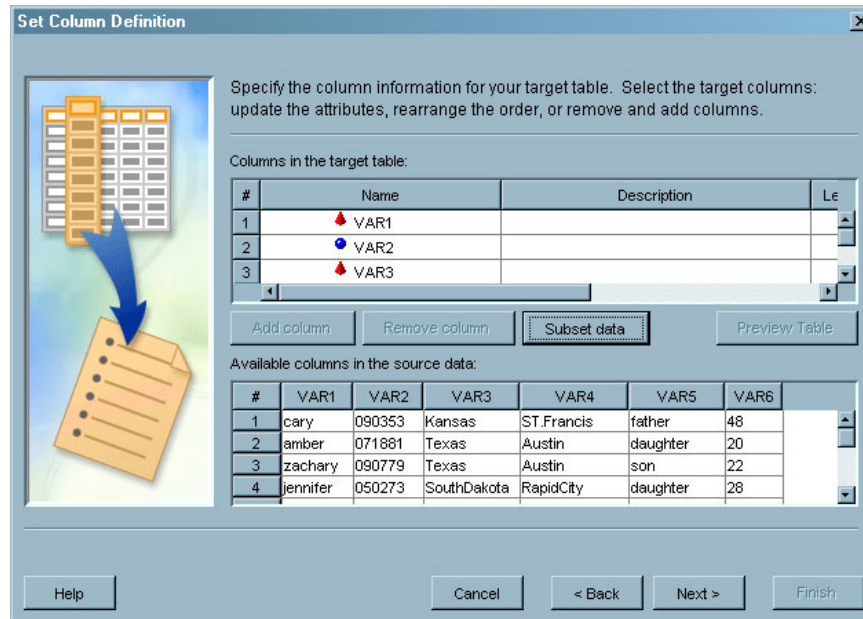
Note: The values and columns in the previous display do not match the data in the file `employeeFlatFile.dat`. They are taken from a different external file, one that does not arrange its data in columns. △

- 2 After the appropriate columns have been specified, click **Next**. A temporary SAS data set is created with the column widths that you have specified on the Fixed Width Column Definition window. The Set Column Definition window is displayed, showing the effect of any changes that you made on the Fixed Width Column Definition window.

Specify Column Variables for the Target

- 1 In the Set Column Definition window, you can accept the default column variable names in the Columns in the target group box, or you can update them.

Scroll to the right to view or update the **Description**, **Length**, **Type**, **Format**, and **Informat** fields. The following display shows a Set Column Definition window with values that are appropriate for the current example.

Display 7.10 Set Column Definition Window


Specify the column information for your target table. Select the target columns: update the attributes, rearrange the order, or remove and add columns.

Columns in the target table:

#	Name	Description	Le
1	VAR1		
2	VAR2		
3	VAR3		

Buttons: Add column, Remove column, **Subset data**, Preview Table

Available columns in the source data:

#	VAR1	VAR2	VAR3	VAR4	VAR5	VAR6
1	cary	090353	Kansas	ST.Francis	father	48
2	amber	071881	Texas	Austin	daughter	20
3	zachary	090779	Texas	Austin	son	22
4	jennifer	050273	SouthDakota	RapidCity	daughter	28

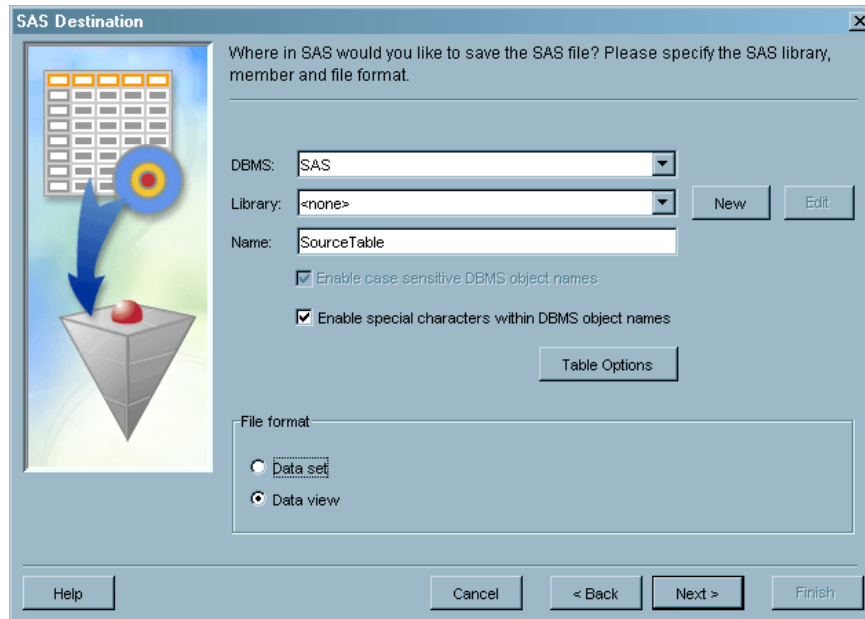
Buttons: Help, Cancel, < Back, Next >, Finish

The **Subset Data** button in this window launches the Expression Builder window. In the context of the External File source designer, the Expression Builder enables you to build a WHERE clause to subset the data that is being imported from an external file. (To see an example of how the Expression Builder can be used to build a WHERE clause, see “Configure the SQL Join Transformation” on page 141.)

- When the column metadata is correct, click **Next**. The SAS Destination window is displayed.

Specify the Location and Format of the Target

When the SAS Destination window is displayed, a number of fields have default values that must be updated. The following display shows the SAS Destination window before you have specified the desired library, member name (table name), and file format for the target (SAS dataset or SAS view).

Display 7.11 SAS Destination Window

- 1 In the SAS Destination window, select the library where the target will be stored (Efiout), a member name for the target (EmployeeSAS), and the file format of the target (SAS dataset). The name for the target must follow the rules for SAS names.
- 2 When the physical storage information is correct, Click **Next**. The General Properties window for the target is displayed.

Specify a Descriptive Name for the Target

Perform the following steps to specify a descriptive name for the target:

- 1 In the General Properties window, specify a descriptive name for the target, and perhaps a brief narrative description.

The default descriptive name for the target is the member name that was entered in the SAS Destination window. A descriptive name does not have the same restrictions as a member name, so it can be changed to something that is easier to understand. For the current example, assume that the SAS dataset name that you entered in the previous window (EmployeeSAS) is acceptable as a descriptive name.

- 2 When the general properties are correct, click **Next**. The Import Data Step Validation window is displayed.

Validate the DATA Step That Will Create the Target

In the Import Data Step Validation window, click **Next** to generate a SAS DATA step from the metadata that you have entered. If the DATA step has no errors, the Wizard Finish window displays. If the DATA step has errors, a window displays that enables you to view the SAS log and take other corrective action.

For this example, assume that the DATA step is valid, and the Wizard Finish window is displayed.

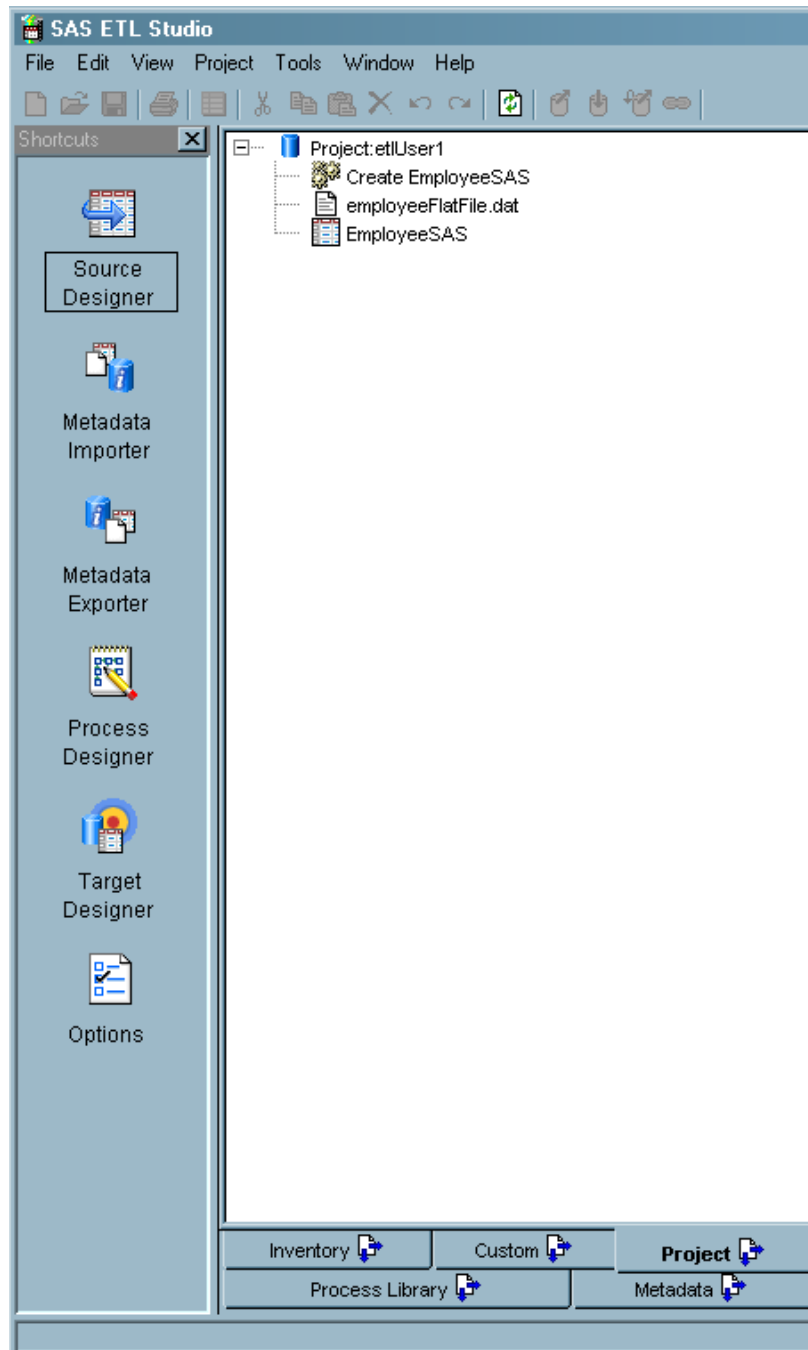
Create the Target

In the Wizard Finish window, review the metadata that you have entered. When you are satisfied that the metadata is correct, click **Finish**. The following actions occur:

- Metadata for the source (the external file) is added to a current metadata repository.
- Metadata for the target (the SAS table) is added to a current metadata repository.
- Metadata for the job that extracts information from the source and writes it to the target is added to a current metadata repository.
- The job is submitted for execution.
- If the job is successful, the target is created on the file system.

Check In the Job for the Target

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

Display 7.12 Project Tree with Output from the External File Source Designer

In the previous display:

- Create EmployeeSAS is the metadata for the job. Jobs that are created by wizards have names in the format *Create target_name*, where *target_name* is the name of the target.
- employeeFlatFile.dat is the metadata for the external file. It will have the same name as the external file.
- EmployeeSAS is the metadata for the SAS table into which information was extracted from the external file. The target has the descriptive name that you specified in the External File wizard.

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

- 1 In the Project tree, select the repository icon (*Project: etlUser1*).
- 2 From the menu bar on the SAS ETL Studio desktop, select

Project ► Check In Repository

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

Additional Information about Sources

The online help for SAS ETL Studio provides additional information about how to enter metadata for sources, including DBMS sources and external files. Perform the following steps to display the relevant help topics:

- 1 Start SAS ETL Studio.
- 2 From the menu bar, select

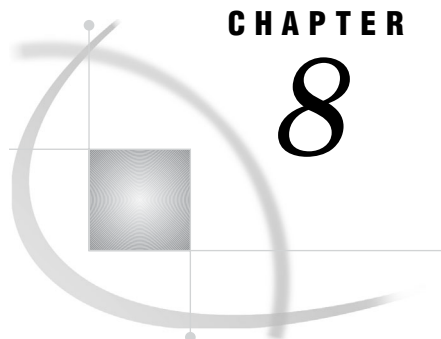
Help ► Contents

The online help window is displayed.

- 3 In the left pane of the help window, select

Working with Sources ► Specifying Metadata for Sources

to view the topic.



CHAPTER

8

Defining Targets for a Data Warehouse or Data Mart

<i>Overview</i>	85
<i>What Is a Target?</i>	85
<i>General Tasks for Targets</i>	86
<i>Prerequisite Tasks</i>	86
<i>Working under Change Management Control</i>	86
<i>Importing a Data Model for a Set of Targets</i>	86
<i>Using the Target Table Designer to Specify Metadata for Tables</i>	87
<i>Using the Cube Designer</i>	87
<i>Viewing the Data in a Target</i>	87
<i>Viewing the Metadata for a Target</i>	88
<i>Updating the Metadata for a Target Table</i>	88
<i>Impact of Updating a Table's Metadata</i>	89
<i>Updating Column and Mapping Metadata</i>	89
<i>Setting Name Options for Individual Tables</i>	89
<i>Example: Specifying Metadata for a Target Table in SAS Format</i>	89
<i>Preparation</i>	89
<i>Start SAS ETL Studio and Open a Metadata Profile</i>	90
<i>Select the Target Designer</i>	90
<i>Enter a Name and Description</i>	91
<i>Select Column Metadata from Existing Tables</i>	92
<i>Specify Column Metadata for the Target</i>	93
<i>Specify Physical Storage Information for the Target</i>	94
<i>Usage Hints for the Physical Storage Window</i>	94
<i>Save Metadata for the Target</i>	95
<i>Check In the Target Metadata</i>	95
<i>Additional Information about Targets</i>	97

Overview

After you have specified the sources for your data warehouse or data mart, you can specify the targets—the tables or other data stores in your warehouse or data mart.

What Is a Target?

In SAS ETL Studio, a target is a table, a view, or a file that receives output from a job. Targets can be in any format that SAS can access, on any supported hardware platform. SAS ETL Studio cannot access a target unless the metadata for that target has been defined in a repository that is available through the current metadata profile.

SAS ETL Studio has a number of target designer wizards for specific data formats. These wizards guide you through the task of specifying metadata for a target. You can then include the target in a job that extracts information from one or more sources and writes it to one or more targets.

Note: When you define the metadata object for a target, you are defining a set of instructions that are used to create the target. In some cases, the target might not exist on the file system until you create and run the job that writes data to the target. △

A project plan should identify the targets in your data warehouse or data mart. For example, the targets that are required for the Orion Star Sports & Outdoors data warehouse are listed under each business question. See the section named Identifying Targets under each business question in Chapter 4, “Example Data Warehouse for Orion Star Sports & Outdoors,” on page 31.

General Tasks for Targets

Prerequisite Tasks

Defining metadata for targets will be easier if the following prerequisites have been met:

- metadata has already been entered for the sources that are inputs to the target, as described in Chapter 7, “Defining Sources for a Data Warehouse or Data Mart,” on page 63
- metadata has been entered for any required libraries, such as any libraries that contain sources or targets, as described in “Entering Metadata for Libraries” on page 48.

Working under Change Management Control

Unless your user profile includes administrative privileges, you will be working under change management control. For a general description of how change management affects user tasks in SAS ETL Studio, see “Working with Change Management in SAS ETL Studio” on page 21.

When working with targets, the main impacts of change management are as follows:

- 1 To update an existing target, you must first check out that target.
- 2 Metadata for new targets is added to the Project tree. New targets are not displayed in the Inventory tree until you check them in.
- 3 Checking out a job checks out all of the sources and targets in that job.

Importing a Data Model for a Set of Targets

You can use the Metadata Import wizard to import a data model for a set of targets. You can display the wizard from the Shortcuts Bar or the Tools menu on the SAS ETL Studio desktop.

If the model to be imported is not in Common Warehouse Metamodel (CWM) format, administrators must install optional bridge software from Meta Integration Technology, Inc. For details, see “Metadata Import and Export” on page 11.

Using the Target Table Designer to Specify Metadata for Tables

The Target Table Designer is one of the standard target designer wizards in SAS ETL Studio. It is used to enter metadata for SAS tables and DBMS tables that will be outputs in SAS ETL Studio jobs. The general steps for using the Target Table Designer are as follows.

- 1 If the target will be accessed with library, verify that metadata for the library is available. This step is the same whether the target is in SAS format or in most other formats. At some sites, administrators might define all libraries and simply tell SAS ETL Studio users which libraries to use. For details, see “Entering Metadata for Libraries” on page 48.
- 2 From the SAS ETL Studio desktop, select

Tools

▶

Target Designer

from the menu bar.
- 3 In the Target Designer selection window, select **Target Table** and click Next.
- 4 In the first window, enter a name and description for the metadata object that will specify the target table and click Next.
- 5 In the import columns window, you can select column metadata from a table that has been defined in a current metadata repository. Import columns only if you want your target columns to be similar to the columns in tables that are already defined. Select column metadata as appropriate and click Next.
- 6 In the target columns window, accept or modify any imported metadata for columns. Add new column metadata for the target as appropriate. Click Next.
- 7 In the physical storage window, specify a data format for the target (SAS format or a DBMS format), the library where the target will be stored, and a valid name for the target. Click Next.
- 8 In the finish window, review the metadata that you have entered. When you are satisfied that the metadata is correct, click Finish. The metadata for the target is written to the current metadata repository.

For an example of how to use the Target Table Designer, see “Example: Specifying Metadata for a Target Table in SAS Format” on page 89. For details about writing your own target designer, see Appendix 2, “Building Java Plug-ins for SAS ETL Studio,” on page 183.

Using the Cube Designer

The Cube Designer is a target designer that is used to enter metadata for SAS cubes. Cubes are described in Chapter 11, “Creating Cubes,” on page 159.

Viewing the Data in a Target

When you create a new target with a target designer wizard, the metadata exists for that target, but the physical table generally does not exist. To view the contents of the physical table that corresponds to the metadata for a target, you probably need to run the job that includes that target, as described in the following steps:

- 1 Create and run the job that contains the new target, as described in Chapter 10, “Loading Targets in a Data Warehouse or Data Mart,” on page 135.

- 2 After running the job, display the Project tree.
- 3 In the Project tree, select the target, then select

View

▶ View Data

from the menu bar. The View Data window displays the column headings, row numbers, and the rows of data in the target. If the column headings are ordered and named as expected, then the metadata for the target is correct.

To view the data in existing targets, display the Inventory tree, select a table, and select

View

▶ View Data

You cannot view the contents of a cube in SAS ETL Studio. You can use Microsoft Excel or SAS Enterprise Guide to view the data in a cube. For details, see the *SAS OLAP Server: Administrator's Guide*.

Viewing the Metadata for a Target

To view the metadata for a target, perform the following steps:

- 1 On the SAS ETL Studio desktop, select the Inventory tree.
- 2 In the Inventory tree, open the **Tables** folder (for target tables) or the **OLAP** folder (for cubes).
- 3 Select a target, then select

File

▶ Properties

from the menu bar. The properties window for the target is displayed.

- 4 Use the tabs in the properties window to view metadata for the table. Each tab has its own Help button.

Updating the Metadata for a Target Table

Perform the following steps to update the metadata for a target table that is under change management. (To update the metadata for cubes, see “Updating a Cube or Its Metadata” on page 161.)

- 1 On the SAS ETL Studio desktop, display the Inventory tree.
- 2 In the Inventory tree, open the **Tables** folder.
- 3 Select the target table to be updated, 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 Display the Project tree, select the table, and select

File

▶ Properties

from the menu bar. The properties window for the table is displayed.

Note that you must display the table from the Project tree in order to update metadata. Displaying the target from the Inventory tree enables browsing only.

- 5 Use the tabs in the properties window to make changes to the metadata for the target table. Each tab has its own Help button. Column metadata is a special case. For details, see “Updating Column and Mapping Metadata” on page 89.
- 6 When you are finished updating the metadata, check in your changes. In the Project tree, select the repository icon.
- 7 From the menu bar on the SAS ETL Studio desktop, select

Project ► Check In Repository

Impact of Updating a Table's Metadata

Keep in mind that a table, such as a target table, can be used in multiple jobs. A table can also be used in multiple places in the same job. Accordingly, when you update the metadata for a table, make sure that the updates are appropriate in all contexts where the metadata is used. For example, if you update the columns for Table 1 in one job, the updates would also have to be appropriate for Table 1 in the context of another job.

Updating Column and Mapping Metadata

If the metadata for a target has not yet been added to a job, you can update its column metadata as previously described. If the metadata for a target has been added to a job, the job might have one or more transformations that depend on the current column metadata for the target. In that case, use the steps that are described in “Updating Column and Mapping Metadata” on page 121.

Setting Name Options for Individual Tables

Same as “Setting Name Options for Individual Tables” on page 68.

Example: Specifying Metadata for a Target Table in SAS Format

This example demonstrates how to use a target designer to enter metadata for a target table in SAS format. The example is based on a target table that is needed for the example data warehouse, as described in “Which Sales Person Is Making the Most Sales?” on page 32. The table in this example will subsequently serve as the target in the SAS ETL Studio job that is described in “Example: Creating a Job That Joins Two Tables and Generates a Report” on page 135.

Preparation

For the current example, assume that the following statements are true:

- A project plan identified the need for a new target table called *Total_Sales_By_Employee*. The new table will be created by joining two other tables, ORDER_FACT and ORGANIZATION_DIM. The new table will include employee name, total revenue, employee ID, job title, company, and department.
- The target table is in SAS format and will be stored in a SAS library called Ordetail.
- Metadata for the Ordetail library has been added to the main metadata repository for the example data warehouse. For details about libraries, see “Entering Metadata for Libraries” on page 48.

- The main metadata repository is under change management control. For details about change management, see “Working with Change Management in SAS ETL Studio” on page 21.
- You have selected a default SAS application server for SAS ETL Studio, as described in “Select a Default SAS Application Server” on page 58.

Start SAS ETL Studio and Open a Metadata Profile

Perform the following steps to begin work in SAS ETL Studio:

- 1 Start SAS ETL Studio as described in “Start SAS ETL Studio” on page 55.
- 2 Open the appropriate metadata profile as described in “Open a Metadata Profile” on page 58. For this example, the appropriate metadata profile has access to metadata about the Ordetail library.

You do not have to check out a library in order to add target metadata. Accordingly, the next task is to select the appropriate target designer.

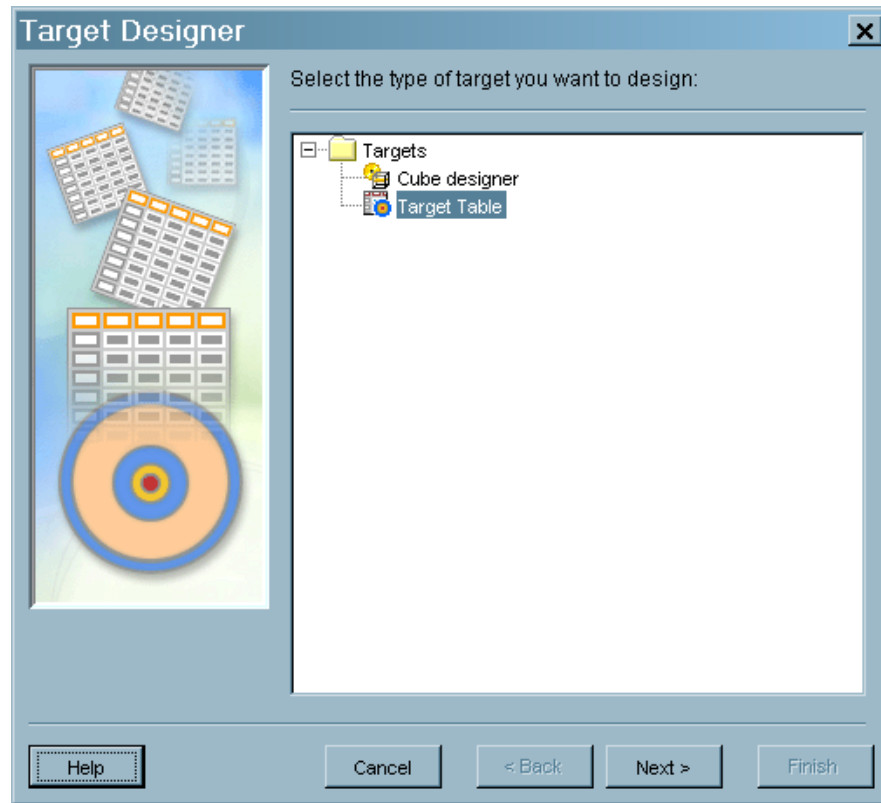
Select the Target Designer

Follow these steps to select the wizard that enables you to enter metadata for a target in SAS format:

- 1 From the menu bar on the SAS ETL Studio desktop, select

Tools ► Target Designer

The Target Designer selection window is displayed as follows. Note that the list of available target designers might differ at your site.

Display 8.1 Target Designer Selection Window

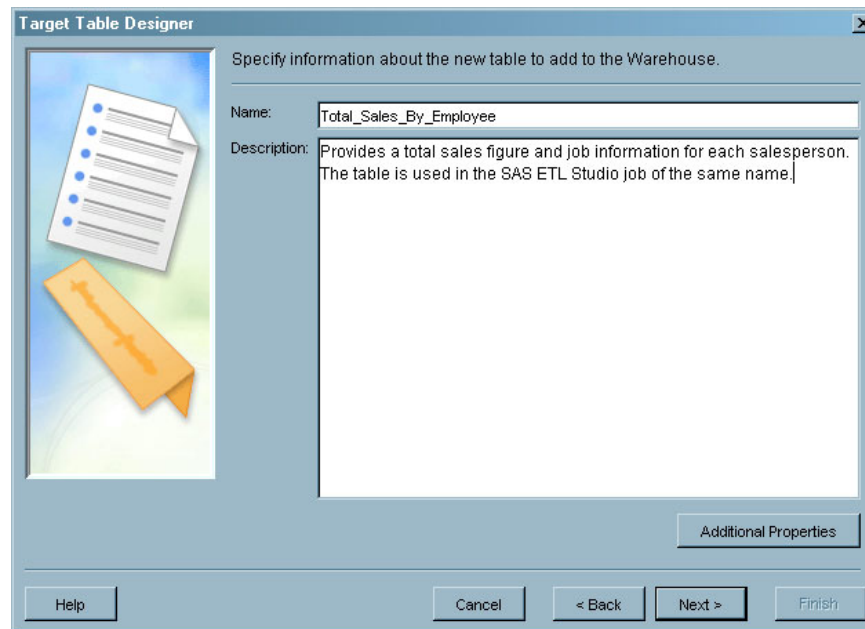
- 2 In the Target Designer selection window, click the **Target Table** icon and click **Next**. The wizard attempts to open a connection to the default SAS application server. If the connection is successful, the first window in the wizard is displayed: the name and description window.

Enter a Name and Description

Use the first window in the Target Table Designer to enter a name and description for the metadata object that will specify the target table.

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 in a later window in this wizard. △

In this example, the name of the metadata object is `Total_Sales_By_Employee`. The description is as follows: “Provides a total sales figure and job information for each salesperson. The table is created by joining the source tables `ORDER_FACT` and `ORGANIZATION_DIM`.”

Display 8.2 Name and Description Window

When the text is ready, click **Next** to display the import columns window.

Select Column Metadata from Existing Tables

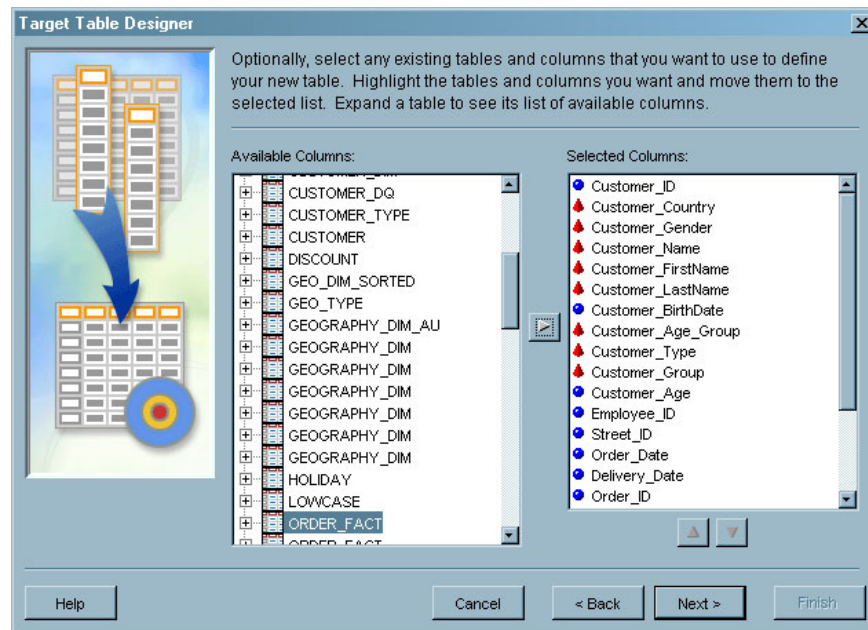
If you want your target columns to be similar to the columns in tables that are already defined, use the import columns window to import metadata for the appropriate columns.

For example, as noted in “Preparation” on page 89, the tables `ORGANIZATION_DIM` and `ORDER_FACT` will be joined and transformed to supply data to the target `Total_Sales_By_Employee`. Accordingly, it would be appropriate to import selected columns from `ORGANIZATION_DIM` and `ORDER_FACT`.

Follow these steps to import metadata for the appropriate columns.

- 1 In the import columns window, locate the **Available Columns** tree. In that tree, click the icon for the table `ORGANIZATION_DIM`. Then click the right arrow to move all of the columns in this table into the **Selected Columns** list box.
- 2 In the **Available Columns** tree, click the icon for the table `ORDER_FACT`, then click the right arrow again to move the columns of that table into the **Selected Columns** list box.

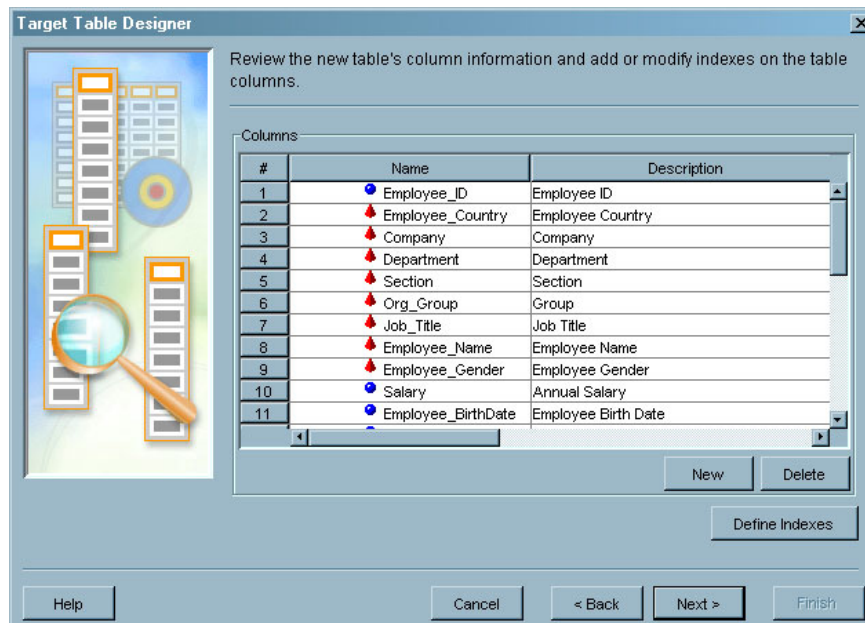
In this example, a pop-up message is displayed to indicate that one column in the table `ORDER_FACT` is not added to the **Selected Columns** list box, because that same column was already added from the table `ORGANIZATION_DIM`. Click **OK** to clear the pop-up message.

Display 8.3 Import Columns Window

3 Click **Next** to display the target columns window.

Specify Column Metadata for the Target

Use the target columns window to review and update any imported metadata for columns. You can also add metadata for new columns.

Display 8.4 Target Columns Window

Scroll down through the target columns to verify that you have the columns that you need. For our example, the columns are correct. When we create and run the job as described in “Example: Creating a Job That Joins Two Tables and Generates a Report” on page 135, we will modify these original column specifications.

Scroll to the top, then scroll right to see the column metadata. You can change any metadata value by selecting it with the left mouse button. In our example, you could add descriptions to the columns that came from the ORDER_FACT table.

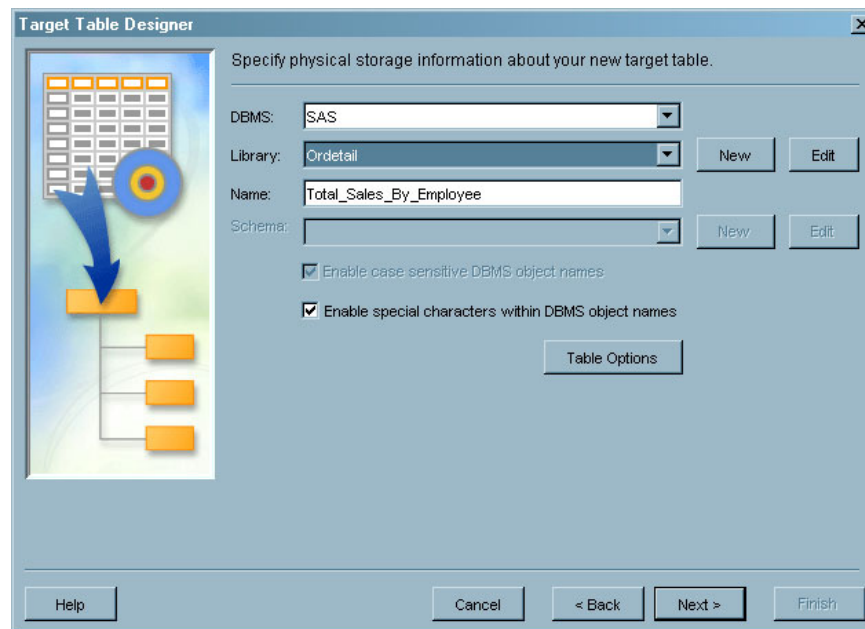
Note that you are defining column metadata for the target. You have not yet created the physical target table. The target metadata indicates where the data can be found and how it is to be formatted.

When you have reviewed and updated the column metadata, click **Next** to display the physical storage window.

Specify Physical Storage Information for the Target

Use the physical storage window to specify the format and location of the target, as shown in the following display.

Display 8.5 Physical Storage Window



For our example target, you would follow these steps:

- 1 In the **DBMS** field, select **SAS**.
- 2 In the **Library** field, click the down arrow. A list of existing libraries is displayed.
- 3 Scroll down through the list and choose the library **Ordetail**.
- 4 In the **Name** field, accept the default, **Total_Sales_By_Employee**. The default is the name that you entered in the first window of the Target Table Designer wizard.
- 5 After you have specified a format, a library, and a table name, click **Next** to go to the finish window.

Usage Hints for the Physical Storage Window

Keep the following in mind as you use the physical storage window:

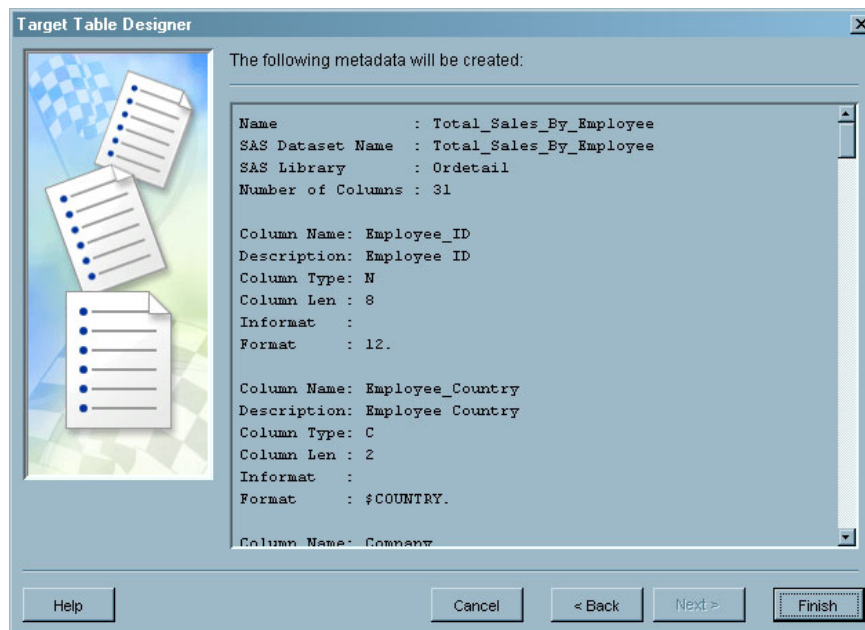
- The name that you specify in the **Name** field must follow the rules for table names in the format that is selected in the **DBMS** field. For example, if SAS is the selected DBMS, the name must follow the rules for SAS data sets. If you select another DBMS, the name must follow the rules for tables 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 68. See also the usage note “Case and Special Characters in SAS Names” on page 181.
- 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 database management system.

Save Metadata for the Target

After you have specified physical storage information, you review all of the metadata that you have defined for your new target.

In the finish window, scroll down to confirm that you have the metadata that you need. If you need to change any of the metadata, click **Back** to display the wizard windows that you need to make your changes.

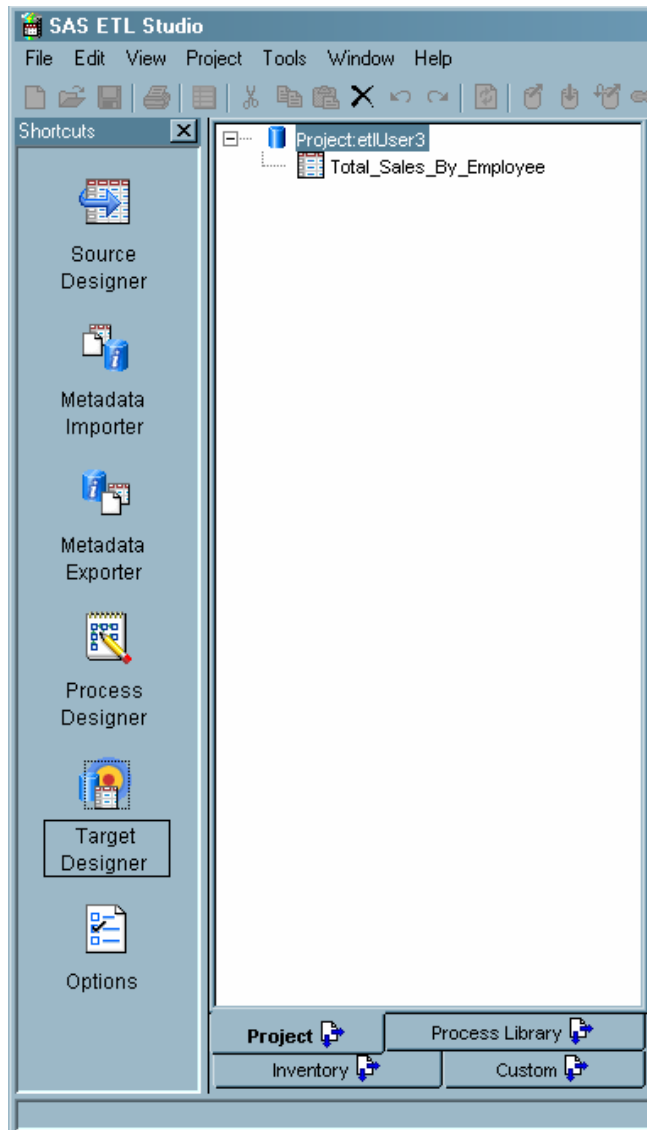
Display 8.6 Finish Window



When you have confirmed that the target metadata is correct, click **Finish** to store the metadata for your new target. The new target is displayed in the Project tree. Next, you check in the metadata object for the target.

Check In the Target Metadata

Under change management, new metadata objects are added to the Project tree on the SAS ETL Studio desktop.

Display 8.7 Project Tree with a Metadata Object for a New Target

Follow these steps to check the new target into the change-managed repository:

- 1 In the Project tree, select the repository icon (*Project: etlUser1*).
- 2 From the menu bar on the SAS ETL Studio desktop, select

Project

 ► Check In Repository

The metadata object in the project repository is checked into the change-managed repository. The new object is displayed as checked-out in the Inventory and Project trees. The new target 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 135.

Additional Information about Targets

Cube targets are described in a separate chapter, Chapter 11, “Creating Cubes,” on page 159.

The online help for SAS ETL Studio provides additional information about how to enter metadata for targets. To display the relevant help topics:

- 1 Start SAS ETL Studio.
- 2 From the menu bar, select

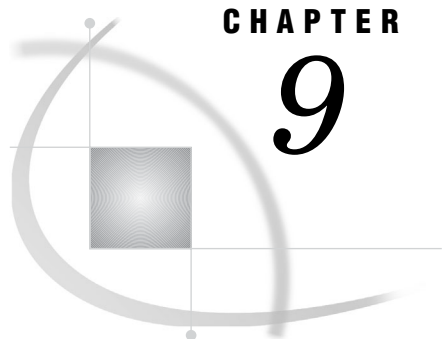
Help ► Contents

The online help window is displayed.

- 3 In the left pane of the help window, select

Working with Targets ► Specifying Metadata for Targets

to view the topic.



CHAPTER

9

Introduction to SAS ETL Studio Jobs

<i>Overview</i>	100
<i>What Is a Job?</i>	100
<i>Jobs with Generated Source Code</i>	100
<i>Jobs with User-Written Source Code</i>	101
<i>Jobs Must Be Executed</i>	102
<i>Jobs Can Be Scheduled</i>	102
<i>Main Windows for Jobs</i>	102
<i>New Job Wizard</i>	103
<i>Process Designer Window</i>	105
<i>Process Editor Tab</i>	107
<i>Source Editor Tab</i>	107
<i>Log Tab</i>	107
<i>Output Tab</i>	107
<i>Process Library Tree</i>	108
<i>Analysis Folder</i>	109
<i>Data Transforms Folder</i>	110
<i>Output Folder</i>	110
<i>Publish Folder</i>	110
<i>Java Transformations and SAS Code Transformations</i>	111
<i>Job Properties Window</i>	111
<i>Table Properties Window</i>	112
<i>Transformation Property Windows</i>	112
<i>Transformation Generator Wizard</i>	113
<i>General Tasks for Jobs</i>	114
<i>Creating and Running Jobs</i>	114
<i>Prerequisites</i>	114
<i>Check Out Any Metadata That Is Needed</i>	114
<i>Create and Populate the New Job</i>	115
<i>View or Update the Job as Needed</i>	115
<i>Run and Troubleshoot the Job</i>	116
<i>Verify the Job's Outputs</i>	116
<i>Check In the Job</i>	117
<i>Working under Change-Management Control</i>	117
<i>Using the New Job Wizard</i>	117
<i>Using Source or Target Designers to Create Jobs</i>	117
<i>Creating Jobs That Retrieve User-Written Code</i>	118
<i>Viewing the Basic Metadata for a Job</i>	118
<i>Updating the Basic Metadata for a Job</i>	118
<i>Viewing the Data for a Source or a Target in a Job</i>	119
<i>Viewing the Metadata for a Source, Target or Transformation in a Job</i>	120
<i>Updating the Metadata for a Source, Target, or Transformation in a Job</i>	120

<i>Impact of Updating a Table's Metadata</i>	121
<i>Updating Column and Mapping Metadata</i>	121
<i>Running a Job</i>	122
<i>Deploy a Job for Scheduling</i>	122
<i>Example: Creating a SAS Code Transformation Template</i>	122
<i>Overview</i>	123
<i>Preparation</i>	123
<i>Start SAS ETL Studio and Open the Appropriate Metadata Profile</i>	124
<i>Display the Transformation Generator Wizard</i>	124
<i>Specify SAS Code for the Transformation Template</i>	125
<i>Define Any User-Defined Variables</i>	126
<i>Specify the Remaining Options for the Transformation Template</i>	128
<i>Save the Transformation Template</i>	129
<i>Document Any Usage Details for the Template</i>	130
<i>General Tasks for SAS Code Transformation Templates</i>	130
<i>Using a SAS Code Transformation Template in a Job</i>	130
<i>Identifying a SAS Code Transformation Template</i>	130
<i>Importing and Exporting SAS Code Transformations</i>	130
<i>Exporting a SAS Code Transformation</i>	131
<i>Importing a SAS Code Transformation</i>	131
<i>Controlling Access to a SAS Code Transformation</i>	131
<i>Deleting Folders for SAS Code Transformations</i>	132
<i>Additional Information about Jobs</i>	132

Overview

After you have entered metadata for sources and targets, you are ready to load the targets in a data warehouse or data mart. This chapter introduces SAS ETL Studio jobs. Use the general steps in this chapter, together with the examples that are described in the next chapter, to load targets in your data warehouse or data mart.

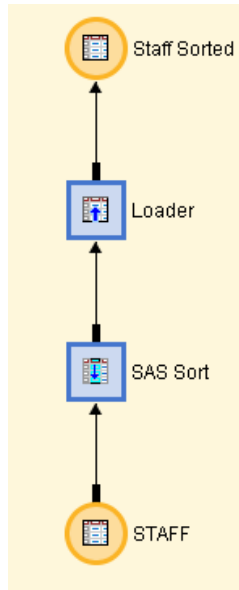
What Is a Job?

In SAS ETL Studio, a job is a metadata object that specifies processes that create output. SAS ETL Studio uses each job to generate or retrieve SAS code that reads sources and creates targets on a file system.

Jobs with Generated Source Code

If you want SAS ETL Studio to generate code for a job, you define a process flow diagram 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.

For example, Display 9.1 on page 101 shows the diagram for a job that will read data from a source table called STAFF, sort the data, then write the sorted data to a target table called Staff Sorted.

Display 9.1 Simple Process Flow Diagram

Given the direction of the arrows in the previous process flow diagram, **STAFF** specifies metadata for the source table. **SAS Sort** specifies metadata for the sort process. **Loader** specifies metadata for a process that loads data into the target table, **Staff Sorted**. The **Staff Sorted** object specifies metadata for the **Staff Sorted** table.

Each process in a process flow diagram is specified by a metadata object called a transformation. In Display 9.1 on page 101, **SAS Sort** and **Loader** are transformations. A transformation 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.

Jobs with User-Written Source Code

SAS ETL Studio enables you to do the following:

- Specify user-written code for an entire job or a transformation within a job. For a summary of this task, see “Creating Jobs That Retrieve User-Written Code” on page 118.
- Drag a User-Written transformation template from the Process Library and drop it into the process flow diagram for a job. You can then update the default metadata for the transformation so that it specifies the location of user-written program. For an overview of the Process Library and its transformation templates, see “Process Library Tree” on page 108.
- Use the Transformation Generator wizard to create your own SAS code transformation templates and add them to the Process Library. After a transformation template has been added to the Process Library, you can drag and drop it into any job. For a description of this wizard, see “Transformation Generator Wizard” on page 113.

The online help for SAS ETL Studio provides additional information about working with user-written components. To display the relevant help topics, do the following:

- 1 From the SAS ETL Studio menu bar, select

[Help](#) ► [Contents](#)

The online help window displays.

- 2 In the left pane of the help window, select

[Working with User-Written Components](#)

► [User-Written Components and SAS ETL Studio](#)

Jobs Must Be Executed

After you have defined the metadata for a job, you can submit the job for execution. Until you do that, the targets (output tables) might not exist on the file system. For a description of this task, see “Running a Job” on page 122.

Jobs Can Be Scheduled

If the appropriate software has been installed, you can deploy a SAS ETL Studio job for scheduling. After a job is deployed, an administrator can use SAS Management Console to schedule the job to run at a specified date and time or when a specified event occurs.

In SAS Management Console, administrators create and schedule groups of jobs, called *flows*. Each job within a flow can be triggered to run based on a certain time, the state of a file on the file system, or the status of another job within the flow. Platform Computing’s Load Sharing Facility (LSF) is used to schedule the job.

The online help for SAS ETL Studio provides details about deploying and scheduling jobs. Perform these steps to display the relevant help topics in SAS ETL Studio:

- 1 From the SAS ETL Studio desktop, select

[Help](#) ► [Contents](#)

from the menu bar. The online help window displays.

- 2 In the left pane of the help window, open the **Working with Jobs** folder and select **Maintaining Jobs**.
- 3 In the **Maintaining Jobs** topic, view the **Deploying a Job for Scheduling** topic.

For scheduling setup and installation, administrators should see the SAS ETL Studio chapter in the *SAS Intelligence Architecture: Planning and Administration Guide*.

For details about using the Schedule Manager in SAS Management Console, see the online help for the Schedule Manager. See also the Managing Job Schedules chapter in the *SAS Management Console: User’s Guide*.

Main Windows for Jobs

The following table lists the main windows and components that are used to maintain jobs and tables in SAS ETL Studio. Each component is briefly described in the sections that follow.

Table 9.1 Jobs Interface

Components	Description
“New Job Wizard” on page 103	Enables you to select one or more tables as the targets (outputs) of a job. Can also be used to create an empty job into which you can drag and drop tables and transformation templates.
“Process Designer Window” on page 105	Enables you to create process flow diagrams, to generate and submit code for jobs, and to perform related tasks.
“Process Library Tree” on page 108	Enables you to drag and drop transformation templates into the process flow diagrams for jobs.
“Job Properties Window” on page 111	Enables you to view or update the basic metadata for a job (metadata other than its process flow diagram).
“Table Properties Window” on page 112	Enables you to view or update the metadata for a source table or a target table, such as metadata for its columns, indexes, keys and other attributes.
“Transformation Property Windows” on page 112	Enables you to view or update the metadata for a transformation in the process flow diagram for a job.
“Transformation Generator Wizard” on page 113	Enables you to create a user-written, SAS code transformation and make it available in the Process Library tree. This is one of the easiest ways to customize SAS ETL Studio.

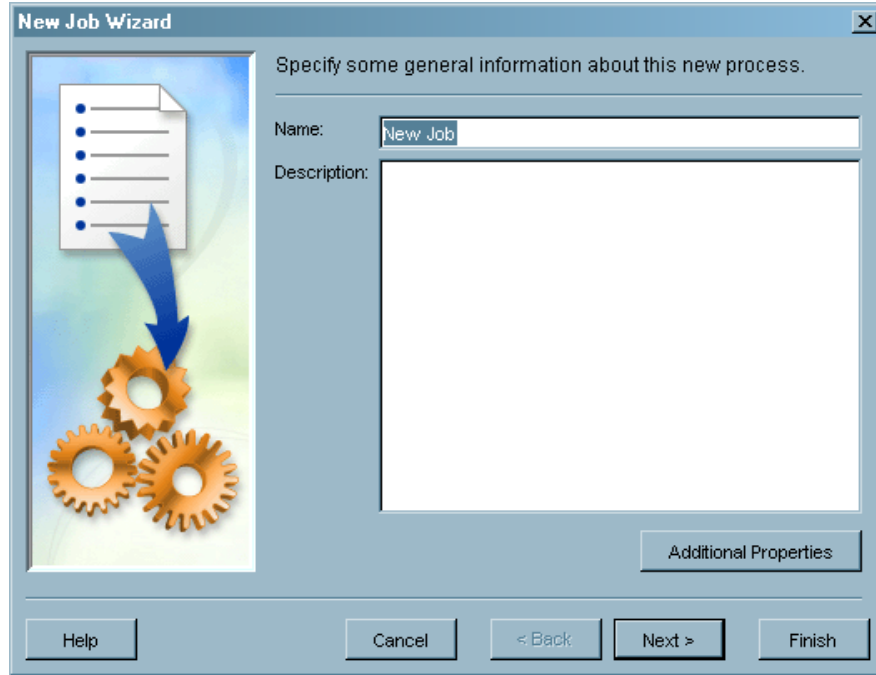
New Job Wizard

Use the New Job wizard to select one or more tables as the targets (outputs) of a job. It 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

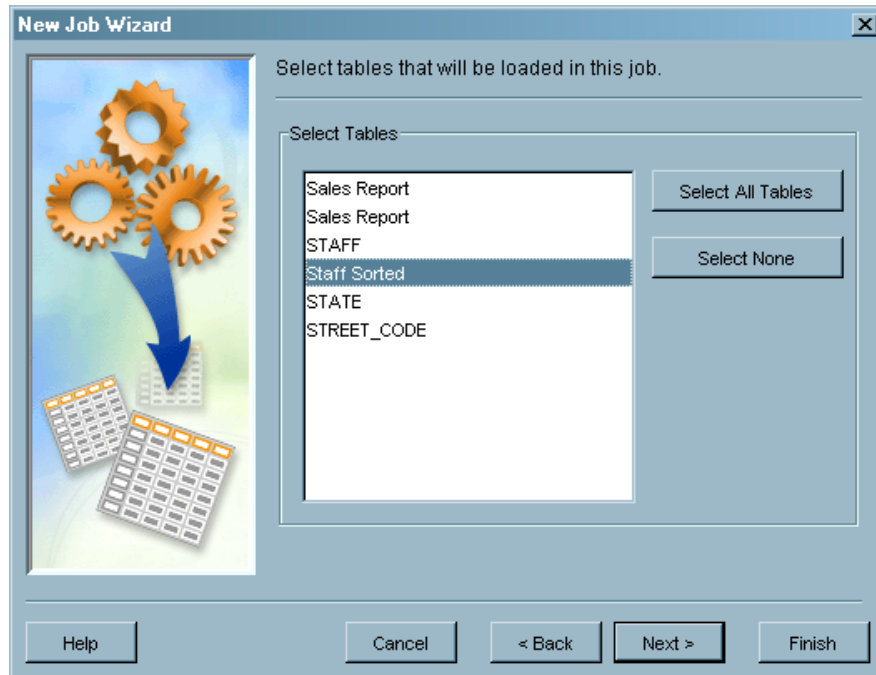
from the menu bar on the SAS ETL Studio desktop. The first window in the wizard is shown in the following display.

Display 9.2 New Job Wizard


The first window of the New Job Wizard is titled "New Job Wizard". It contains a graphic on the left showing a document icon with a blue arrow pointing to three interlocking orange gears. The main area is titled "Specify some general information about this new process." and contains two input fields: "Name:" with the text "New Job" and "Description:" with a large empty text area. At the bottom right is an "Additional Properties" button. The bottom of the window has a row of buttons: "Help", "Cancel", "< Back", "Next >", and "Finish".

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.

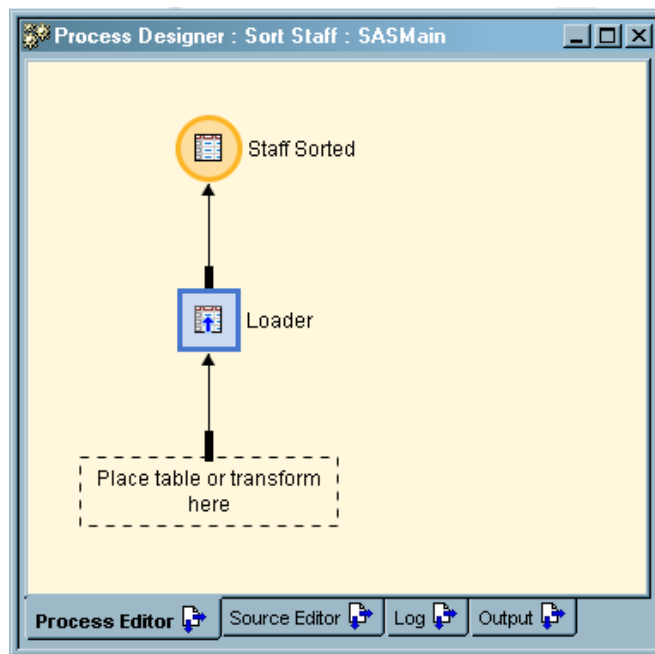
Display 9.3 New Job Wizard, Second Window


The second window of the New Job Wizard is titled "New Job Wizard". It contains a graphic on the left showing three interlocking orange gears with a blue arrow pointing to two spreadsheets. The main area is titled "Select tables that will be loaded in this job." and contains a "Select Tables" section with a list box. The list box contains the following items: "Sales Report", "Sales Report", "STAFF", "Staff Sorted" (which is selected), "STATE", and "STREET_CODE". To the right of the list box are two buttons: "Select All Tables" and "Select None". The bottom of the window has a row of buttons: "Help", "Cancel", "< Back", "Next >", and "Finish".

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 Display 9.4 on page 105.

Display 9.4 Transformation Template for Sort Staff Job



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

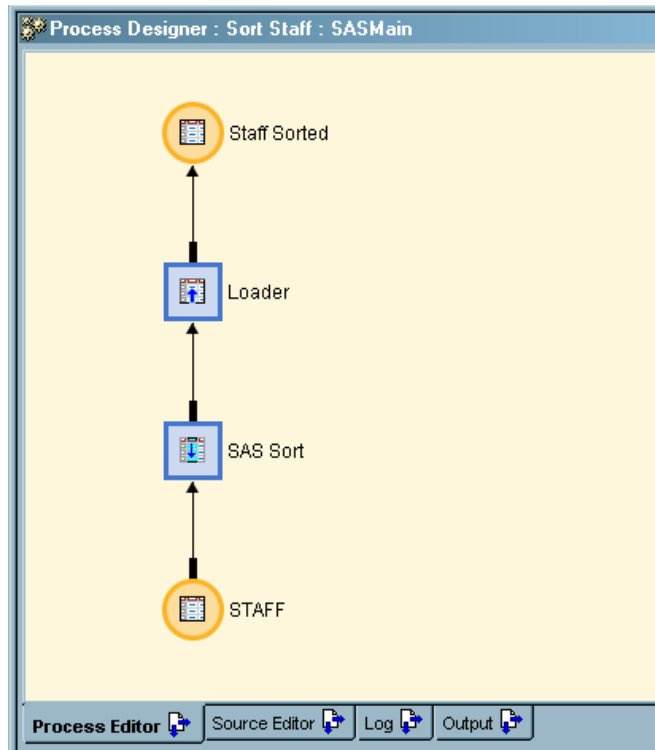
Alternatively, in the second window of the New Job wizard, you can select no targets and simply click **Finish** after entering a name for the job. The wizard will open an empty job in the Process Designer window. After you have an empty job, you can create a process flow diagram by dragging and dropping tables and transformations into the Process Designer window. This is the approach that is described in “Create and Populate the New Job” on page 115.

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.

Display 9.5 on page 106 shows a typical view of this window.

Display 9.5 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 “Jobs with Generated Source Code” on page 100. Note that the **Process Editor** tab is shown by default. You might have 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 18.

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

- 1 From the SAS ETL 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 of 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.

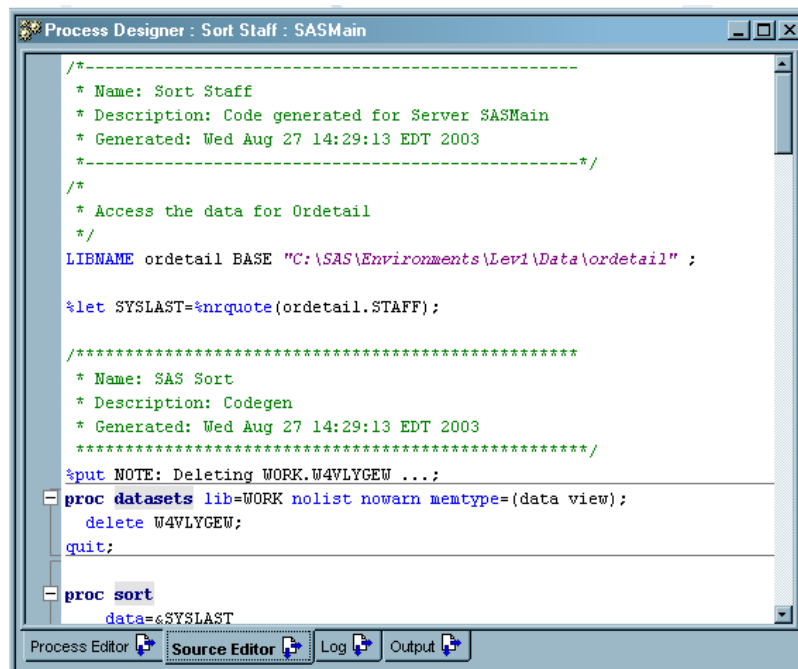
Process Editor Tab

Use the **Process Editor** tab to add and maintain a process flow diagram for the selected job. For summary of how you can use the Process Editor to create a process flow diagram for a job, see “Creating and Running Jobs” on page 114.

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 in the **Process Editor** tab, and you selected the **Source Editor** tab, code for the entire job would be generated and displayed. Display 9.6 on page 107 shows some of the code that would be generated for the Sort Staff job.

Display 9.6 Source Editor Tab



Log Tab

Use the **Log** tab to view the SAS log 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 “Run and Troubleshoot the Job” on page 116.

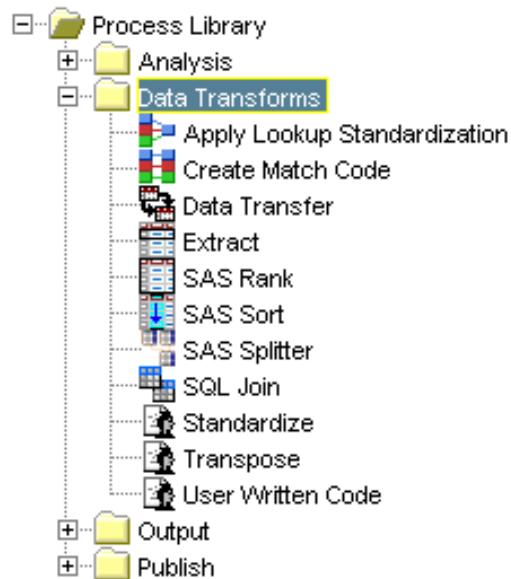
Output Tab

Use the **Output** tab to view any printed output from a SAS program. For example, SAS ETL Studio jobs that produce reports can specify that the reports are sent to the Output tab. For examples of such reports, see “Analysis Folder” on page 109.

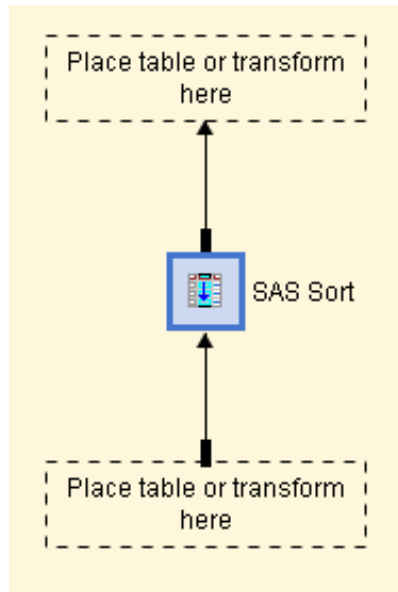
Process Library Tree

The Process Library tree is one of the tabs in the tree view of the SAS ETL Studio desktop. If you select this tab, it displays a collection of transformation templates. As shown in Display 9.7 on page 108, the templates are organized into folders, such as Analysis, Data Transforms, Output, and Publish.

Display 9.7 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 Display 9.8 on page 109.

Display 9.8 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 Display 9.8 on page 109.
- 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 9.5 on page 106.
- Transformation objects themselves. Transformations that can take multiple inputs or outputs have drop zones on the transformation itself.

The sort template shown in the previous display could be used to create the diagram that is shown in Display 9.1 on page 101. For summary of how you can use the Process Editor and the Process Library tree to create a process flow diagram for a job, see “Create and Populate the New Job” on page 115.

SAS provides a number of transformation templates with SAS ETL Studio. The standard templates in the Process Library are described in the following sections.

Analysis Folder

The Analysis folder in the Process Library includes the following transformation templates:

- Correlations— creates an output table containing correlation statistics.
- Correlations-Report—creates a report containing summary correlation statistics.
- Distribution Analysis—creates an output table containing a distribution analysis.
- Distribution Analysis- Report—creates a report containing a distribution analysis.
- Frequency—creates an output table containing frequency information.
- Frequency-Report—creates a report containing frequency information.
- Summary Statistics—creates an output table containing summary statistics.
- Summary Statistics-Report—creates a report containing summary statistics.
- Summary Tables Report—creates a report containing summary tables.

Report output can be saved as an HTML file. Alternatively, it can be sent to the Output tab in the Process Designer window.

Note: Report transformations must be the last object in a process flow diagram. \triangle

Data Transforms Folder

The Data Transforms folder in the Process Library includes the following templates:

- ☐ Apply Lookup Standardization—applies schema data sets to transform source columns. Requires SAS data quality software.
- ☐ Create Match Codes—creates match codes for specified source columns. Requires SAS data quality software.
- ☐ Data Transfer—moves data directly from one machine to another. Use this when user options are required.
- ☐ Extract—selects rows from a source and writes those rows to a target. Typically used to create ordered rows in a target. Can also be used to create columns in a target that are derived from columns in a source. Uses an SQL WHERE clause.
- ☐ 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. Can also be used to eliminate duplicates (by specifying NODUP in the **Duplicates** field on the **Options** tab of the properties window for this transformation).
- ☐ 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.
- ☐ 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. Supports various SQL joins, including inner, outer, left, and right. Can also be used to merge two or more copies of a single source.
- ☐ Standardize—creates an output table containing data standardized to a particular number.
- ☐ Transpose—creates an output table containing transposed data.
- ☐ User Written Code—retrieves user-written code for a process in a job. It 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 just as you can for other transformations.

Output Folder

The Output folder in the Process Library includes the following template:

- ☐ List Data—Creates a report that contains selected columns from a source table. Use PROC PRINT.

Report output can be saved as an HTML file. Alternatively, it can be sent to the Output tab in the Process Designer window.

Note: Report transformations must be the last object in a process flow diagram. \triangle

Publish Folder

The Publish folder in the Process Library includes the following templates:

- ❑ Publish to Archive—creates a report and an archive of the report.
- ❑ Publish to Email—creates a report and emails it to a designated address.
- ❑ Publish to Queue—creates a report and publishes it to a queue using MQSeries.

Report output can be saved as an HTML file. Alternatively, it can be sent to the Output tab in the Process Designer window.

Note: Report transformations must be the last object in a process flow diagram. △

Java Transformations and SAS Code Transformations

The Process Library tree contains two different kinds of transformation templates: Java plug-in transformation templates and SAS code transformation templates.

Java plug-in transformation templates are created with the Java programming language. Examples include most of the templates in the Data Analysis folder, such as SAS Sort, SAS Splitter, and User Written. For details about creating your own Java plug-ins, see Appendix 2, “Building Java Plug-ins for SAS ETL Studio,” on page 183.

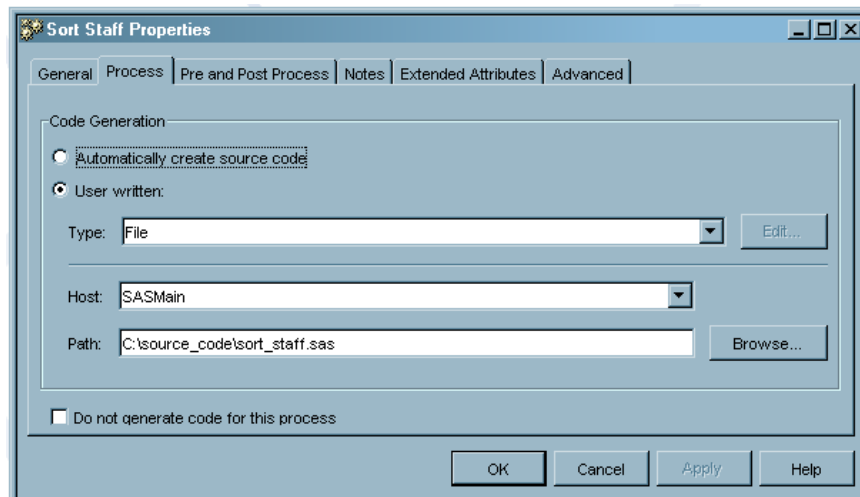
SAS code transformation templates are created with the Transformation Generator wizard. SAS provides a number of these templates in the Process Library tree. With the exception of the User Written template, which is a Java plug-in, all of the other templates in the Process Library that have the user-written icon are SAS code transformations. Examples include all of the templates in the Analysis folder.

For details about the Transformation Generator wizard, see “Transformation Generator Wizard” on page 113. For details about working with SAS code transformations, see “Example: Creating a SAS Code Transformation Template” on page 122 and “General Tasks for SAS Code Transformation Templates” on page 130.

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 ETL 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. Display 9.9 on page 111 shows a typical window.

Display 9.9 Job Properties Window



If you wanted to specify user-written code for the Sort Staff job that is described in “Jobs with Generated Source Code” on page 100, you could enter metadata similar to that shown in Display 9.9 on page 111. 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.

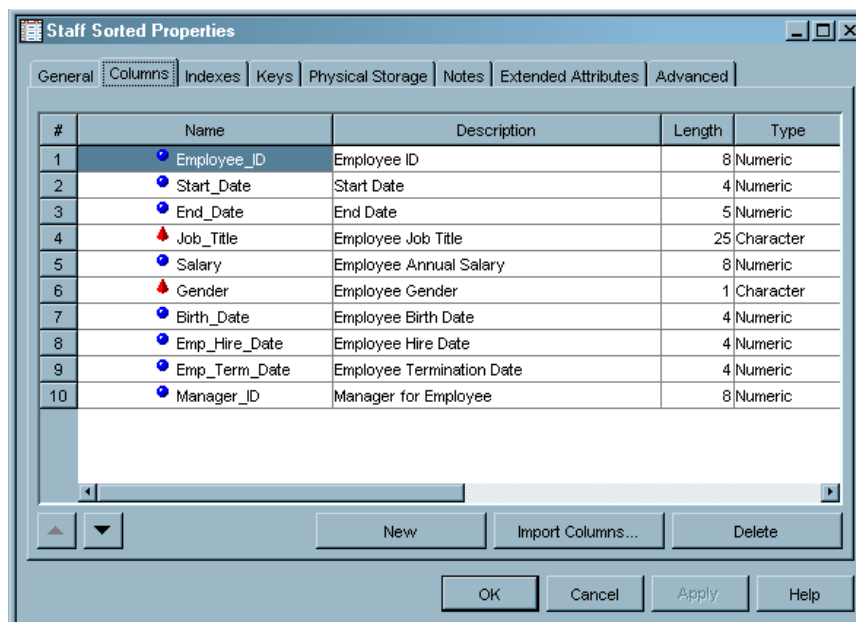
If you wanted to execute code before or after the Sort Staff job is executed, you could click the **Pre and Post Process** tab and specify the code. For example, you might want to issue a SAS LIBNAME statement before the job is run.

For a summary of how to use the job properties window, see “Viewing the Basic Metadata for a Job” on page 118 and “Updating the Basic Metadata for a Job” on page 118.

Table Properties Window

Use the table properties window for a source or a target to view or update the metadata for its columns, indexes, keys and other attributes. Display 9.10 on page 112 shows a typical window.

Display 9.10 Table Properties Window



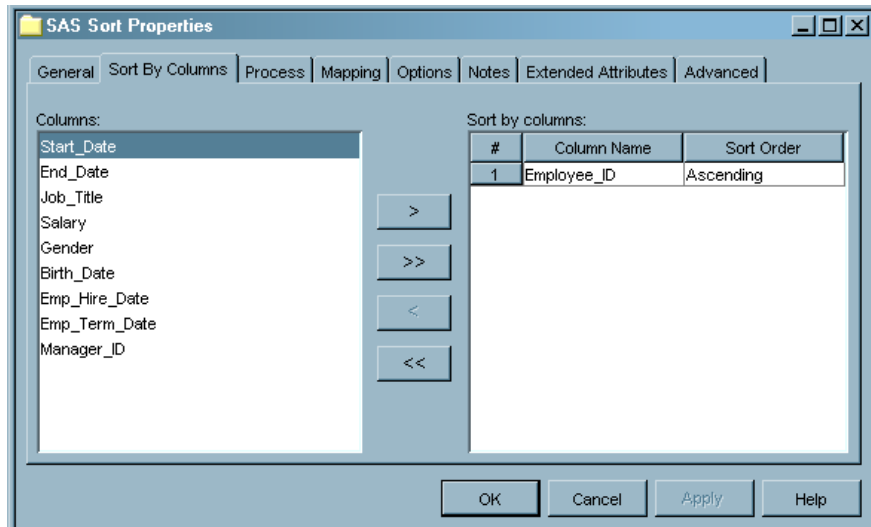
The window shown in the previous display contains the metadata for the Staff Sorted table from Display 9.1 on page 101. For a summary of how to use this window in the context of a job, see “Viewing the Metadata for a Source, Target or Transformation in a Job” on page 120 and “Updating the Metadata for a Source, Target, or Transformation in a Job” on page 120.

Transformation Property Windows

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 ETL 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. Display 9.11 on page 113 shows a typical window.

Display 9.11 Transformation Properties Window



The window shown in the previous display contains the metadata for the SAS Sort transformation from Display 9.1 on page 101. 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 “Viewing the Metadata for a Source, Target or Transformation in a Job” on page 120 and “Updating the Metadata for a Source, Target, or Transformation in a Job” on page 120.

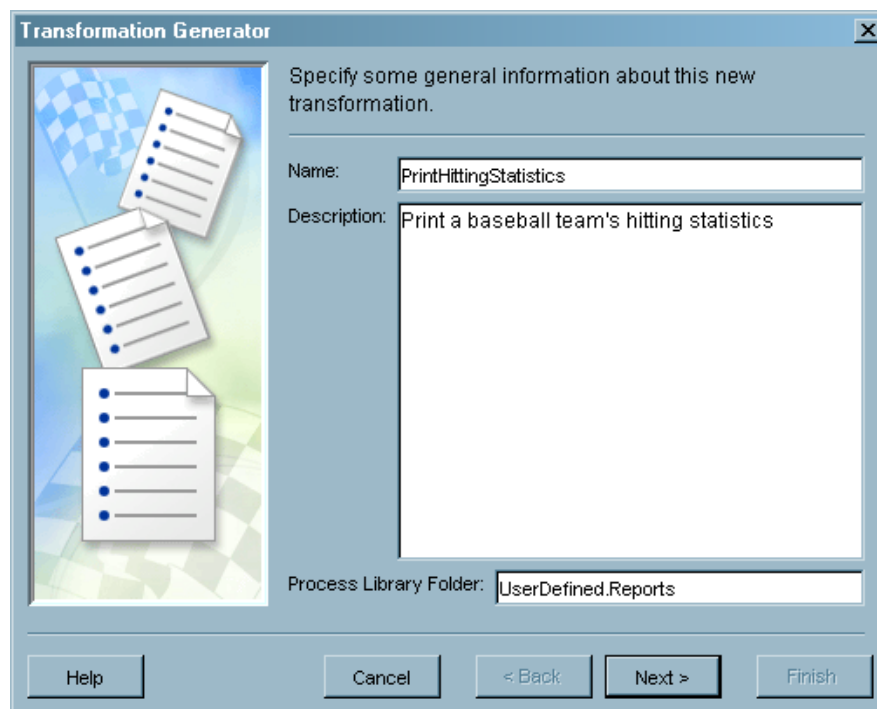
Transformation Generator Wizard

One of the easiest ways to customize SAS ETL Studio is to write your own SAS code transformation templates. Unlike Java-based plug-ins that require software development, SAS code transformation templates are created with a wizard. The Transformation Generator wizard guides you through the steps of specifying SAS code for a transformation template and registering the template in the current metadata repository. Once registered, your custom template is displayed in the Process Library tree, where it is available for use in any job.

To display the Transformation Generator wizard, display the SAS ETL Studio desktop, then select

Tools ► Transformation Generator

from the menu bar. The first window of the wizard is displayed, as shown in the following display.

Display 9.12 General Information Window, Transformation Generator Wizard


The image shows a Windows-style dialog box titled "Transformation Generator". On the left is a graphic of three overlapping document icons. The main area contains the text "Specify some general information about this new transformation." followed by three input fields: "Name:" with the value "PrintHittingStatistics", "Description:" with the value "Print a baseball team's hitting statistics", and "Process Library Folder:" with the value "UserDefined.Reports". At the bottom are five buttons: "Help", "Cancel", "< Back", "Next >", and "Finish".

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

For details about using the Transformation Generator wizard, see “Example: Creating a SAS Code Transformation Template” on page 122.

General Tasks for Jobs

Creating and Running Jobs

Here is a summary of the main tasks for creating and running jobs. For examples that illustrates all of these tasks, see Chapter 10, “Loading Targets in a Data Warehouse or Data Mart,” on page 135.

Prerequisites

It is much easier to create a job if metadata for the sources and targets in the job are created first. For details about these tasks, see Chapter 7, “Defining Sources for a Data Warehouse or Data Mart,” on page 63 and Chapter 8, “Defining Targets for a Data Warehouse or Data Mart,” on page 85.

Check Out Any Metadata That Is Needed

You must check out the metadata for any existing sources and targets that you want to add to a job.

- 1 On the SAS ETL Studio desktop, select the Inventory tree.

- 2 In the Inventory tree, open the **Tables** folder.
- 3 Select all source tables and target tables that you want to add to the new job, then select

Project ► Check Out

The metadata for these tables 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 New Job

With the relevant sources and targets checked out in the Project tree, 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 From the SAS ETL 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 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 ETL Studio desktop.

As you add sources, targets, and transformations to a process flow diagram, SAS ETL 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.

View or Update the Job as Needed

The following steps describe a general approach for viewing or 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, “Loading Targets in a Data Warehouse or Data Mart,” on page 135 describe two scenarios in which a few, specific updates are needed to the automatic column mappings and the other default metadata in a job.

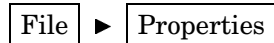
- 1 In the Process Designer window, select the first source in the flow, then select

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. See also “Updating Column and Mapping Metadata” on page 121.

- 3 In the Process Designer window, select the first transformation in the flow, then select



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 from working in a source-to-target direction. For details about updating mapping metadata, click the **Help** button on the **Mapping** tab. See also “Updating Column and Mapping Metadata” on page 121.

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

Run and Troubleshoot the Job

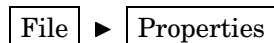
After the metadata for a job is complete, you must submit the job for execution in order to create targets on the file system.

- 1 With the job displayed in the Process Designer window, select



from the menu bar. SAS ETL 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.

- 2 If a pop-up error message appears, or if you simply want to look at the log for the completed job, click the **Log** tab in the Process Designer window.
- 3 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.)
- 4 If you find errors in the source code for a step, select the corresponding transformation in the process flow diagram, then select



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



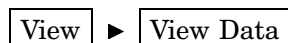
from the menu bar.

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

Verify the Job's Outputs

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

- 1 To view the data for a target in the job's process flow diagram, select the desired target, then select



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.

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 Job

Perform these steps to check in a job in the Project tree:

- 1 In the Project tree, select the repository icon.
- 2 On the SAS ETL Studio desktop, select

Project

▶
Check In Repository

from the menu bar. All of the objects in the Project repository are checked in to the change-managed repository.

Working under Change-Management Control

Unless your user profile includes administrative privileges, you will be working under change management control. For a general description of how change management affects user tasks in SAS ETL Studio, see “Working with Change Management in SAS ETL Studio” on page 21.

When working with jobs, the main impacts of change management are as follows:

- 1 To update an existing job, you must check out the job.
- 2 When you check out a job, the metadata that is associated with the job will be checked out and will appear in the Project tree. The metadata that will be checked out includes the metadata object for the job as a whole and the metadata objects for any sources and targets that have been added to the job.
- 3 You must check out any existing sources and targets that you want to add to a job.
- 4 Metadata for new objects such as jobs, sources, and targets is added to the Project tree. At some point, you must check in new objects to the change-managed repository.

Using the New Job Wizard

See “New Job Wizard” on page 103.

Using Source or Target Designers to Create Jobs

Most source designers and target designers do not create jobs, but some do. The External File source designer and the Cube Designer create and execute jobs. The metadata for the job is saved so that you can run it as desired—to refresh the data in the target, for example.

For details about using the External File source designer, see “Example: Extracting Information from a Flat File” on page 74.

For details about using the Cube Designer, see “Example: Building a Cube from a Star Schema” on page 162.

Creating Jobs That Retrieve User-Written Code

When you create a job, SAS ETL Studio will generate code for that job unless you specify otherwise. This generated code will often suffice, but in some cases, you might want to specify user-written code for a whole job or for transformations in a job.

In order to track the jobs in a data warehouse, it is best to capture as much metadata as possible about a job, even if the job is handled by user-written code. Accordingly, a good way to specify user-written code for a job is to use SAS ETL Studio wizards to create a job as usual, then update the properties for the whole job or for transformations within the job so that the metadata specifies the location of user-written code.

The general steps for doing this are as follows:

- 1 Use SAS ETL Studio wizards to create a job.
- 2 Display the properties window for the job or for a transformation within a job. Follow the instructions in “Updating the Basic Metadata for a Job” on page 118 or “Updating the Metadata for a Source, Target, or Transformation in a Job” on page 120.
- 3 In the properties window for the job or a transformation within the job, click the **Process** tab.
- 4 On the **Process** tab, specify the location of user-written code.

The online help for SAS ETL Studio provides more details about user-written code for jobs and transformations. To display the relevant help topics:

- 1 From the menu bar on the SAS ETL Studio desktop, select

Help ► Contents

The online help window displays.

- 2 In the left pane of the help window, select

Working with User-Written Code ► User-Written Code and SAS ETL Studio
 ► Understanding User-Written Source Code for Jobs

Viewing the Basic Metadata for a Job

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.

- 1 From the SAS ETL Studio desktop, display the Inventory tree.
- 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.

Updating the Basic Metadata for a Job

Use the property window for a job to update its basic metadata. For example, you can specify user-written code for the entire job, or you can specify code that should be

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

- 1 On the SAS ETL Studio desktop, select the Inventory tree.
- 2 In the Inventory tree, open the **Jobs** folder.
- 3 Select the desired job, then select

Project ► **Check Out**

The metadata that is associated with the job will be checked out and will appear in the Project tree. The metadata that will be checked out includes the metadata object for the job as a whole and the metadata objects for any sources and targets that have been added to the job.

- 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, you can check in your changes. In the Project tree, select the repository icon.
- 7 From the menu bar on the SAS ETL Studio desktop, select

Project ► **Check In Repository**

Viewing the Data for a Source or a Target in a Job

After the metadata for a source table or a target table has been added to a job, you might want to verify that the corresponding physical table contains the data that you were expecting. Perform the following steps to view the data that corresponds to the metadata for a source or a target.

Note: The metadata for a target might not point to a physical table until after the target's job has been 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 source or a target in the process flow diagram for a job:

- 1 From the SAS ETL Studio desktop, display the Inventory tree.
- 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 in the **Process Editor** tab of the Process Designer window.

- 4 To view the data for a source or a target in the process flow diagram, select the desired source or target, then select

View ► **View Data**

from the menu bar. The data in the source or target is displayed. If the data is correctly displayed, the metadata for the source or target is correct.

Viewing the Metadata for a Source, Target or Transformation in a Job

To view the metadata for a source table, target table, or a transformation in the process flow diagram for a job, perform the following steps.

- 1 From the SAS ETL Studio desktop, display the Inventory tree.
- 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 in the **Process Editor** tab of the Process Designer window.

- 4 To view the metadata for a source, target, or transformation in the process flow diagram, select the desired object, then select

File ► Properties

from the menu bar. A properties window for the object is displayed.

- 5 Use the tabs in this window to view the metadata for the object. Each tab has its own Help button.

Updating the Metadata for a Source, Target, or Transformation in a Job

Perform the following steps to update the metadata for a source table, target table or transformation in a job. Assume that the metadata for the job is currently checked into a change-managed repository.

- 1 On the SAS ETL Studio desktop, select the Inventory tree.
- 2 In the Inventory tree, open the **Jobs** folder.
- 3 Select the desired job, then select

Project ► Check Out

The metadata that is associated with the job will be checked out and will appear in the Project tree. The metadata that will be checked out includes the metadata object for the job as a whole and the metadata objects for any sources and targets that have been added to the job.

- 4 In the Project tree, select the metadata for the job, then select

View ► View Job

from the menu bar. The process flow diagram for the job displays in the **Process Editor** tab of the Process Designer window.

- 5 To update the metadata for a source, target, or transformation in the process flow diagram, select the desired object, then select

File ► Properties

from the menu bar. A properties window for the object is displayed.

- 6 Use the tabs in this window to update the metadata for the object. Each tab has its own Help button.

- 7 When you are finished updating the metadata, you can check in your changes. In the Project tree, select the repository icon.
- 8 From the menu bar on the SAS ETL Studio desktop, select

Project ► Check In Repository

Impact of Updating a Table's Metadata

Keep in mind that a table, such as a source table or a target table, can be used in multiple jobs. A table can also be used in multiple places in the same job. Accordingly, when you update the metadata for a table, make sure that the updates are appropriate in all contexts where the metadata is used. For example, if you update the columns for Table 1 in one job, the updates would also have to be appropriate for Table 1 in the context of another job.

Updating Column and Mapping Metadata

In general, perform the following steps to update the column metadata or mapping metadata in a job. Assume that the metadata for the job is currently checked into a change-managed repository.

- 1 On the SAS ETL Studio desktop, select the Inventory tree.
- 2 In the Inventory tree, open the **Jobs** folder.
- 3 Select the desired job, then select

Project ► Check Out

The metadata that is associated with the job will be checked out and will appear in the Project tree. The metadata that will be checked out includes the metadata object for the job as a whole and the metadata objects for any sources and targets that have been added to the job.

- 4 In the Project tree, select the metadata object for the job, then select

View ► View Job

from the menu bar. The process flow diagram for the job is displayed in the **Process Editor** tab of the Process Designer window.

- 5 In the Process Designer window, select the source or target whose columns you wish to update, or select the transformation whose mappings you wish to update. Then select

File ► Properties

from the menu bar. The properties window for the source, target, or transformation displays

- 6 For a source or a target, click the **Columns** tab. Update the columns as needed. For a transformation, click the **Mappings** tab. Update the Mappings as needed. Click the **Help** button on each tab to see topics that describe how to edit columns and mappings.
- 7 After making your changes, make sure that source columns are correctly mapped through the job. For one-to-one mappings, the column lengths and data types for the source and target columns must match. For derived mappings (mappings in which the target column is a function of the source column), the column lengths and data types for the source and target columns might be different.

To verify that the updated columns are correctly mapped through the job, display the property windows for tables and transformations that follow the updated table. For tables, review the metadata in the **Columns** tab. For transformations, review the metadata in the **Mapping** tab. Make updates as needed. Each tab has its own Help button.

- 8 When you are finished updating the metadata, you can check in your changes. In the Project tree, select the repository icon.
- 9 From the menu bar on the SAS ETL Studio desktop, select

Project ► Check In Repository

Running a Job

After you define the metadata for a job, you must submit the job for execution in order to create targets on the file system.

If the job to be submitted is displayed in the Process Designer window, select

Process ► Submit

from the menu bar. The job is submitted to the default SAS application server and to any server that is specified in the metadata for a transformation within the job.

If the job to be submitted is not displayed in the Process Designer window, perform the following steps:

- 1 From the SAS ETL Studio desktop, display the Inventory tree.
- 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 in the **Process Editor** tab of the Process Designer window.

- 4 Select

Process ► Submit

from the menu bar. The job is submitted for execution.

Deploy a Job for Scheduling

If the appropriate software has been installed, you can deploy a SAS ETL Studio job for scheduling. After a job is deployed, an administrator can use SAS Management Console to schedule the job to run at a specified date and time or when a specified event occurs. For details, see “Jobs Can Be Scheduled” on page 102.

Example: Creating a SAS Code Transformation Template

This example demonstrates how to create a user-written SAS code transformation template.

Overview

As described in “Transformation Generator Wizard” on page 113, one of the easiest ways to customize SAS ETL Studio is to write your own SAS code transformation templates. The Transformation Generator wizard guides you through the steps of specifying SAS code for a transformation template and registering the template in the current metadata repository. Once registered, your custom template is displayed in the Process Library tree, where it is available for use in any job.

The Transformation Generator wizard is used to create custom SAS code transformation templates. The wizard enables you to enter the SAS code that runs when the template is executed as part of a job. This code typically includes macro variables. When you use a macro variable, the person who configures the job in which the template appears must specify the value of the variable, and SAS ETL Studio generates the %let statement that creates the variable and assigns a value to it.

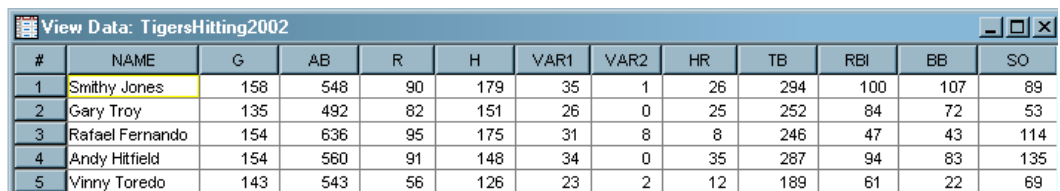
The rules for writing SAS code transformations templates are fairly simple:

- A template can have 0 or 1 input tables or transformation objects and 0 or 1 output tables or transformation objects.
- You cannot use hard-coded names for the input or output.
- The code that is entered must have valid SAS syntax.

Preparation

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

Display 9.13 Contents of Dataset TigersHitting2002



#	NAME	G	AB	R	H	VAR1	VAR2	HR	TB	RBI	BB	SO
1	Smithy Jones	158	548	90	179	35	1	26	294	100	107	89
2	Gary Troy	135	492	82	151	26	0	25	252	84	72	53
3	Rafael Fernando	154	636	95	175	31	8	8	246	47	43	114
4	Andy Hittfield	154	560	91	148	34	0	35	287	94	83	135
5	Vinny Toredó	143	543	56	126	23	2	12	189	61	22	69

The goal is to create a transformation template that takes a dataset such as TigersHitting2002 as input and produces a report. The report will display a user-defined title, a user-defined set of columns, and it will calculate the sum of the values in one column of the table. The following display shows the kind of output that is desired.

Display 9.14 Example Hitting Statistics Report

```

Tigers Hitting Statistics 2002
1
13:29 Monday, November 3, 2003

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:

- 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 directly to the Process Library tree, without having to be checked in. For details about change management, see “Working with Change Management in SAS ETL Studio” on page 21.
- You have selected a default SAS application server for SAS ETL Studio, as described in “Select a Default SAS Application Server” on page 58.

Start SAS ETL Studio and Open the Appropriate Metadata Profile

Perform the following steps to begin work in SAS ETL Studio:

- 1 Start SAS ETL Studio as described in “Start SAS ETL Studio” on page 55.
- 2 Open the appropriate metadata profile as described in “Open a Metadata Profile” on page 58.

The next task is to display the Transformation Generator and start entering metadata.

Display the Transformation Generator Wizard

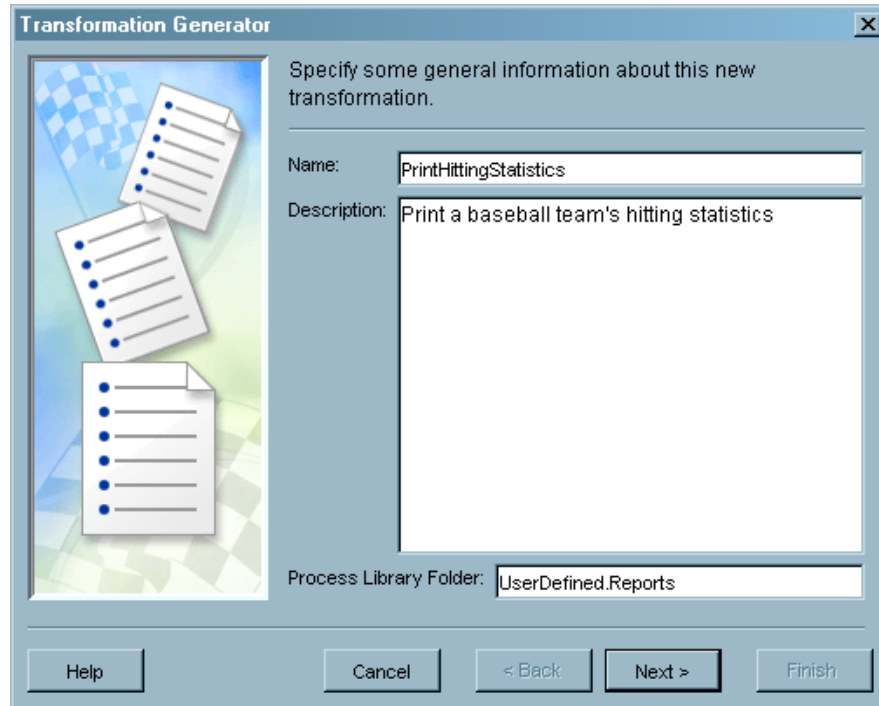
Perform the following steps to display the wizard that will guide you through the process of creating a user-defined SAS code transformation template.

- 1 From the SAS ETL Studio desktop, select

Tools

▶
Transformation Generator

from the menu bar. The first window of the wizard is displayed, as shown in the following display.

Display 9.15 First Window in the Transformation Generator WizardThe image shows a Windows-style dialog box titled "Transformation Generator". On the left is a graphic of three overlapping document icons. The main area contains the text "Specify some general information about this new transformation." followed by three input fields: "Name:" with the value "PrintHittingStatistics", "Description:" with the value "Print a baseball team's hitting statistics", and "Process Library Folder:" with the value "UserDefined.Reports". At the bottom are five buttons: "Help", "Cancel", "< Back", "Next >", and "Finish".

Transformation Generator

Specify some general information about this new transformation.

Name: PrintHittingStatistics

Description: Print a baseball team's hitting statistics

Process Library Folder: UserDefined.Reports

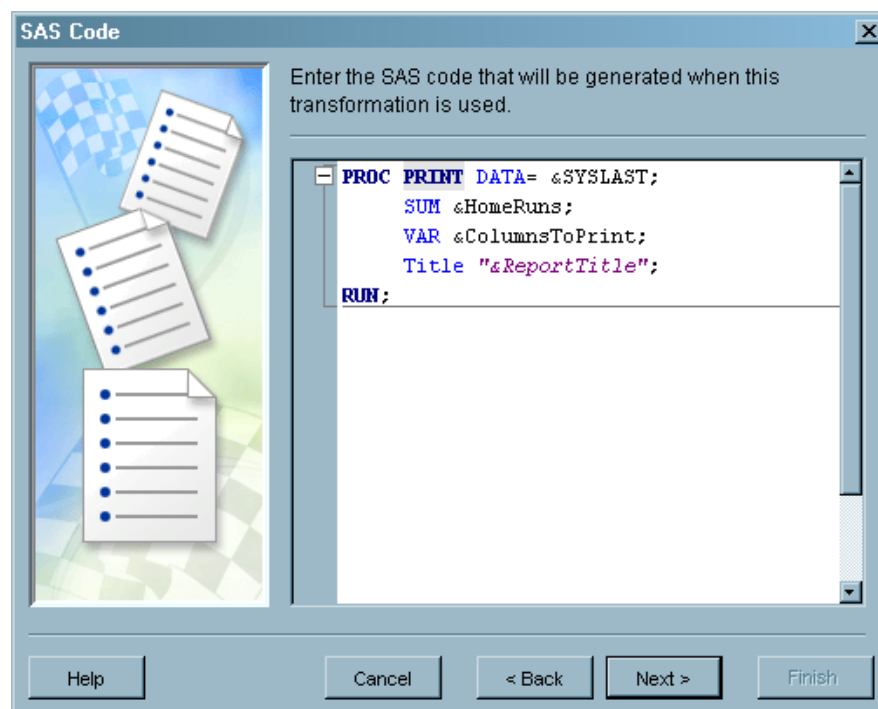
Help Cancel < Back Next > Finish

- 2 Enter a name and a description for the new transformation template, as shown in the previous display.
- 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 When you are finished with this window, click **Next**.

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

Specify SAS Code for the Transformation Template

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

Display 9.16 SAS Code Window

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

Another system variable, `&_OUTPUT`, is also available, but we have not used it in this example. `&_OUTPUT` enables a transformation in a process flow diagram to send its output to a temporary work table.

The other variables that are shown in Display 9.16 on page 126, such as `&ColumnsToPrint`, are user-defined variables. Any user-defined variables must be defined in the next window of the Transformation Generator wizard: the SAS Code Options window.

After you have finished writing your SAS code, click **Next**.

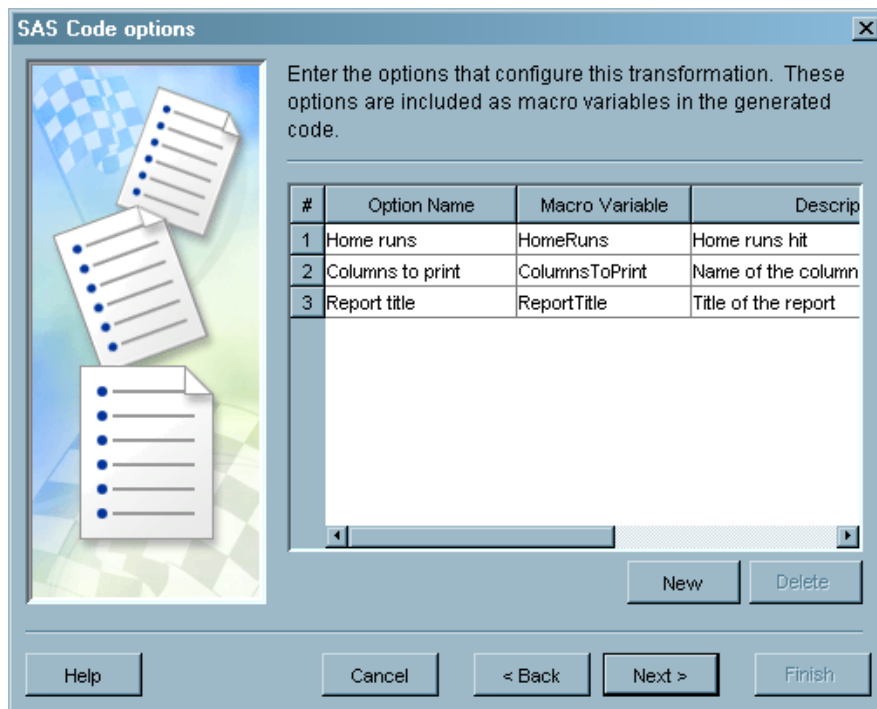
Define Any User-Defined Variables

In the SAS Code Options window of the wizard, define any user-defined variables that you used in the SAS Code window. The next table shows the values that would be entered for the user-defined variables in Display 9.16 on page 126.

Table 9.2 User-Defined Variables from the SAS Code Window

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

The next display shows the SAS Code Options window after the values in the previous table have been entered.

Display 9.17 SAS Code Options Window

The SAS Code Option window enables you to specify five types of variables:

- ☐ OPTION
- ☐ INPUTS SAS
- ☐ INPUTS SQL
- ☐ OUPUTS SAS
- ☐ OUTPUTS SQL

The variables that you define in the Code Options window will be used in the transformation template that you are creating. For example, variables of type OPTION will appear on the Options tab of the properties window for the PrintHittingStatistics template. Users will display the Options tab of the window and enter values for each option.

The INPUTS and OUTPUTS variables will appear on the Column Options tab of the properties window for the transformation template that you are creating. For example, users will display the Column Options tab of the properties window for the

PrintHittingStatistics template and select columns that correspond to the ColumnsToPrint variable of type INPUTS SAS.

To define any user-defined variables that you used in the SAS Code window, perform the following steps for each variable:

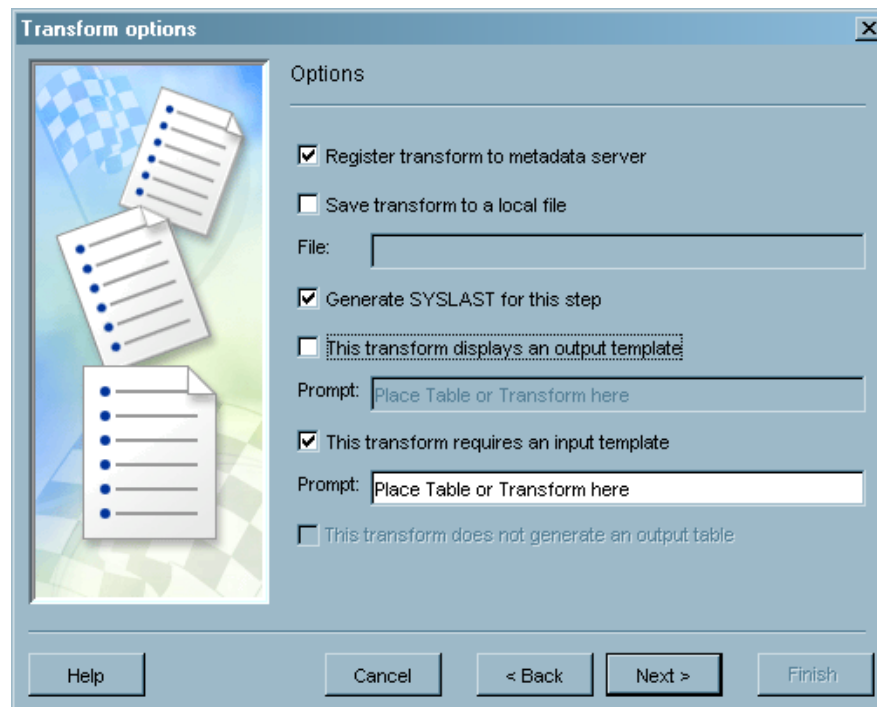
- 1 Click the **New** button. A new row displays in the options table.
- 2 In the **Option Name** field, enter a descriptive name. Replace the initial value (Untitled) by double-clicking that value to highlight it and then typing over the highlighted value.
- 3 In the **Macro Variable** field, enter the name of the macro variable as it appears in the SAS Code window.
- 4 In the **Description** field, enter a description of the variable.
- 5 In the **Type** field, double-click the current value and a down arrow displays. Click the down arrow to reveal a list of types, and select one of them.

When you are finished defining the user-defined variables in your transformation, click **Next**.

Specify the Remaining Options for the Transformation Template

Use the Transform Options window to specify the remaining options for your transformation template. The Transform Options window for the example transformation resembles the following display.

Display 9.18 Transform Options Window



Use the controls that are described as follows:

Register transform to metadata server—Select this check box if you want to save your transformation as a metadata object in the current metadata repository. Do

this if you want the transformation template to be available in the Process Library tree. For this example, assume that this option is selected.

Save transform to a local file—Select this check box if you want to save your transformation as an XML file on the local file system. Other SAS ETL Studio users can import transformations that are saved this way.

File—The name of, and path to, the XML file that was previously described.

Generate SYSLAST for this step—Determines whether the &SYSLAST macro variable is available to your code. Leave this check box selected unless your transformation does not require any input.

This transform displays an output template—If you select this check box, when a user drags your transformation to a process flow diagram, the template displayed includes a drop zone for an output table or transformation.

Prompt—The text that you want displayed in the output-table drop zone that was previously described.

This transform displays an input template—Analogous to the output template check box.

Prompt—Analogous to the Prompt control that was described previously.

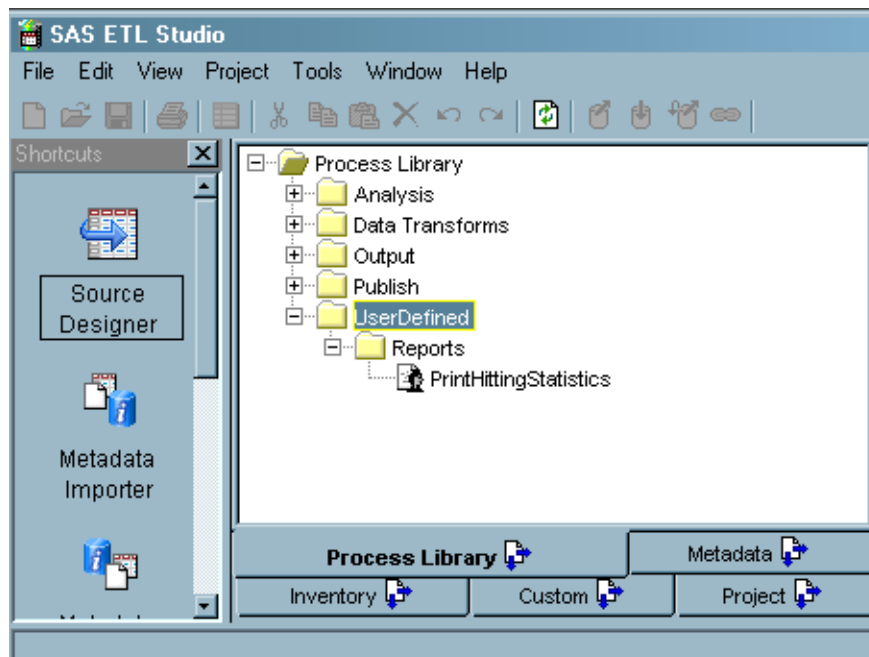
When you are finished defining options for your transformation, click **Next**.

Save the Transformation Template

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 or an XML file (or both), as specified in Display 9.18 on page 128.

For this example, assume that the SAS code transformation was saved to the current metadata repository. The template will now be visible in the Process Library tree, as specified in Display 9.12 on page 114. The following display illustrates the updated Process Library tree.

Display 9.19 Process Library Tree With User-Defined Transformation Template



The new template, `PrintHittingStatistics`, can now be used to create a job, as described in “Example: Using a SAS Code Transformation Template in a Job” on page 150.

Document Any Usage Details for the Template

The person who creates a user-written transformation template should document how it can be used to get the desired result. SAS ETL 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 9.13 on page 123.

- 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 9.14 on page 124, 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.

General Tasks for SAS Code Transformation Templates

Using a SAS Code Transformation Template in a Job

See “Example: Using a SAS Code Transformation Template in a Job” on page 150.

Identifying a SAS Code Transformation Template

SAS provides a number of SAS code transformation templates in the Process Library tree. Perform the following tasks to identify a transformation as a SAS code transformation:

- 1 From the SAS ETL Studio desktop, display the Process Library tree.
- 2 Open the folders to display the transformations. Right-click a transformation to display a pop-up menu. SAS code transformations have two unique pop-up menu options: **Edit Source** and **Transformation Export**.

The **Edit Source** option enables you to edit the SAS code for the selected transformation. For details about transformation export, see “Importing and Exporting SAS Code Transformations” on page 130.

Importing and Exporting SAS Code Transformations

The Transformation Generator wizard enables you to save a SAS code transformation in one of two ways. First, you can save it as a metadata object in your metadata repository. Such a transformation is said to be imported because it is ready for use in a SAS ETL Studio job. Second, you can save a SAS code transformation as an XML file. Such a transformation is said to be exported because it is not available for use in jobs.

The export feature enables you to create a custom transformation template and make it available to SAS ETL Studio users who are using different metadata repositories.

The sections following sections explain illustrate how to export a transformation at the time you define it, 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 SAS Code Transformation

As previously described, you can export a SAS code transformation at the time you create it or at a later time.

Perform these steps to export a new transformation when you are creating it:

- 1 Create the transformation using the Transformation Generator wizard, as described in “Example: Creating a SAS Code Transformation Template” on page 122.
- 2 In the Transform Options window, select **Save transform to a local file**, and enter a filename in the **File** text box. When you have finished running the wizard, the SAS code transformation is saved in XML format in the file you specified.

Perform these steps to export a transformation that already exists in your metadata repository:

- 1 Select the SAS code transformation in the Process Library tree.
- 2 From the SAS ETL 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 SAS Code Transformation

Perform these steps to import a SAS code transformation that has been saved in an XML file:

- 1 From the SAS ETL Studio desktop, select

Tools ► Transformation Import

from the menu bar. The Transformation Importer window displays. Build a list of XML files to import by performing the following 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.

Controlling Access to a SAS Code Transformation

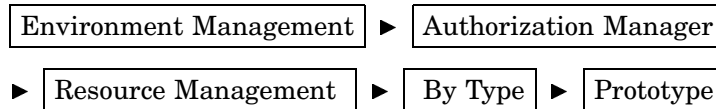
As described in “Planning Security for a Data Warehouse” on page 30, administrators should develop a security plan for controlling access to libraries, tables, and other resources that are associated with a data warehouse. Part of the security plan might be

to allow only certain users to use a SAS code transformation. This security must be set up using SAS Management Console.

Perform these steps to apply security to a SAS code transformation:

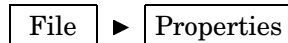
- 1 Start SAS Management Console. Open the metadata profile that contains the SAS code transformation for which access privileges must be defined.

- 2 Select



from the navigation tree.

- 3 In the Prototype list, select the transformation for which you want to provide security, then select



from the menu bar. The Properties window for the selected prototype displays.

- 4 From the properties window, click the **Authorization** tab.
- 5 Use the **Authorization** tab to define security for the transformation. Click the **Help** button on the tab for details.

Deleting Folders for SAS Code Transformations

As described in “Transformation Generator Wizard” on page 113, when you create a SAS code transformation template, you can specify a folder for that template in the Process Library tree. For example, in Display 9.12 on page 114, the **UserDefined** folder in the Process Library will contain a subfolder, **Reports**, which will contain the SAS code template, **PrintHittingStatistics**.

You cannot directly delete a folder that contains SAS code transformation templates. You must delete the templates in the folder, then exit SAS ETL Studio. The next time that you open the Process Library, the folder will be gone.

Perform these steps to delete a folder that contains SAS code transformation templates:

- 1 In the Process Library tree, delete all SAS code transformation templates in the folder. To delete a template, select its icon, then select



from the menu bar. (You cannot delete Java transformation templates.)

- 2 Exit SAS ETL Studio.
- 3 Start SAS ETL Studio and view the Process Library tree. The empty folder is no longer in the Process Library tree.

Additional Information about Jobs

Chapter 10, “Loading Targets in a Data Warehouse or Data Mart,” on page 135 consists of example jobs that could be created for the data warehouse that is described in Chapter 4, “Example Data Warehouse for Orion Star Sports & Outdoors,” on page 31.

The online help for SAS ETL Studio provides additional information about how to maintain jobs. Perform these steps to display the relevant help topics:

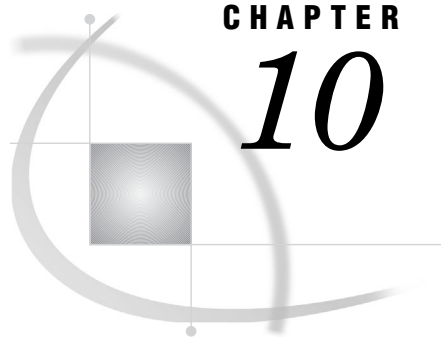
- 1 From the SAS ETL Studio menu bar, select

Help ► Contents

The online help window displays.

- 2 In the left pane of the help window, select

Working with Jobs ► Maintaining Jobs



CHAPTER

10

Loading Targets in a Data Warehouse or Data Mart

<i>Overview</i>	135
<i>Example: Creating a Job That Joins Two Tables and Generates a Report</i>	135
<i>Preparation</i>	136
<i>Start SAS ETL Studio and Open the Appropriate Metadata Profile</i>	136
<i>Check Out Any Metadata That Is Needed</i>	136
<i>Create and Populate the New Job</i>	137
<i>Configure the SQL Join Transformation</i>	141
<i>Configure the Columns in the Target and the Loader</i>	144
<i>Configure the Publish to Archive Transformation</i>	146
<i>Run and Troubleshoot the Job</i>	147
<i>Verify the Job's Outputs</i>	148
<i>Check In the Job</i>	150
<i>Example: Using a SAS Code Transformation Template in a Job</i>	150
<i>Preparation</i>	150
<i>Start SAS ETL Studio and Open the Appropriate Metadata Profile</i>	151
<i>Check Out Any Metadata That Is Needed</i>	152
<i>Create and Populate the New Job</i>	152
<i>Update the Template as Necessary</i>	154
<i>Run and Troubleshoot the Job</i>	155
<i>Verify the Job's Outputs</i>	156
<i>Check In the Job</i>	156

Overview

After you have entered metadata for sources and targets, and you know the basics of SAS ETL Studio jobs as described in Chapter 9, “Introduction to SAS ETL Studio Jobs,” on page 99, you are ready to load the targets in your data warehouse or data mart. Use the examples in this chapter, together with the general steps that are described in the previous chapter, to create jobs that will load targets.

Example: Creating a Job That Joins Two Tables and Generates a Report

This example demonstrates one way to use the New Job wizard, the Process Library, and the Process Designer window to enter metadata for a job. The example describes one way to create a report that is needed for the example data warehouse, as described in “Example: Creating a SAS Code Transformation Template” on page 122.

Preparation

Assume the following about the job in the current example:

- A data warehouse project plan identified the need for a report that ranks salespeople by total sales revenue. The report will be produced by a SAS ETL Studio job. The job will combine sales information with human resources information. A total revenue number will be summarized from individual sales. A new target table will be created, and that table will be used as the source for the creation of an HTML report.
- Metadata for both source tables in the job (ORGANIZATION_DIM and ORDER_FACT) is available in a current metadata repository.
- Metadata for the main target table in the job (Total_Sales_By_Employee) is available in a current metadata repository. “Example: Specifying Metadata for a Target Table in SAS Format” on page 89 describes how the metadata for this table could be specified. As described in that topic, the columns in Total_Sales_By_Employee were modeled after the columns in one source table, ORGANIZATION_DIM. However, when the job is fully configured in this example, the Total_Sales_By_Employee table will contain selected columns from both source tables: ORGANIZATION_DIM and ORDER_FACT.
- All sources and targets are in SAS format and are stored in a SAS library called Ordetail.
- Metadata for the Ordetail library has been added to the main metadata repository for the example data warehouse. For details about libraries, see “Entering Metadata for Libraries” on page 48.
- The main metadata repository is under change management control. For details about change management, see “Working with Change Management in SAS ETL Studio” on page 21.
- You have selected a default SAS application server for SAS ETL Studio, as described in “Select a Default SAS Application Server” on page 58.

Start SAS ETL Studio and Open the Appropriate Metadata Profile

Perform the following steps to begin work in SAS ETL Studio:

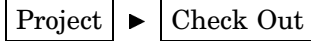
- 1 Start SAS ETL Studio as described in “Start SAS ETL Studio” on page 55.
- 2 Open the appropriate metadata profile as described in “Open a Metadata Profile” on page 58. For this example, the appropriate metadata profile would specify the project repository that will enable you to access metadata for the required sources and targets: ORGANIZATION_DIM, ORDER_FACT, and Total_Sales_By_Employee.

Check Out Any Metadata That Is Needed

To add a source or a target to a job, the metadata for the source or target must be defined and available in the Project tree. In the current example, assume that the metadata for the relevant sources and targets must be checked out. The following steps would be required:

- 1 On the SAS ETL Studio desktop, select the Inventory tree.
- 2 In the Inventory tree, open the **Tables** folder.

- 3 Select all source tables and target tables that you want to add to the new job: ORGANIZATION_DIM, ORDER_FACT, and Total_Sales_By_Employee.
- 4 Select



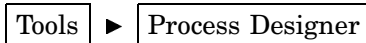
from the menu bar. The metadata for these tables will be checked out and will appear in the Project tree.

The next task is to create and populate the job.

Create and Populate the New Job

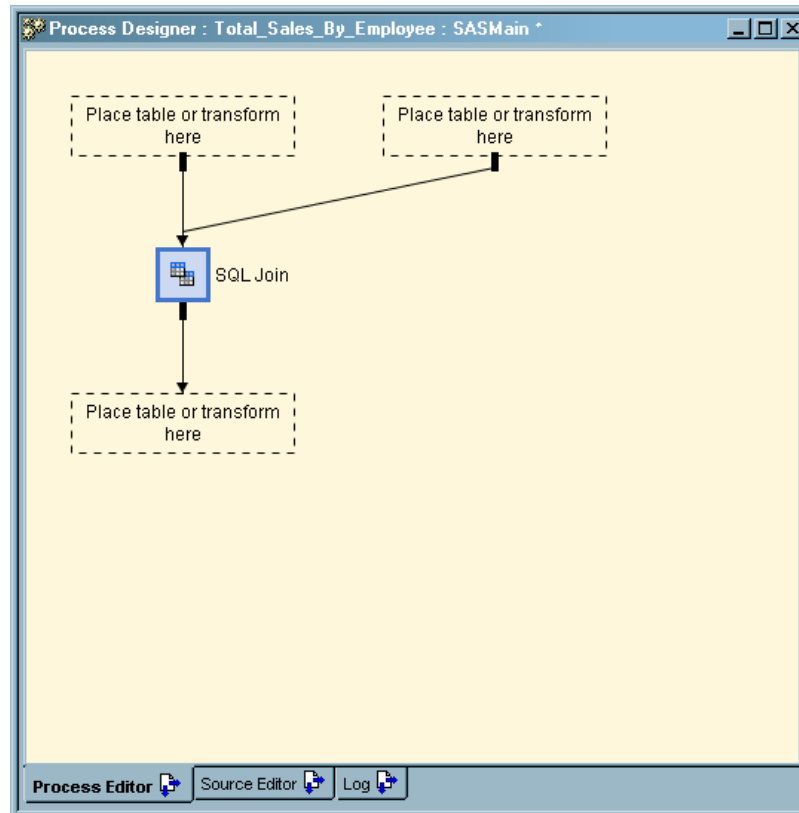
With the relevant sources and targets checked out in the Project tree, follow these steps to create and populate a new job. To “populate a job” means to create a complete process flow diagram, from sources, through transformations, to targets.

- 1 From the SAS ETL Studio desktop, select

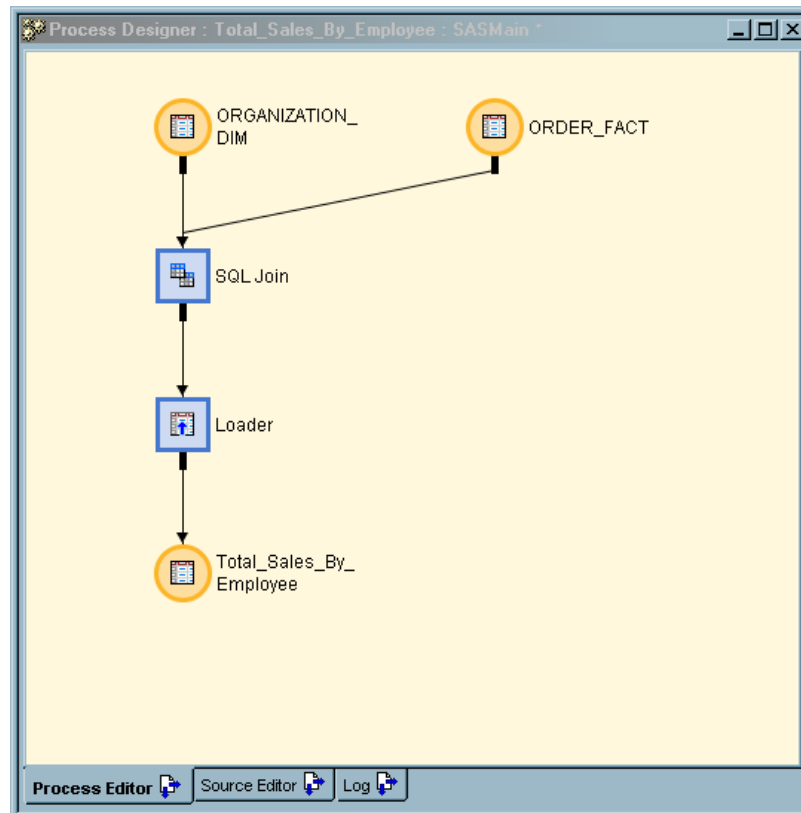


from the menu bar. The New Job wizard is displayed.

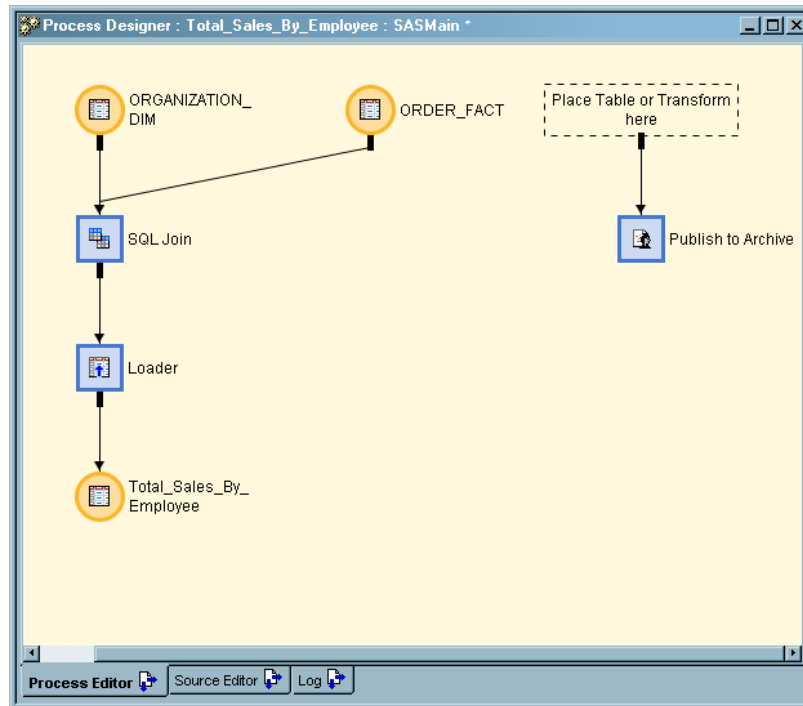
- 2 Enter a name and description for the job. Type the name **Total_Sales_By_Employee**, press the Tab key, then enter the description **Generates a report that ranks salespeople by total sales revenue**.
- 3 Click **Finish**. An empty job will open in the Process Designer window. The job has now been created and is ready to be populated with two sources, a target, a SQL Join transformation, and a Publish to Archive transformation.
- 4 From the SAS ETL Studio desktop, click the **Process** 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 is displayed with drop zones for two sources and one target, as shown in the following display.

Display 10.1 The New SQL Join Transformation in the New Job

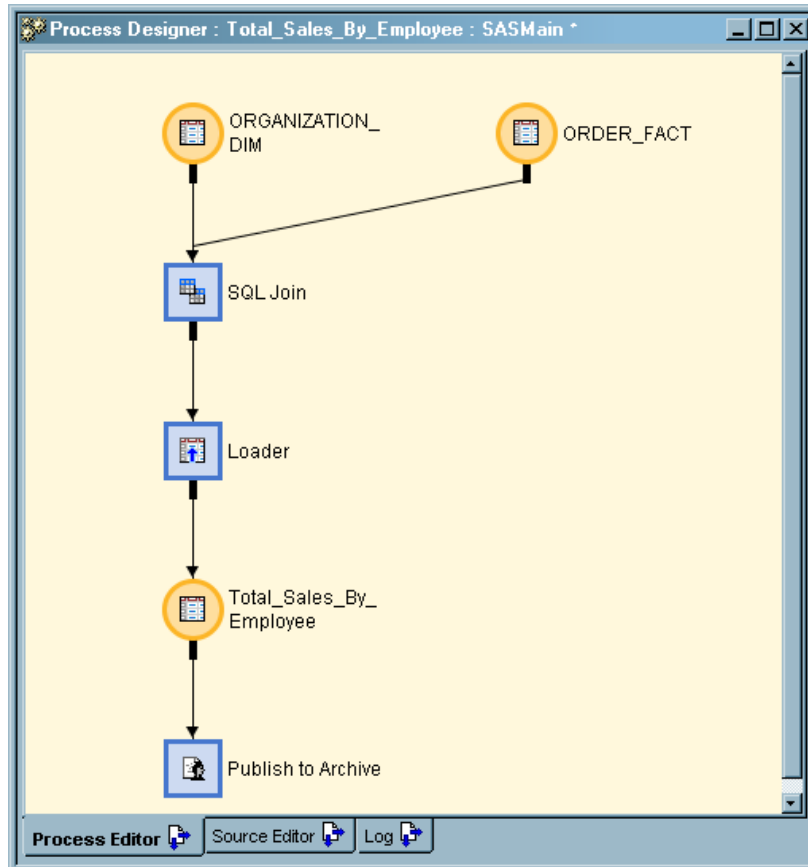
- 7 From the SAS ETL Studio desktop, click the **Project** tab to display the Project tree. You will see the new job and the three tables that you checked out.
- 8 In the Project tree, click and drag the **ORGANIZATION_DIM** table into one of the two input drop zones in the Process Designer window, then release the mouse button. The **ORGANIZATION_DIM** table appears as a source in the new job.
- 9 Repeat the preceding step to identify the **ORDER_FACT** table as the second of the two sources in the new job.
- 10 Click and drag the table **Total_Sales_By_Employee** 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 10.2 Sources and Targets in the Example Job

- 11 From the SAS ETL Studio desktop, click the **Process** tab to display the Process Library.
- 12 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 a input drop zone appear in the Process Designer.

Display 10.3 Example Job with Publish to Archive

- 13 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 cursor to identify the target as the source for **Publish to Archive**, as shown in the following display.

Display 10.4 Target Table Used as the Source for Publish to Archive

The job now contains a complete process flow diagram, from sources, through transformations, to targets. The next task is to update the default metadata for the transformations and the target.

Configure the SQL Join Transformation

The example job now contains the necessary sources, target, and transformations. Follow these steps to configure the SQL Join transformation.

- 1 In the Process Designer window, select the **SQL Join** transformation object, then select

File ► **Properties**

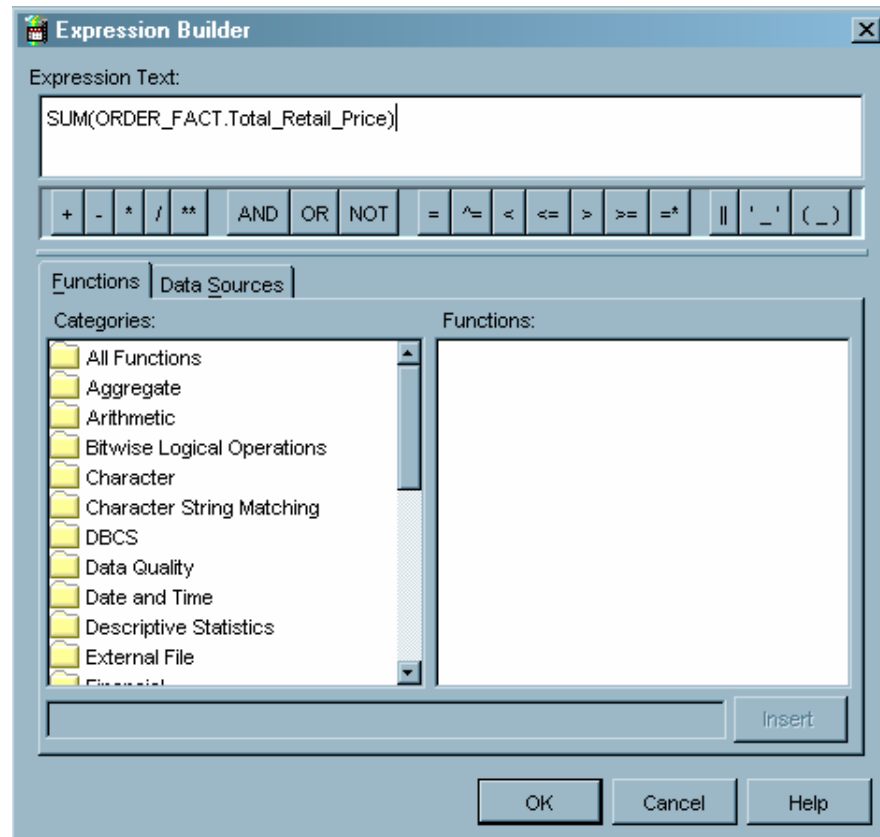
from the menu bar. A properties window is displayed.

- 2 Click the **SQL** tab. Note that all columns from both source tables are included in the join operation by default.
- 3 Click the **Mapping** tab. Click on the column named **Employee_Country** and press the Delete key. The **Employee_Country** column and mapping are removed. This column is not needed in the report.
- 4 In the target table on the right of the **Mapping** tab (a temporary work table), retain the **Company** column. Select the column named **Department**. Press the Delete key twice to delete the **Department** column and the **Section** column.

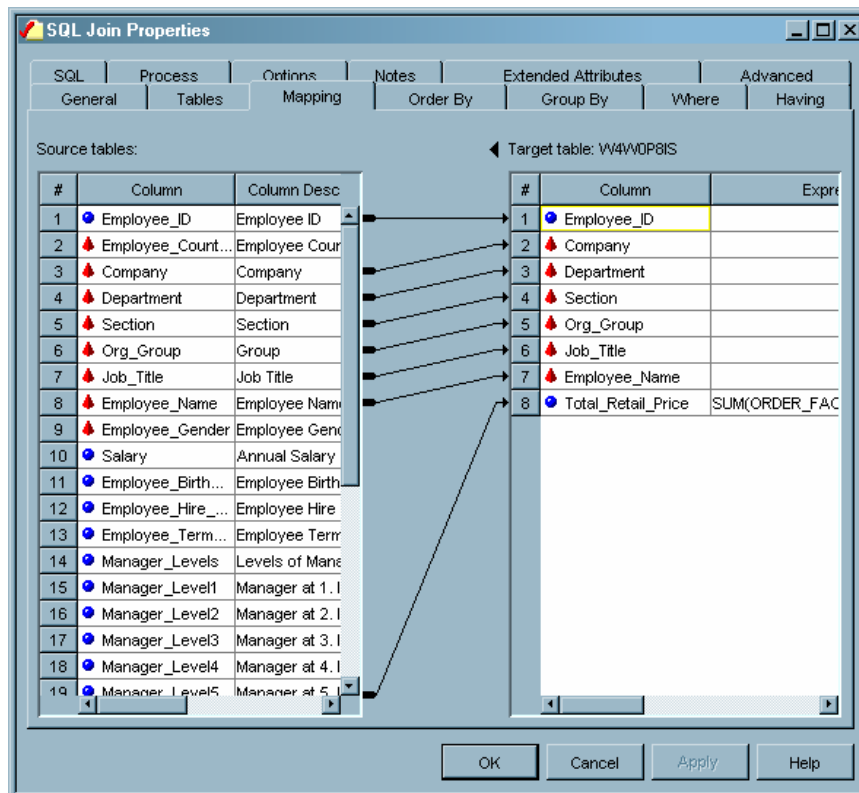
- 5 In the target table on the right, retain the **Org_Group** column, the **Job_Title** column, and the **Employee_Name** column. Select the **Employee_Gender** column. Delete the next 20 columns. Retain the **Total_Retail_Price** column, which will be summarized to create the total revenue number for each salesperson.
- 6 In the target table on the right, select the **CostPrice_Per_Unit** column. Delete the last two columns. The temporary target now contains only the columns that are needed in the report. Eliminating extraneous columns at this early stage maximizes the job's run-time performance.
- 7 In the target table on the right, click twice in the **Expression** column for **Total_Retail_Price**. Then click again in the icon that appears at 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.
- 8 In the Expression Builder, enter the following expression and click **OK**, as shown in the following display. The Expression Builder window closes and the expression appears in the Expression column of the **Mapping** tab.

```
SUM(ORDER_FACT.Total_Retail_Price)
```

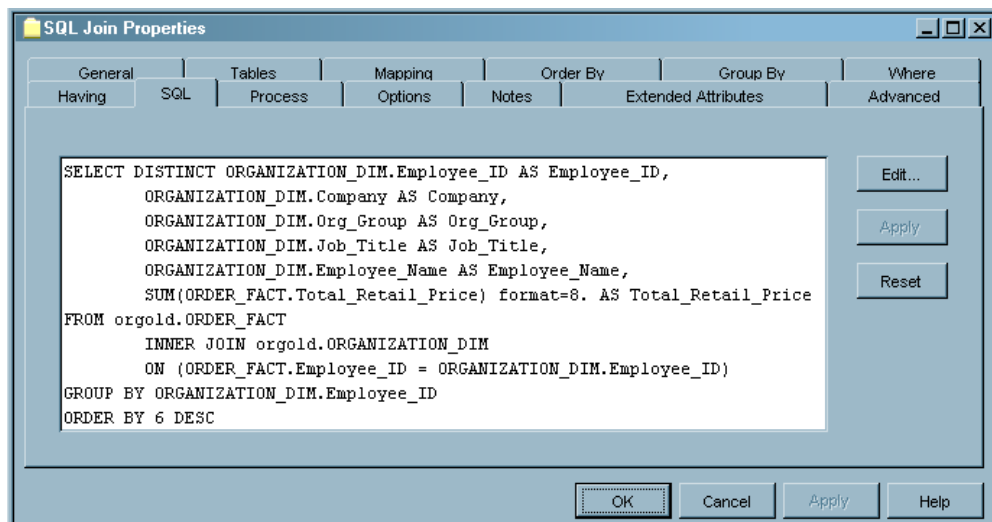
Display 10.5 SUM Statement in the Expression Builder



The following display shows the configuration of the **Mapping** tab.

Display 10.6 Mapping Source Columns in the SQL Join Transformation

- 9 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. In the SQL code that is shown in the following display, note that the number of target columns has been reduced to six, and a SUM expression has been added for the column **Total_Retail_Price**.

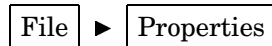
Display 10.7 SQL Code Configured Automatically

- 10 The SQL Join transformation is now ready. Click **OK** to save input and close the properties window.

Configure the Columns in the Target and the Loader

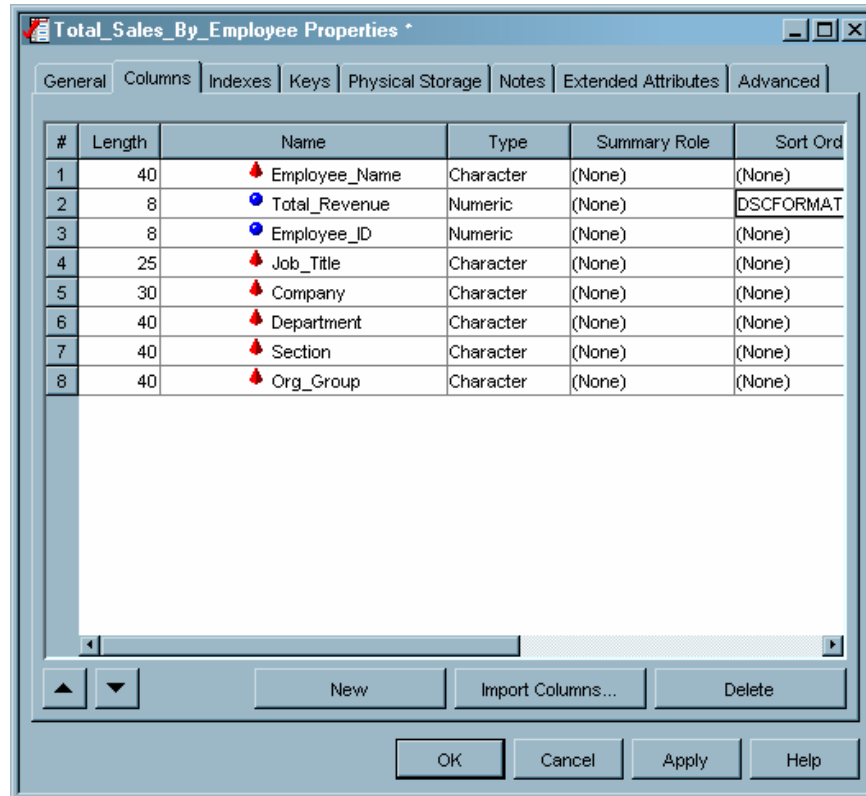
In our example job, the SQL Join transformation is now ready to run. Follow these steps to configure the target table `Total_Sales_By_Employee`.

- 1 In the Process Designer window, select the target **Total_Sales_By_Employee**, then select



from the menu bar. A properties window is displayed.

- 2 In the properties window, click the **Columns** tab.
- 3 Click the column name **Total_Retail_Price**. Change the name to **Total_Revenue**. The new name is a better representation of the newly summarized data.
- 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 sorts rows in descending order of the formatted value of the **Total_Revenue** column.
- 6 Reorder the columns by selecting rows in the list view and clicking the up or down arrows. The end result is a column order that formats the report for easy reading, as shown in the following display. When the column order is ready, the target is ready. Click **OK** to save input and close the properties window.

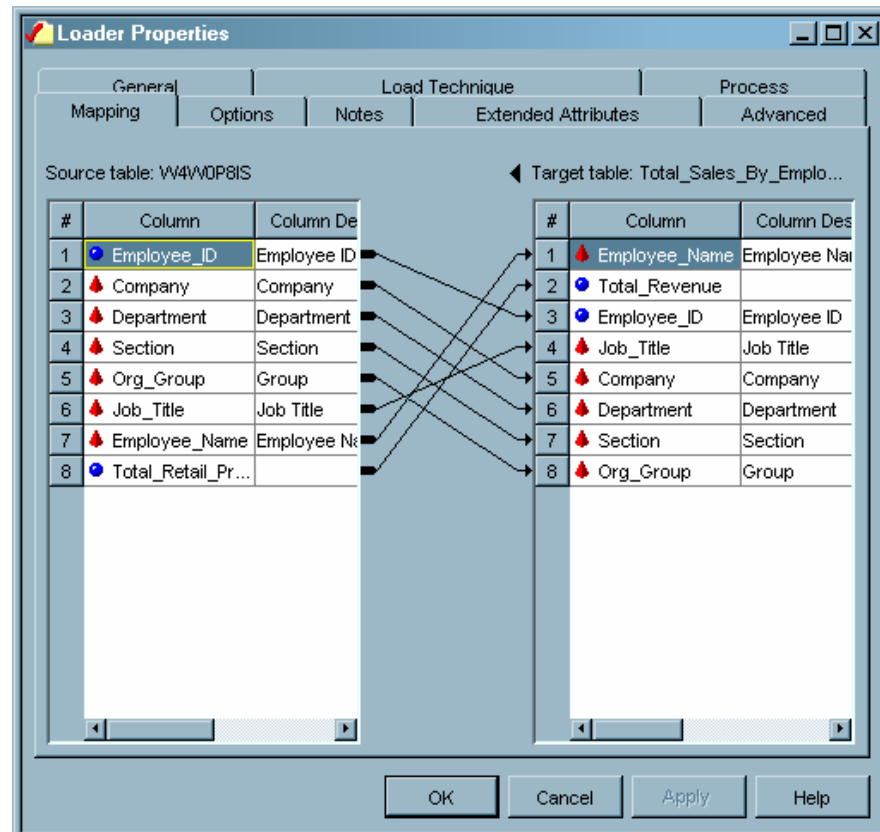
Display 10.8 Configured Target Columns

- 7 In the Process Designer window, select the **Loader** transformation, then select

File ► **Properties**

from the menu bar. A properties window is displayed.

- 8 In the properties window, click the **Mapping** tab to confirm that the columns of the SQL Join transformation are correctly mapped to the columns of the target, as shown in the following display.

Display 10.9 Column Mapping in the Loader

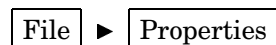
- 9 In the Properties window, click the **Load Technique** tab. Select the **Drop Target** radio button to replace the physical table each time the job is run.

The Loader is now configured and is ready to run.

Configure the Publish to Archive Transformation

The example job is now fully configured through the SQL Join and Loader transformations, and through the target table. Follow these steps to configure HTML output using the Publish to Archive transformation. 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.

- 1 In the Process Designer window, select the **Publish to Archive** transformation, then select



from the menu bar. A properties window is displayed.

- 2 In the properties window, click the **Options** tab. Type in values for the fields that are shown in the following display.

Display 10.10 Options in the Publish to Archive Transformation

Option Name	Option Value
Create SYSLAST Macro Variable	YES
System options	
PROC PRINT options	
Path and filename for report	WD9585\public\salesRank.html
First title on the report	Sales Performance
Second title on the report	Ranking Salespersons by Total Revenue
Descriptive name of the package	Sales Ranking Table, in HTML
Package Name/Value pairs	
Report Name/Value pairs	
Path to the Archive	WD9585\public\
Name of the Archive	salesRankArchive

Buttons: OK, Cancel, Apply, Help

- 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 and Troubleshoot the Job

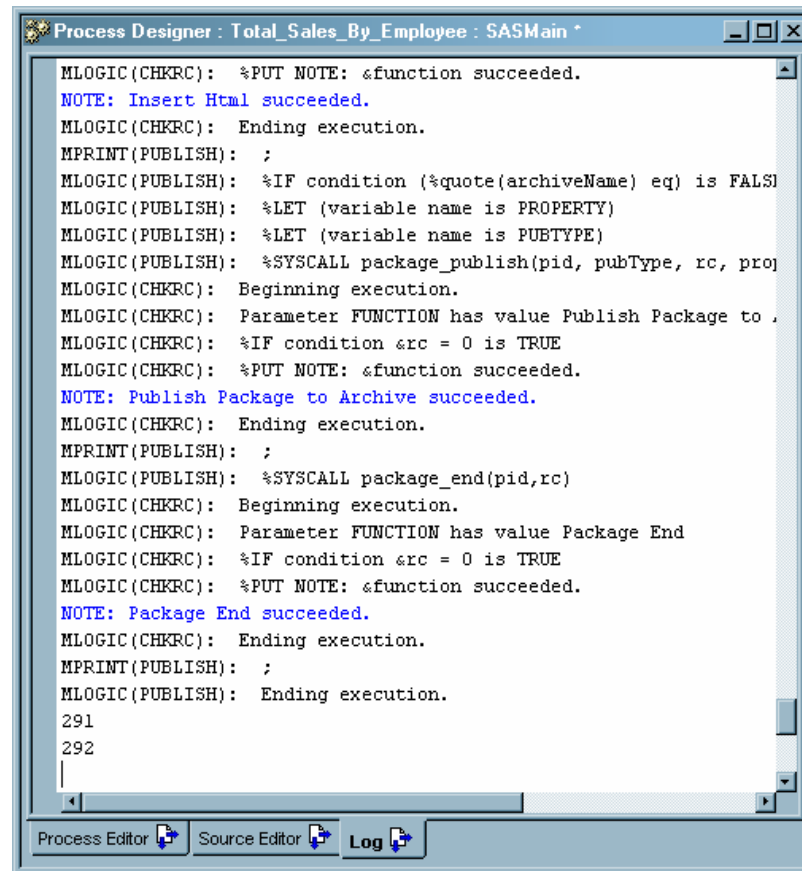
After the metadata for a job is complete, you must submit the job for execution in order to create targets on the file system.

- 1 With the job displayed in the Process Designer window, select

Process ► **Submit**

from the menu bar. SAS ETL 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.

- 2 If a pop-up error message appears, or if you simply want to look at the log for the completed job, click the **Log** tab in the Process Designer window.
- 3 In 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 10.11 Log Tab with Text from the Example Job

The code that was executed for the job is available in the **Source Code** tab of 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 Job's Outputs

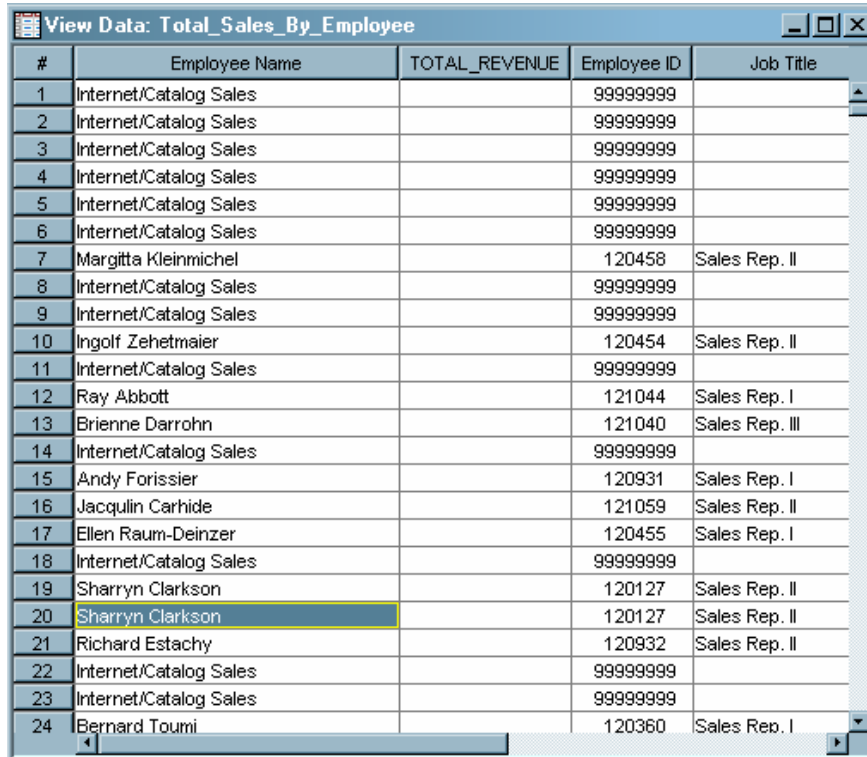
After the job runs without error and has been saved, you should confirm that the target(s) contain the data you need, in the format that best communicates the purpose of the target(s). In the current example, the main target table is the **Total_Sales_By_Employee** table. The example job also creates an HTML report.

- 1 To view the data in the **Total_Sales_By_Employee** target, select the target, then select

View ► View Data

from the menu bar. The data in the target is displayed in a View Data window, as shown in the following display.

Display 10.12 Viewing the Target



#	Employee Name	TOTAL_REVENUE	Employee ID	Job Title
1	Internet/Catalog Sales		99999999	
2	Internet/Catalog Sales		99999999	
3	Internet/Catalog Sales		99999999	
4	Internet/Catalog Sales		99999999	
5	Internet/Catalog Sales		99999999	
6	Internet/Catalog Sales		99999999	
7	Margitta Kleinmichel		120458	Sales Rep. II
8	Internet/Catalog Sales		99999999	
9	Internet/Catalog Sales		99999999	
10	Ingolf Zehetmaier		120454	Sales Rep. II
11	Internet/Catalog Sales		99999999	
12	Ray Abbott		121044	Sales Rep. I
13	Brienne Darrohn		121040	Sales Rep. III
14	Internet/Catalog Sales		99999999	
15	Andy Forissier		120931	Sales Rep. I
16	Jacquelin Carhide		121059	Sales Rep. II
17	Ellen Raum-Deinzer		120455	Sales Rep. I
18	Internet/Catalog Sales		99999999	
19	Sharryn Clarkson		120127	Sales Rep. II
20	Sharryn Clarkson		120127	Sales Rep. II
21	Richard Estachy		120932	Sales Rep. II
22	Internet/Catalog Sales		99999999	
23	Internet/Catalog Sales		99999999	
24	Bernard Toumi		120360	Sales Rep. I

- 2 Confirm that the target contains the data you need, in the format that best communicates the purpose of the target.
- 3 To display the HTML report, open the output file. In this case, the example generated a file in the public directory on the SAS application server. The file specification is as follows:

\\D9585\public\salesRank.html

The following display shows how the example file appears in a Web browser.

Display 10.13 HTML Report Generated by the Example Job


Obs	Employee_Name	Total_Revenue	Employee_ID	Job_Title	Company	Org_Group
1	Internet/Catalog Sales	\$2,159,323.48	99999999		Logistics	Internet/Catalog Sales Management
2	Christelle Bourrier	\$289,616.15	120359	Sales Rep. II	Orion France	Clothes
3	Agnes de Fourtou	\$271,983.27	120361	Sales Rep. II	Orion France	Clothes
4	Inés Niqui Salvat	\$265,715.66	120836	Sales Rep. IV	Orion Spain	Clothes
5	Brienne Darrohn	\$264,824.70	121040	Sales Rep. III	Orion USA	Clothes
6	Joseph Robbin-Coker	\$262,429.76	121042	Sales Rep. III	Orion USA	Clothes

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, you can check in the job.

Check In the Job

To check in a job in the Project tree:

- 1 In the Project tree, select the repository icon.
- 2 On the SAS ETL Studio desktop, select

Project ► Check In Repository

from the menu bar. All of the objects in the project repository are checked in to the change-managed repository.

Example: Using a SAS Code Transformation Template in a Job

This example demonstrates how a user-written SAS code transformation template can be used in a job. This example is based on the PrintHittingStatistics template that is described in “Example: Creating a SAS Code Transformation Template” on page 122.

Preparation

Assume the following about the job in the current example:

- A data warehouse project plan identified the need for a report that displays hitting statistics for baseball teams. The following display shows the kind of output that is desired.

Display 10.14 Example Hitting Report

1
13:29 Monday, November 3, 2003

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	

- The input for the report 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 10.15 Contents of Table: TigersHitting2002

View Data: TigersHitting2002												
#	NAME	G	AB	R	H	VAR1	VAR2	HR	TB	RBI	BB	SO
1	Smithy Jones	158	548	90	179	35	1	26	294	100	107	89
2	Gary Troy	135	492	82	151	26	0	25	252	84	72	53
3	Rafael Fernando	154	636	95	175	31	8	8	246	47	43	114
4	Andy Hitfield	154	560	91	148	34	0	35	287	94	83	135
5	Vinny Toredo	143	543	56	126	23	2	12	189	61	22	69

- Metadata for the source table, a SAS dataset called TigersHitting2002, is available in a current metadata repository.
- The report will be produced by a SAS ETL Studio job, using the PrintHittingStatistics transformation template. The template has already been created as described in “Example: Creating a SAS Code Transformation Template” on page 122. Usage details for the template have been documented, as described in “Document Any Usage Details for the Template” on page 130.
- Output for the report will be sent to the Output tab of 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 105.
- The main metadata repository is under change management control. For details about change management, see “Working with Change Management in SAS ETL Studio” on page 21.
- You have selected a default SAS application server for SAS ETL Studio, as described in “Select a Default SAS Application Server” on page 58.

Start SAS ETL Studio and Open the Appropriate Metadata Profile

Perform the following steps to begin work in SAS ETL Studio:

- 1 Start SAS ETL Studio as described in “Start SAS ETL Studio” on page 55.
- 2 Open the appropriate metadata profile as described in “Open a Metadata Profile” on page 58. For this example, the appropriate metadata profile would specify the project repository that will enable you to access the PrintHittingStatistics transformation template and the metadata for the required source, TigersHitting2002.

Check Out Any Metadata That Is Needed

To add a source or a target to a job, the metadata for the source or target must be defined and available in the Project tree. In the current example, assume that the metadata for the relevant source must be checked out. The following steps would be required:

- 1 On the SAS ETL Studio desktop, select the Inventory tree.
- 2 In the Inventory tree, open the **Tables** folder.
- 3 Select the source table that you want to add to the new job: **TigersHitting2002**.
- 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 New Job

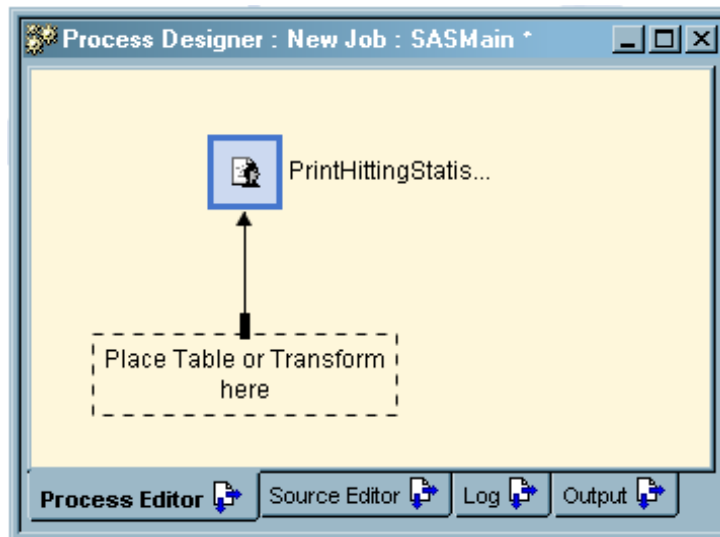
With the relevant source checked out in the Project tree, follow these steps to create and populate a new job. To “populate a job” means to create a complete process flow diagram, from sources, through transformations, to targets.

- 1 From the SAS ETL Studio desktop, select

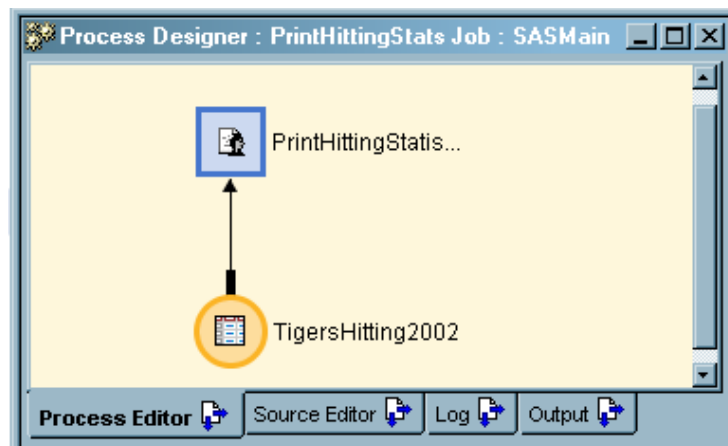
Tools ► Process Designer

from the menu bar. The New Job wizard is displayed.

- 2 Enter a name and description for the job. Type the name **PrintHittingStats Job**, press the Tab key, then enter the description **Generates a report that prints hitting statistics for a baseball team**.
- 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 From the SAS ETL 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 10.16 PrintHittingStatistics Template, Unpopulated

- 7 From the SAS ETL Studio desktop, click the **Project** tab to display the Project tree. You will see the new job and the source table that you checked out, **TigersHitting2002**.
- 8 In the Project 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.
- 9 Click and drag the **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 10.17 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 of the Process Designer window.

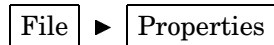
The next task is to update the default metadata in the process flow diagram.

Update the Template as Necessary

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 10.14 on page 151, 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. It is assumed that the steps for doing these tasks have been documented by the person who created the `PrintHittingStatistics` template.

Follow these steps to update the transformation in the process flow diagram:

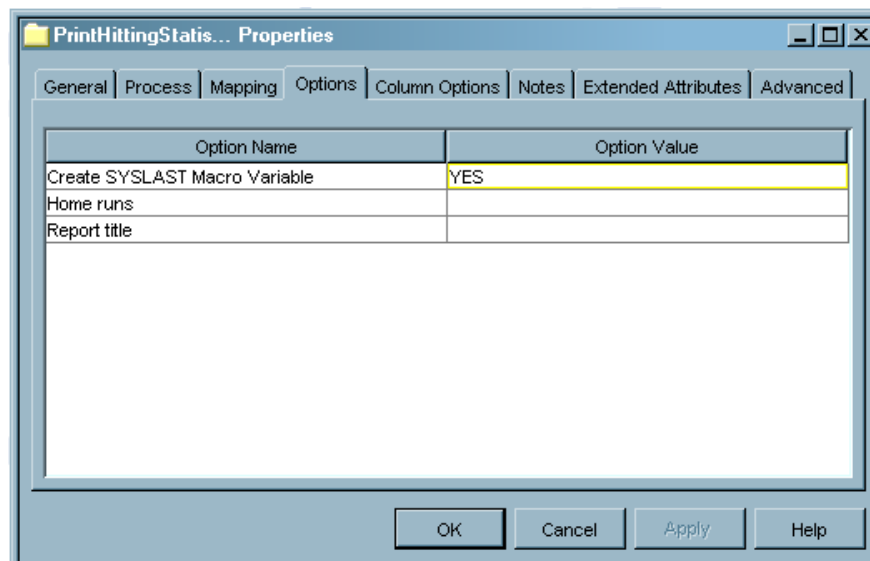
- 1 In the Process Designer window, select the **PrintHittingStatistics** transformation, then select



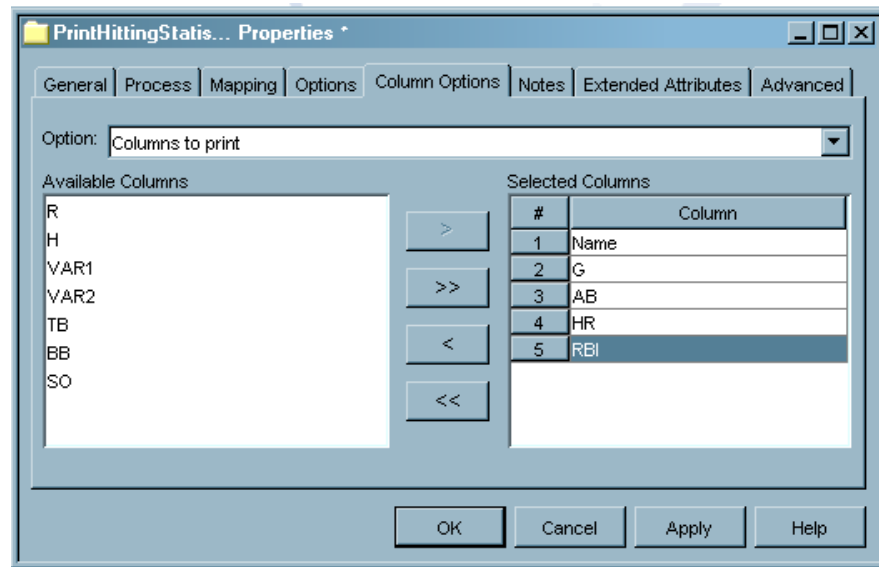
from the menu bar. A properties window is displayed.

- 2 Click the **Options** tab. The default options for the **PrintHittingStatistics** transformation are shown in the following display.

Display 10.18 Options Tab, PrintHittingStatistics Properties Window



- 3 In the **Home runs** field, enter the name of the source table column that contains home run values. In Display 10.15 on page 151, 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 10.14 on page 151, select the columns **Name**, **G**, **AB**, **HR**, and **RBI**. When you are finished, the **Column Options** tab should look similar to the following display.

Display 10.19 Column Options Tab, PrintHittingStatistics Properties Window

- 6 When you are finished entering metadata, click **OK** to save your changes. The job is now ready to run.

Run and Troubleshoot the Job

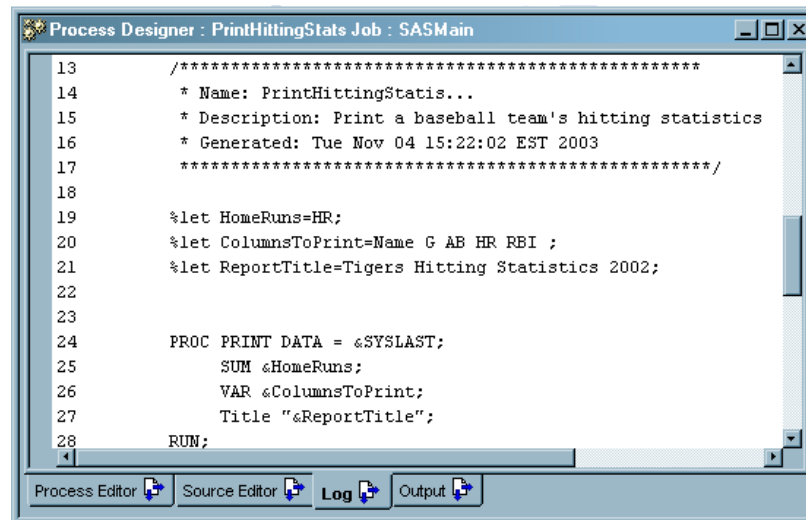
After the metadata for a job is complete, you must submit the job for execution in order to create targets on the file system.

- 1 With the job displayed in the Process Designer window, select



from the menu bar. SAS ETL 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.

- 2 If a pop-up error message appears, or if you simply want to look at the log for the completed job, click the **Log** tab in the Process Designer window.
- 3 In 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 10.20 Log Tab with Text from the Example Job

The code that was executed for the job is available in the **Source Code** tab of 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 Job's Outputs

After the job runs without error and has been saved, you should confirm that the target(s) contain the data you need, in the format that best communicates the purpose of the target(s). In the current example, the output is sent to the Output tab of the Process Designer window. When you click that tab, a report similar to the one shown in Display 10.14 on page 151 should be displayed.

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.

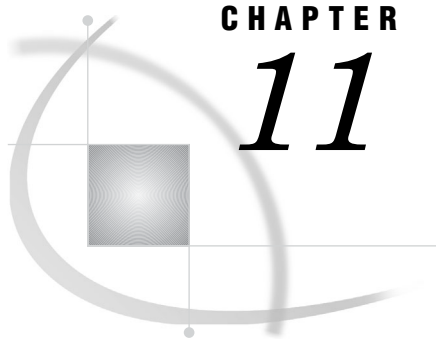
Check In the Job

To check in a job in the Project tree:

- 1 In the Project tree, select the repository icon.
- 2 On the SAS ETL Studio desktop, select

Project ► **Check In Repository**

from the menu bar. All of the objects in the project repository are checked in to the change-managed repository.



Creating Cubes

<i>Overview of Cubes</i>	159
<i>General Tasks for Cubes</i>	160
<i>Prerequisites for Cubes</i>	160
<i>Working under Change-Management Control</i>	160
<i>Using the Cube Designer to Create a Cube</i>	161
<i>Viewing the Data in a Cube</i>	161
<i>Updating a Cube or Its Metadata</i>	161
<i>Example: Building a Cube from a Star Schema</i>	162
<i>Preparation</i>	163
<i>Start SAS ETL Studio and Open the Appropriate Metadata Profile</i>	163
<i>Use the Cube Designer</i>	163
<i>Display the Cube Designer</i>	163
<i>Enter General Information</i>	164
<i>Select Input for the Cube</i>	164
<i>Select a Table for Drill-Through Reporting</i>	165
<i>Define Dimensions, Hierarchies, and Levels</i>	165
<i>Specify Measures (Columns) and Measure Details</i>	167
<i>Specify Member Properties</i>	168
<i>Specify Aggregations</i>	168
<i>Create the Cube</i>	169
<i>Check In the Cube</i>	169
<i>Example: Using the Source Editor to Submit User-Written Code for a Cube</i>	169
<i>Preparation</i>	170
<i>Write the Code</i>	170
<i>Submit the Code</i>	172
<i>Check In the Cube</i>	172
<i>Additional Information about Cubes</i>	173

Overview of Cubes

A cube is a logical set of data that is organized and structured in a hierarchical, multidimensional arrangement. It is a data store that supports Online Analytical Processing (OLAP).

When you define a cube, you define the dimensions and measures for the cube along with information about how aggregations should be created and stored. There are two main ways to create a SAS cube:

- Use the Cube Designer wizard in SAS ETL Studio or SAS OLAP Cube Studio to define and create the cube. The Cube Designer generates a long form of OLAP procedure code that stores the cube definition in a metadata repository. If you

specify the appropriate option, the wizard can submit a shorter form of OLAP procedure code to create the cube on the file system.

- Use the SAS OLAP procedure to create a cube. You can submit the OLAP procedure code interactively (using the Source Editor window in SAS ETL Studio or another SAS Program Editor), or you can submit the code in batch mode. The code stores the cube definition in a metadata repository, then creates the specified cube on the file system.

General Tasks for Cubes

Prerequisites for Cubes

A cube can be quite complex. Accordingly, someone who is familiar with OLAP design and the business goals for a particular cube should design the cube before you create it. For details about the design and structure of a cube, see the *SAS OLAP Server: Administrator's Guide*.

The metadata for the source tables that supply information to the cube must be available from a metadata repository. For examples of the kind of tables that can serve as inputs to a cube, see “What Are the Time and Place Dependencies of Product Sales?” on page 36.

The Cube Designer wizard does not require a connection to a SAS OLAP Server, but it does require an OLAP schema, a metadata object that is used to control access to a group of cubes. Accordingly, before using the Cube Designer in SAS ETL Studio, it is recommended that you perform the following tasks:

- A SAS OLAP Server is installed.
- Metadata for the installed SAS OLAP Server should be added to a metadata repository.
- An OLAP schema should be defined, and the appropriate SAS OLAP Server should be assigned to that schema. (You can also define the schema in the Cube Designer wizard).

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

If you use the SAS OLAP procedure to create a cube, see the *SAS OLAP Server: Administrator's Guide* for details about this procedure.

Working under Change-Management Control

Unless your user profile includes administrative privileges, you will be working under change management control. For a general description of how change management affects user tasks in SAS ETL Studio, see “Working with Change Management in SAS ETL Studio” on page 21.

When working with cubes in SAS ETL Studio, the main impacts of change management are as follows:

- 1 To update an existing cube, you must check out the cube.
- 2 Metadata for a new cube is added to the Project tree. At some point, you must check in new objects to the change-managed repository.

Using the Cube Designer to Create a Cube

Assume that the prerequisites that are described in “Prerequisites for Cubes” on page 160 have been met. The general steps for using the Cube Designer wizard to add a cube are as follows.

- 1 From the SAS ETL Studio desktop, select

Tools ► Target Designer

from the menu bar. The Target Designer selection window is displayed.

- 2 Select the **Cube Designer** and click Next.
- 3 Enter other metadata as prompted by the wizard.

For an example of how the Cube Designer can be used, see “Example: Building a Cube from a Star Schema” on page 162. For an alternative, see “Example: Using the Source Editor to Submit User-Written Code for a Cube” on page 169.

Viewing the Data in a Cube

You cannot view the contents of a cube in SAS ETL Studio. You can use Microsoft Excel or SAS Enterprise Guide to view the data in a cube. For details, see the *SAS OLAP Server: Administrator's Guide*.

Updating a Cube or Its Metadata

After a cube is built on the file system, you can update its metadata or the cube itself.

The properties window for a cube enables you to view or update some of its basic metadata. The basic metadata includes a descriptive name for the cube's metadata object, metadata for the user who is responsible for the cube, and a read-only table that shows the cube's structure. To update other cube metadata, or to update the physical cube, you must use the Cube Designer wizard.

Perform the following steps to update the metadata for a cube or to update the cube itself. The cube is assumed to be under change management control.

- 1 On the SAS ETL Studio desktop, display the Inventory tree.
- 2 In the Inventory tree, open the **OLAP** folder.
- 3 Select the cube to be updated, then select

Project ► Check Out

The metadata for the cube is checked out. A check mark is displayed next to the cube in the Inventory tree. An icon indicating a checked-out cube appears in the Project tree.

- 4 Display the Project tree, right-click the cube, then select the appropriate option from the pop-up menu.

Note that you must display the cube from the Project tree in order to update it. Displaying the cube from the Inventory tree enables browsing only.

The options that you can select from the pop-up menu include the following:

Properties	The properties window displays information about the cube and includes the following tabs:
------------	--

	General	The General tab displays the cube's name and description. It also lists users who have been assigned either owner or administrator responsibility for the cube.
	Extended attributes	The Extended attributes tab displays site-defined metadata that is not part of the standard metadata for cube. You can enter attribute information on this tab.
	Advanced	The Advanced tab displays the metadata information registered for the selected cube in the active metadata repository. The information on the Advanced tab is read-only.
	Structure	The Structure tab displays the cube structure. It has a standard navigational tree structure on the left side and a blank window on the right side. The navigational tree contains folders that represent the components in the selected cube, such as dimensions, hierarchies, measures, and aggregations. The information on the Structure tab is read-only.
Create		The selected cube metadata is re-read and the cube is re-created. The existing cube is overwritten.
Edit cube structure		The Cube Designer is displayed. You can then step through the Cube Designer windows to edit the metadata for the cube.
Manual Tuning		You can add new aggregations, modify user-defined aggregations, and delete aggregations for the cube from the Manual Tuning window.
Save PROC OLAP code		<p>The PROC OLAP code that is used to create the selected cube is saved in a text file. In the Path field on this window, enter a fully qualified path to the location of a text file. For example, you might enter</p> <pre>c:\finance_code.txt</pre> <p>for a cube that contains financial data.</p>
Delete		The cube metadata is deleted.
Group		You can specify a user-defined group for a selected object. The Select Group window allows you to select the user-defined group into which the current object should be placed.
Refresh		The cube metadata is re-read, and the cube is updated.

Example: Building a Cube from a Star Schema

This example demonstrates how to use the Cube Designer to create a cube that is based on a star schema. The example is based on the scenario that is described in “What Are the Time and Place Dependencies of Product Sales?” on page 36.

Preparation

For the current example, assume that the following statements are true::

- A warehouse project plan identified the need for a SAS cube to support OLAP reporting.
- The cube will be based on a star schema in which ORDER_FACT is the central fact table, and CUSTOMER_DIM, GEOGRAPHY_DIM, ORGANIZATION_DIM, and TIME_DIM are the dimension tables. For details about this star schema, see “Identifying Targets” on page 38.
- The star schema has already been created, and metadata for the star schema has already been added to a metadata repository.
- The metadata repository is under change management control. For details about cubes and change management, see “Working under Change-Management Control” on page 160.
- The prerequisites that are described in “Prerequisites for Cubes” on page 160 have been met.

Start SAS ETL Studio and Open the Appropriate Metadata Profile

Perform the following steps to begin work in SAS ETL Studio:

- 1 Start SAS ETL Studio as described in “Start SAS ETL Studio” on page 55.
- 2 Open the appropriate metadata profile as described in “Open a Metadata Profile” on page 58. For this example, the appropriate metadata profile would specify the project repository that will enable you to access metadata about the star schema.

You do not have to check out the star schema in order to specify it as the input to the cube. Accordingly, the next task is to display the Cube Designer and enter metadata as prompted by the wizard.

Use the Cube Designer

Perform these steps to create a cube using the Cube Designer.

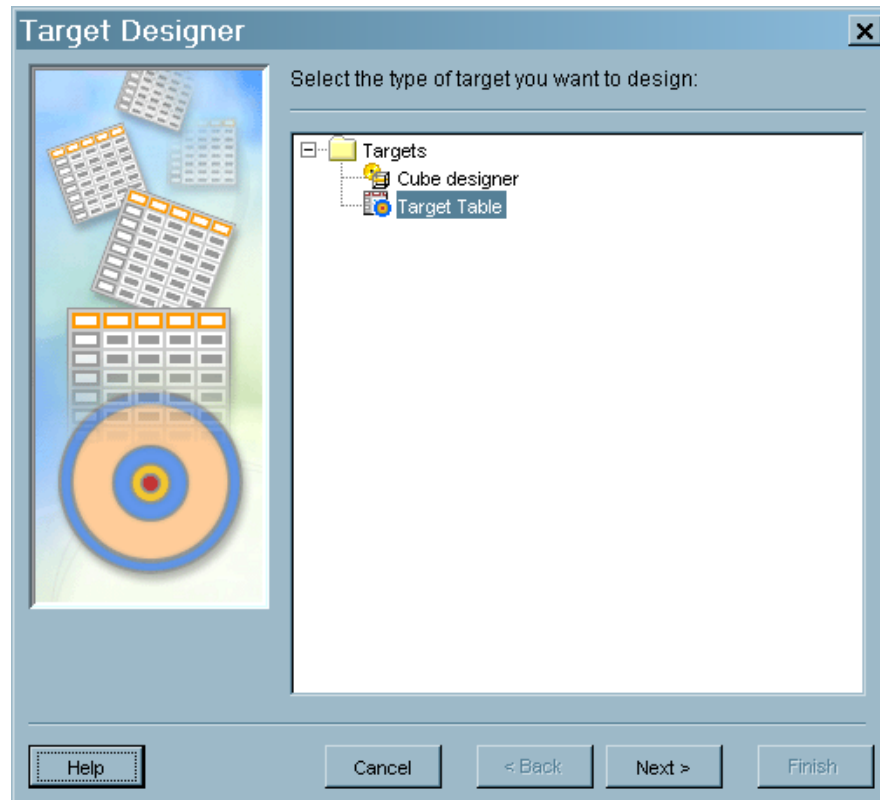
Display the Cube Designer

Perform these steps to display the Cube Designer:

- 1 From the menu bar on the SAS ETL Studio desktop, select



The Target Designer selection window is displayed.

Display 11.1 Target Designer Selection Window

- 2 Select the Cube Designer icon and click **Next**. The Cube Designer is displayed. The next task is to enter general information about the cube.

Enter General Information

In the Cube Designer—General window, enter the following information:

- ☐ Cube Name
- ☐ Description
- ☐ Repository
- ☐ OLAP Schema
- ☐ Path in file system to store the cube
- ☐ Input Type.

For input type, select **Star Schema**. Click **Next** when finished

The next task is to select the input for the cube.

Select Input for the Cube

In the Cube Designer—Input window, select a data source for your cube. For this example, select the **ORDER_FACT** table, which is the central fact table for a star schema. (If a source table does not exist for your data, you can select **Define Table**, and then define the source from which you will import metadata.)

Note: If the cube is built from a star schema, then the keys that link the dimension table and the fact table are also defined by using the DIMKEY= and FACTKEY= options. See the *SAS OLAP Server: Administrator's Guide* for further information. △

Click [Next](#) when finished. The next task is to select a table for drill-through reporting.

Select a Table for Drill-Through Reporting

In the Cube Designer—Drill-Through window, determine whether you will have a drill-through table. In this example, you will not use a drill-through table, so you can select the option **No table for Drill-Through**.

Click [Next](#) when finished. The next task is to define dimensions, hierarchies, and levels for the cube.

Define Dimensions, Hierarchies, and Levels

- 1 In the Cube Designer—Dimension Tables window, select dimension tables that are associated with the ORDER_FACT star schema that you specified as the data source for the cube. For this example, select the following tables:

- ☐ CUSTOMER_DIM
- ☐ GEOGRAPHY_DIM
- ☐ ORGANIZATION_DIM
- ☐ TIME_DIM.

- 2 Now that your basic metadata server and cube information has been entered, define the different dimensions and their respective levels and hierarchies. This example cube has these dimensions and levels:

- ☐ Time
 - ☐ Year_ID
 - ☐ Quarter
 - ☐ Month_Name
 - ☐ Week_Name
 - ☐ Date_ID.

For the Time dimension, the following star schema information is also included:

Table	TIME_DIM
Key	Date_ID
Fact Key	Order_Date

- ☐ Customers
 - ☐ Customer_Name
 - ☐ Customer_Age
 - ☐ Customer_Gender
 - ☐ Customer_Group
 - ☐ Customer_Type.

For the Customers dimension, the following star schema information is also included:

Table 11.1

Table	CUSTOMER_DIM
Key	Customer_Id
Fact Key	Customer_Id

- Geography
 - Continent_Name
 - Country
 - State
 - Region
 - Province
 - County
 - City.

For the Geography dimension, the following star schema information is also included:

Table 11.2

Table	GEOGRAPHY_DIM
Key	Street_Id
Fact Key	Street_Id

- Organization
 - Employee_Name
 - Job_Title
 - Salary
 - Gender
 - Company
 - Department
 - Org_Group
 - Section.

For the Organization dimension, the following star schema information is also included:

Table 11.3

Table	ORGANIZATION_DIM
Key	Employee_Id
Fact Key	Employee_Id

Define the dimensions for the cube. For each dimension, you define the dimension, its levels, and its hierarchies.

- At the Cube Designer—Dimensions window, select the **Add** button. This opens the Dimension Designer—General window. Enter the following information:
 - Dimension name
 - Caption
 - Description
 - Type of dimension (standard or time)
 - Sort order.

When you define the dimensions for a cube based on a star schema, you will need to provide additional information about the dimensions in the Dimension Designer—General window. On the Star Schema Dimension Tables Definition panel, enter the following information:

- Table
- Key
- Fact Key
- Data Set Options.
- Select the necessary dimension levels at the Dimension Designer—Levels window.
- Define properties such as format, time type, and sort order at the Dimension Designer—Level Properties window.
- Define hierarchies for the levels at the Dimension Designer—Define a Hierarchy window.
- Repeat this task for each dimension.

Note: You use the DIMENSION, HIERARCHY, and LEVEL statements here. For time-specific levels in a dimension, the LEVEL statement is required. Also, there can be only one time-specific dimension. △

Click **Next** when finished. The next task is to specify columns or measures for the cube.

Specify Measures (Columns) and Measure Details

- 1 In the Cube Designer—Selected Measures window, select the following columns and associated Sum statistics:
 - Total_Retail_Price /Sum
 - Quantity /Sum
 - CostPrice_Per_Unit /Sum
 - Discount /Sum.
- 2 Specify detail information for the measures. In the Cube Designer—Measure Details window, enter any necessary information for the different measures:
 - Caption
 - Format
 - Unit
 - Description.

For the measure Total_Retail_Price, enter a format value of DOLLAR12.2. For the measure CostPrice_Per_Unit, enter a format value of DOLLAR10.2.

Click **Next** when finished. The next task is to specify member property information for the levels in the cube.

Specify Member Properties

- 1 In the Cube Designer—Member Property window, select the **Add** button to create a new member property. In the Define a Member Property window, enter the following information about the member property:

- ☐ Name
- ☐ Level
- ☐ Column
- ☐ Format
- ☐ Caption Description
- ☐ Selected Hierarchies.

In this example, the following member properties are created:

Table 11.4

Property Name	Level	Column	Caption	Selected Hierarchy
WeekDay_Number_US	date	weekday_no	US WeekDay Number	
WeekDay_Number_EU	date	weekday_eu	EU WeekDay Number	
Week_Number_EU	week_name	week_no	EU Week Number	YWD
Month_Number	month_name	month_no	Month Number	YMD
Month_Number	month_name	month_no	Month Number	YQMD
Holiday_US	date	Holiday_US	US Holidays	

Click [Next](#) when finished. The next task is to specify aggregations for the cube.

Specify Aggregations

Aggregations are summaries of detailed data that is stored with a cube or referred by a cube. They can help reduce the build time that is required for the cube and contribute to faster query response.

- 1 In the Cube Designer—Generated Aggregations window, select the **Add** button to specify aggregations and associated levels. Order the levels for the aggregations to follow the hierarchy drill path. The aggregations include the following:

- ☐ RegionalCustomerUse
- ☐ QuarterlyCustomerUse
- ☐ YearlyCustomerUse
- ☐ WorldwideStaff
- ☐ WorldwideSalaries.

Note: When you create cubes in the Cube Designer, a default aggregation, which is the NWAY aggregation, is automatically created and listed in the Cube Designer—Generated Aggregations window. △

Click **Next** when finished. The next task is to review the metadata that you have entered and create the cube.

Create the Cube

In the Cube Designer—Finish window, select whether you want the cube to be physically created after the metadata is saved. When you click **Finish**, the metadata for the cube is always saved.

If you select **Save the metadata and create the cube**, the short form of the OLAP procedure code is generated along with the necessary LIBNAME statements and submitted to a SAS application server. You can also select whether to save the OLAP procedure code that is generated. At the Save PROC OLAP Code window, enter the file location where you want to save the resulting code.

If the cube you created is processed successfully and a cube is built, the cube will appear in the Project tree.

Note: When a SAS OLAP cube is created, a directory for that cube is also created. This directory is assigned the same name as the cube, but in uppercase letters. For example, when you save a cube in

```
c:\olapcubes
```

and name the cube

```
Campaigns
```

, the cube is saved in the directory

```
c:\olapcubes\CAMPAIGNS
```

. △

Check In the Cube

Under change management, new metadata objects are added to the Project tree on the SAS ETL Studio desktop. 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 (such as *Project: etlUser1*).
- 2 From the menu bar on the SAS ETL Studio desktop, select

Project ► **Check In Repository**

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

Example: Using the Source Editor to Submit User-Written Code for a Cube

This example demonstrates how to use the Source Editor window in SAS ETL Studio to submit user-written OLAP procedure code. The code will store the cube definition in a project repository, then create the specified cube on the file system. The user must then run SAS ETL Studio and check in the new cube, just as you would if you had used the Cube Designer to create the cube. The content and structure of the cube is the same

as the cube that is described in “Example: Building a Cube from a Star Schema” on page 162.

Preparation

For this example, assume that the following statements are true:

- A warehouse project plan identified the need for a SAS cube to support OLAP reporting.
 - The cube will be based on a star schema in which ORDER_FACT is the central fact table, and CUSTOMER_DIM, GEOGRAPHY_DIM, ORGANIZATION_DIM, and TIME_DIM are the dimension tables. For details about this star schema, see “Identifying Targets” on page 38.
 - The star schema has already been created, and metadata for the star schema has already been added to a metadata repository.
 - The metadata repository is under change management control. For details about cubes and change management, see “Working under Change-Management Control” on page 160.
 - The prerequisites that are described in “Prerequisites for Cubes” on page 160 have been met.
-

Write the Code

Use the SAS OLAP procedure to write a program that will store the cube definition in a project repository, then create the specified cube on the file system. Here is an example program:

```
proc olap cube=Star
    path=c:\cubes
    fact=olapsio.ordfact
metasvr host=localhost
    port=9999t
    protocol=bridge
    userid=userid
    pw=pw
    repository=Project:etlUser1
    olap_schema=OLAP Schema
;

dimension Time hierarchies=(YWD YMD YQMD) type=time
    dimtbl=olapsio.timedim dimkey=date_ID factkey=order_date
;

hierarchy YWD caption="Year-Week-Day"
    levels=(Year_ID Week_Name Date_ID );
hierarchy YMD caption="Year-Month-Day"
    levels=(Year_ID Month_Name Date_ID);
hierarchy YQMD caption="Year-Quarter-Month-Day"
    levels=(Year_ID Quarter Month_name Date_ID);

level year_ID      type=year;
level quarter      type=quarters;
level month_name   type=months;
level week_name    type=weeks;
level date_ID      type=days;
property WeekDay_Number_US caption="US WeekDay Number" column=weekday_no
```



```

level=date;
property WeekDay_Number_EU caption="EU WeekDay Number" column=weekday_eu
level=date;
property Week_Number_EU caption="EU Week Number" column=week_no
hierarchy=YWD level=week_name;
property Month_Number caption="Month Number" column=month_no
hierarchy=YMD level=month_name;
property Month_Number caption="Month Number" column=month_no
hierarchy=YQMD level=month_name;
property Holidays_US caption="US Holidays" column=Holiday_us
level=date;

dimension Customers hierarchies=(PersonalData CompanyUsage)
    dimtbl=olapsio.custdim dimkey=customer_id factkey=customer_id;

hierarchy PersonalData levels=(Customer_Name Customer_Age Customer_Gender);

hierarchy CompanyUsage
    empty_char=_missing_
    levels=(Customer_Group Customer_Type);

dimension Geography hierarchies=(Geography)
    dimtbl=olapsio.geogdim dimkey=street_id factkey=street_id;
hierarchy Geography
    empty_char=_missing_
    levels=(Continent_Name Country State Region Province County City)
;

dimension Organization hierarchies=(PersonalStats Organization)
    dimtbl=olapsio.orgdim dimkey=employee_id factkey=employee_id;
hierarchy PersonalStats levels=(Employee_name Job_Title Salary Gender);

hierarchy Organization
    empty_char=_missing_
    levels=(Company Department Org_Group Section Job_Title);

MEASURE DiscountSUM STAT=SUM COLUMN=Discount;

MEASURE CostPrice_Per_UnitSUM STAT=SUM COLUMN=CostPrice_Per_Unit
    FORMAT=DOLLAR10.2
;

MEASURE QuantitySUM STAT=SUM COLUMN=Quantity
    CAPTION='Sum of Quantity'
;

MEASURE Total_Retail_PriceSUM STAT=SUM COLUMN=Total_Retail_Price
    FORMAT=DOLLAR12.2
;

AGGREGATION Continent_Name Country State Region Customer_Group Customer_Type
    / NAME='RegionalCustomerUse'
;

AGGREGATION Year Quarter Customer_Group Customer_Type
    / NAME='QuarterlyCustomerUse'
;

AGGREGATION Year Customer_Group Customer_Type
    / NAME='YearlyCustomerUse'

```

```

;
AGGREGATION Continent_Name Country State Region Province Company Department
              Org_Group Section
              / NAME='WorldwideStaff'
;
AGGREGATION Continent_Name Country State Region Province Employee_Name
              Job_Title Salary
              / NAME='WorldwideSalaries'
;

run;

```

For details about the OLAP procedure, see the *SAS OLAP Server: Administrator's Guide*.

Submit the Code

Perform these steps to submit your SAS code to the Source Editor window in SAS ETL Studio.

- 1 From the SAS ETL Studio desktop, select

Tools ► Source Editor

from the menu bar. The Source Editor window is displayed.

- 2 Paste the SAS code for the cube into the Source Editor window.

- 3 To submit the code, select

Editor ► Submit

from the menu bar. The code is submitted to the default SAS application server.

- 4 After the code has completed execution, use Log tab on the Source Editor window to view any messages, statistics, warnings, or errors. If you find errors, edit and resubmit the code until the code runs successfully.

In the current example, when the code is successful, it will write the cube to the specified project repository. The next task is to run SAS ETL Studio and check in the cube to the change-managed repository (just as you would if you used the Cube Designer to create a new cube).

Check In the Cube

Under change management, new metadata objects are added to the Project tree on the SAS ETL Studio desktop. You must check in the new table metadata in order to save it to the change-managed repository.

- 1 Run SAS ETL Studio and open the metadata profile that specifies the project repository where the new cube was added.
- 2 In the Project tree, select the repository icon (such as *Project: etlUser1*).
- 3 From the menu bar on the SAS ETL Studio desktop, select

Project ► Check In Repository

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

Additional Information about Cubes

The online help for SAS ETL Studio provides additional information about cubes. Perform these steps to display the relevant help topics:

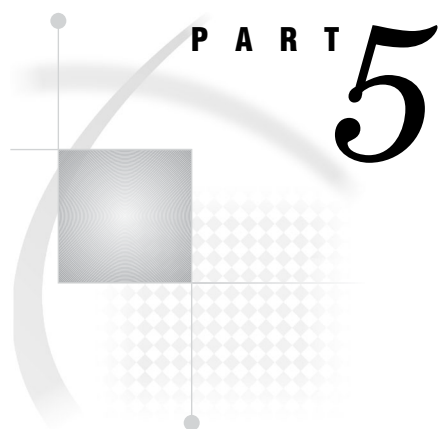
- 1 From the SAS ETL Studio menu bar, select

Help ► Contents

The online help window is displayed.

- 2 In the left pane of the help window, select

Working with Cubes

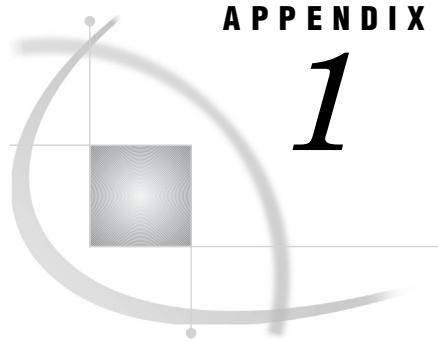


Appendices

Appendix 1 **Usage Notes** 177

Appendix 2 **Building Java Plug-ins for SAS ETL Studio** 183

Appendix 3 **Recommended Reading** 199



Usage Notes

<i>Migrating from SAS/Warehouse Administrator to SAS ETL Studio</i>	177
<i>ODBC Informix Library</i>	177
<i>SQL Join Transformation</i>	178
<i>Reordering Group by Rows or Columns</i>	178
<i>Using Compound Expressions</i>	178
<i>Signon Scripts for SAS/CONNECT Servers</i>	179
<i>Preassigned Libraries in SAS ETL Studio</i>	179
<i>Separate Login for Each Authentication Domain</i>	179
<i>Access to Tables Using ODBC DB2 z/OS Pass-Through</i>	179
<i>SAS Source Designer Does Not Import Keys</i>	180
<i>Truncating Tables Does Not Free Physical Storage</i>	180
<i>DBMS Tables Might Be Unusable after Dropping or Re-Creating</i>	180
<i>Saving Metadata Changes to the Corresponding Physical Table</i>	180
<i>New Schema Names Must Match the Names in the DBMS</i>	181
<i>Usage Notes for Metadata Import/Export</i>	181
<i>Case and Special Characters in SAS Names</i>	181

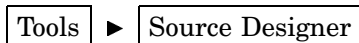
Migrating from SAS/Warehouse Administrator to SAS ETL Studio

For information about migrating from SAS/Warehouse Administrator to SAS ETL Studio, see *Migration: Converting from SAS/Warehouse Administrator to SAS ETL Studio* at support.sas.com/rnd/migration/planning/files/etlstatement.html

ODBC Informix Library

Follow these steps to preserve the case of your table names when using an ODBC Informix library:

- 1 From the SAS ETL Studio desktop, select



from the menu bar. The Source Designer selection window is displayed.

- 2 Select the ODBC Source Designer icon and click **Next**. The library selection window is displayed.
- 3 In the library selection window, select the appropriate ODBC library, then click **Edit**. A library properties window is displayed.
- 4 On the library properties window, click the **Options** tab.

- 5 On the **Options** tab, click **Advanced Options**. The Advanced Options window is displayed.
- 6 In the Advanced Options window, select the **Output** tab.
- 7 On the **Output** tab, select YES in the **Preserve column names as in the DBMS** field.
- 8 Enter the following expression in the **Options used in DBMS CREATE TABLE** field:
`QUOTE_CHAR=`
- 9 Select the **Input/Output** tab.
- 10 Select YES in the **Preserve DBMS table names** field.
- 11 Click **OK** to save your changes.

SQL Join Transformation

Reordering Group by Rows or Columns

In the properties window for the SQL Join transformation, on the **Group By** tab, you can select and reorder only one row or column at time in the **Column Name** table.

Using Compound Expressions

In the properties window for the SQL Join transformation, you can enter expressions in the flowing tabs:

- ☐ **Tables** tab
- ☐ **Mapping** tab
- ☐ **Where** tab
- ☐ **Having** tab

If you enter an expression in which AND or OR are combined with any of the following functions, you must enclose those functions in parentheses.

- ☐ DATE()
- ☐ DATETIME()
- ☐ TIME()
- ☐ TODAY()

Here are some examples:

```
Delivery_Date > (TODAY()) AND Order_Type = "AB"
(Delivery_Date > TODAY()) AND Order_Type = "AB"
```

Signon Scripts for SAS/CONNECT Servers

SAS ETL Studio uses a SAS/CONNECT server to submit generated SAS code to machines that are remote from the default SAS application server. A SAS/CONNECT server can also be used for interactive access to remote libraries.

For SAS ETL Studio to generate the appropriate code for scripted signon to a SAS/CONNECT server, you must specify a valid user ID and password in the signon script.

Preassigned Libraries in SAS ETL Studio

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*.

If you use the metadata LIBNAME engine to preassign a library, and you use SAS ETL Studio to access that library, SAS ETL Studio will display only those tables in the library that have been registered in a current metadata repository. It will not display all tables in the library simply because the library itself has been preassigned with the metadata LIBNAME engine.

Accordingly, when using SAS ETL Studio wizards to access tables in a library, you might need to use a library that was not preassigned using the metadata LIBNAME engine.

Separate Login for Each Authentication Domain

Administrators define the metadata for users and groups as part of the setup tasks for a data warehousing project. The login metadata for each user and group includes an authentication domain.

To access a relational database from SAS ETL Studio, your login and the database server on which the database resides must belong to the same authentication domain. Otherwise, you will not be able to read any existing tables in the relational database, and you will not be able to use SAS ETL Studio source designers or target designers to create tables in the relational database.

Accordingly, you must have a separate login for each authentication domain that contains a database server that you need to access. For more information about defining login metadata for users and groups, see the *SAS Management Console: User's Guide*.

Access to Tables Using ODBC DB2 z/OS Pass-Through

To use the pass-through facility for ODBC DB2/zOS to access tables, you must configure the password and user ID. Since DB2/zOS pass-through does not support the PASSWORD= and USER= options, you must configure these options on the ODBC DB2/zOS source using the ODBC Administrator.

SAS Source Designer Does Not Import Keys

In this release, the SAS Source Designer does not import the metadata for keys, such as the primary key or a foreign key.

Truncating Tables Does Not Free Physical Storage

The default load technique for a table is **Truncate Table**. There is a known issue with SAS tables that use the truncate option. If you run the job several times, SAS marks the existing data set observations as deleted and then appends the new observations but does not free the physical storage of those deleted observations.

To prevent this from happening, perform the following steps before running the job:

- 1 Open the SAS ETL Studio job that contains the SAS table for which physical storage must be freed.
- 2 Display the properties window for the Loader transformation that populates the SAS table.
- 3 In the properties window, select the **Load Technique** tab.
- 4 On the **Load Technique** tab, select the **Drop Table** option.
- 5 Click **OK** to save changes.
- 6 Rerun the job.

DBMS Tables Might Be Unusable after Dropping or Re-Creating

When SAS drops and recreates a table in a DBMS, it can destroy key metadata that is necessary for operation. For example, dropping and creating Siebel interface tables in Oracle result in tables that are unusable for running the Siebel process that uses those interface tables.

To prevent this from happening, perform the following steps before running the job:

- 1 Open the SAS ETL Studio job that contains the DBMS table whose metadata must be updated.
- 2 Display the properties window for the Loader transformation that populates the DBMS table.
- 3 In the properties window, select the **Load Technique** tab.
- 4 On the **Load Technique** tab, select the **Truncate Table** option.
- 5 Click **OK** to save changes.
- 6 Re-run the job.

Saving Metadata Changes to the Corresponding Physical Table

For jobs that have been run once and contain a Loader transformation, metadata changes to columns are saved in the physical target only when you select **Drop Target** in the **Load Technique** tab of the Loader transformation. **Drop Target** is not selected by default.

New Schema Names Must Match the Names in the DBMS

When you are adding or editing a schema in a New Schema wizard or in the Physical Storage tab of the Properties window of a DBMS table, the name of the schema in the metadata must exactly match (including case) the name of the corresponding schema in the DBMS.

Usage Notes for Metadata Import/Export

The following notes apply to the metadata import and export wizards that are displayed from the Shortcut Bar or from the Tools item on the menu bar of the SAS ETL Studio desktop:

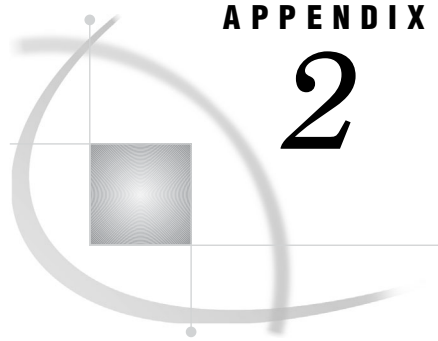
- 1 The metadata export wizard does not export notes or user-defined properties, such as SAS formats and informats.
- 2 The metadata import and export wizards do not recognize numeric lengths less than 8.

Case and Special Characters in SAS Names

By default, the names for SAS tables and columns must follow the standard rules for SAS names. However, SAS ETL 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. For details, see “Case and Special Characters in SAS Table and Column Names” on page 51.

Note the following exceptions:

- Special characters are not supported in SAS table names.
- Leading blanks are not supported for SAS column names. Leading blanks in a SAS column name are stripped out.
- Neither the External File source designer nor SAS/SHARE libraries and tables support case-sensitive names for SAS tables or special characters in column names. When using these components, the names for SAS tables and columns must follow the standard rules for SAS names.



APPENDIX

2

Building Java Plug-ins for SAS ETL Studio

Overview	183
Shortcut Plug-ins	183
PluginInterface	184
ShortcutInterface	184
Installing a Shortcut Plug-in	184
Example: Building a Source Designer Plug-in	185
Mapping the Metadata and Building the Plug-in	185
Installing and Running the Plug-in	197
Plug-in Output	197

Overview

SAS ETL Studio plug-ins are Java files that provide specific functions by creating specific types of metadata. For example, as described in “Java Transformations and SAS Code Transformations” on page 111, a number of the transformation templates in the Process Library are Java plug-ins, such as the SAS Sort template and the Create Match Code template.

Several plug-ins are installed by default, and other plug-ins are available from SAS to provide specialized functions. You can also develop your own plug-in transformation templates, source designer wizards, target designer wizards, and new object wizards.

To develop Java plug-ins, you should be familiar with the following:

- Java plug-in software. For details about this software, see the following location:
java.sun.com/products/plugin
- SAS Management Console plug-ins, which are similar to the plug-ins for SAS ETL Studio. See the *Guide to Building SAS Management Console Plug-Ins*, which is provided on the SAS Management Console installation CD.
- SAS Metadata Model. For details, see *SAS Open Metadata Architecture: Reference*, which is available on the SAS Online Doc CD and in SAS Help and Documentation.
- The *SAS ETL Studio Plug-In Framework*. For details about this framework, see the SAS BI Package Libraries at **support.sas.com/rnd/gendoc/bi/api/**

Shortcut Plug-ins

Use the `ShortcutInterface` and `PluginInterface` to add a Java plug-in to the Tools menu, to the Shortcut Bar on the SAS ETL Studio desktop, or to both. The methods for each of these interfaces are described as follows:

PluginInterface

```
public void initPlugin()
    performs any necessary initialization for the plug-in

public void dispose()
    performs any necessary cleanup for disposing of the plug-in

public String getDescription()
    returns a string that contains a description of the plug-in

public Icon getIcon()
    returns a 16x16 icon for the plug-in that will be displayed on the Tools menu

public String getName()
    returns a string that represents the name of the plug-in.
```

ShortcutInterface

```
public void onSelected()
    contains the actions to take when the user opens the plug-in

public Icon getLargeIcon()
    returns a 32x32 icon for the plug-in that will be displayed on the shortcut bar

public JMenuItem getMenuItems()
    returns a menu item (using JMenuItem) that will be added to the Tools menu. If
    you want, you can define a mnemonic, an accelerator key, or both by using this
    method with JMenuItem.setMnemonic() and JMenuItem.setAccelerator(),
    respectively.

public int getLocations()
    returns one of three values:

        SHOW_ON_SHORTCUT
            displays the plug-in only on the shortcut bar

        SHOW_ON_MENU
            displays the plug-in only on the Tools menu

        SHOW_ON_ALL
            displays the plug-in on both the shortcut bar and the Tools menu.
```

Installing a Shortcut Plug-in

After you have created a plug-in, create a JAR file containing the implementation of `ShortcutInterface` for your plug-in, along with any needed images, classes, or other files. In the manifest for the JAR, you must include a line that defines the `Plugin-Init` attribute. The sample JAR that is shipped with SAS ETL Studio (in the `com.sas.wadmin.visuals` package) contains a plug-in called `SampleShortcutPlugin`. This sample contains the following attribute line in its manifest:

```
Plugin-Init: com.sas.wadmin.visuals.SampleShortcutPlugin.class
```

Save this JAR in the `plugins` directory in the SAS ETL Studio home directory. For example, if you installed SAS ETL Studio in `C:\Program Files\SAS\SAS ETL Studio\9.1`, save the JAR for your new plug-in in `C:\Program Files\SAS\SAS ETL`

Studio\9.1\plugins. After you do this, the next time you start SAS ETL Studio this plug-in is automatically loaded and displayed.

After you have created a JAR for SampleShortcutPlugin in the plug-ins directory and restarted SAS ETL Studio, your new plug-in will be available from the the Shortcut bar and Tools menu.

The shortcut plug-ins are added at the bottom of the Shortcut bar, just above the Options item in the Tools menu, or both.

Example: Building a Source Designer Plug-in

This section shows you how to develop a source designer plug-in for SAS ETL Studio. The SourceDesignerPlugin sample provides a method for building a new source designer, along with links to web site resources for more detailed information.

Before you can design a new source designer plug-in, determine the the format of the source and the metadata that you want to capture about the source. You need this information in order to design property windows that will gather information about your source.

Mapping the Metadata and Building the Plug-in

Decide what type of metadata that you want to register as a result of running your plug-in. For more details about defining metadata, see the SAS Metadata Model in the *SAS Open Metadata Architecture: Reference*, which is available in SAS Help and Documentation.

Source designer plug-ins are integrated into SAS ETL Studio using SourceDesignerInterface, which is a Java interface. For technical details about the *SAS ETL Studio Plug-In Framework* and the SourceDesignerInterface (com.sas.wadmin.plugins.SourceDesignerInterface), see the SAS BI Package Libraries at support.sas.com/rnd/gendoc/bi/api/

You can also refer to the following sample Java programs:

□ Manifest.mf:

```
Manifest-Version: 1.0
Main-Class: plugindir
Created-By: 1.3.0 (Sun Microsystems Inc.)
Plugin-Init: plugindir.SourceDesignerPlugin.class
```

□ PropertyBundle.properties

```
ImageLocation.notrans = com/sas/wadmin/visuals/res/
StepOut.image=DataSetSource16.gif
Common.warehouse_w2.image=warehouse_w2.gif
Common.warehouse_w3.image=warehouse_w3.gif
Common.warehouse_w4.image=warehouse_w4.gif
Common.warehouse_w5.image=warehouse_w5.gif
Common.warehouse_w6.image=warehouse_w6.gif
wa_source_connectInfo.image=wa_source_connectInfo.gif
gen_select_table.image=gen_select_table.gif
gen_target_location.image=gen_target_location.gif
gen_summary.image=gen_summary.gif
gen_subset_tables.image=gen_subset_tables.gif
FinishTab.Title.txt=Finish
```

- SourceDesignerPlugin.class
- SourceDesignerPlugin.java

```

/**
 * Required by PluginInterface, returns the icon to be used with the
 * Interface
 * In the Source Designer, this is the icon that shows up in the tree.
 *
 * @return Icon to be displayed with the Plugin
 */
public Icon getIcon(){return m_icon;};

/**
 * Return a tooltip string that to be displayed
 *
 * @return String containing the tooltip to be displayed.
 */
public String getToolTip() {return m_tooltip;};

/**
 * The designer can choose to place the plugin in any category that the
 * designer chooses.
 * You can concatenate categories with a "."; each level is a level Source
 * Designer selection tree. For example, a category name of Levels.MyStuff
 * would show up as:
 *
 * - Source Designers
 *   - Levels
 *     - MyStuff
 *       + mynewtransform
 *
 * Designers should take care not when describing these hierarchies
 * such that they fit in well with other similar designers.
 *
 * @return the Category to place this addin into
 */
public String getCategory() {return m_category;};

/**
 * Required by the PluginInterface: returns the description of the Plugin.
 *
 * @return String containing the description of the plugin
 */
public String getDescription() {return "";}

public void dispose()
{
}

} //end public class

```

- Panel1.class
- Panel1.java

```

/**
 * Title:          Panel1

```



```

* Description:  Panell
* Copyright:    Copyright 2003, SAS Institute Inc * Company:  SAS Institute
* Inc. * @version 1.0
*/
package plugindir.visuals;

import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.Insets;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.*;
import javax.swing.JRadioButton;
import javax.swing.border.EtchedBorder;

import com.sas.metadata.MdException;
import com.sas.plugins.PluginResourceBundle;
import com.sas.workspace.WAPanel;

/**
 * Panell
 */
public class Panell extends WAPanel
{
    /** Property bundle */
    private static PluginResourceBundle bundle =
        new PluginResourceBundle( Panell.class );

    protected JLabel m_label = new JLabel("Put Label here..");

    /** Boolean to turn on a border around this panel */
    protected boolean      m_fBorder=false;

    /**
     * Constructs a panell
     *
     * @param fBorder - create a border around this panel (true) or not
     * (false)
     */
    public Panell(boolean fBorder)
    {
        super();
        m_fBorder = fBorder;
        initialize();
        layoutWidgets();
    }

    /**
     * Initialization routine.  Creates and initializes all of the widgets
     * on the panel.
     */
    public void initialize()

```

```

{
    super.initialize();
} //end public void initialize()

/**
 * Validate the data on the panel.
 */
public boolean validateData()
{
    return true;
} //end public boolean validateData()

/**
 * Just like in a property tab, this method is called before the panel
 * is made visible to do the model/view data exchange.
 * @param saveToModel true - move widget values to model values;
 *                   false - move model values to widgets values
 */
public boolean doDataExchange(boolean bSaveToModel) throws MdException
{
    if (bSaveToModel == false)
    {
    }
    return true;
} //end public boolean doDataExchange(boolean bsaveToModel) throws MdException

/**
 * Arrange the widgets in displayed panel.
 */
public void layoutWidgets()
{
    //Let's layout the button panel
    GridBagLayout gridBagLayout1 = new GridBagLayout();
    GridBagConstraints gbc1 = new GridBagConstraints();
    WAPanel myPanel = new WAPanel();
    myPanel.setLayout( gridBagLayout1 );
    myPanel.setBorder(new EtchedBorder(EtchedBorder.LOWERED));

    // Add the radio buttons to the panel
    gbc1.gridx = 0;
    gbc1.gridy = 0;
    gbc1.gridwidth = GridBagConstraints.RELATIVE;
    gbc1.gridheight = 1;
    gbc1.weightx = 1.0;
    gbc1.weighty = 1.0;
    gbc1.anchor = GridBagConstraints.WEST;
    gbc1.fill = GridBagConstraints.HORIZONTAL;
    gbc1.insets = new Insets(0,5,0,0);
    gridBagLayout1.setConstraints( m_label, gbc1 );
    myPanel.add( m_label );

    //The Main panel's gridbaglayout stuff
    GridBagLayout gridBagLayout = new GridBagLayout();

```

```

        GridBagConstraints gbc = new GridBagConstraints();
        setLayout( gridBagLayout );

        // Add the radio buttons to the panel
        gbc.gridx = 0;
        gbc.gridy = 0;
        gbc.gridwidth = GridBagConstraints.RELATIVE;
        gbc.gridheight = 1;
        gbc.weightx = 1.0;
        gbc.weighty = 1.0;
        gbc.anchor = GridBagConstraints.NORTHWEST;
        gbc.fill = GridBagConstraints.HORIZONTAL;
        gbc.insets = new Insets(0,0,0,0);
        gridBagLayout.setConstraints( myPanel, gbc );
        add( myPanel );

    } //end public void layoutWidgets()

}

```

□ Panel2.class

□ Panel2.java

```

/**
 * Title:          Panel2
 * Description:    Panel2
 * Copyright:      Copyright 2003, SAS Institute Inc * Company:  SAS Institute
 * Inc. * @version 1.0
 */
package plugindir.visuals;

import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.Insets;

import javax.swing.JLabel;
import javax.swing.border.EtchedBorder;

import com.sas.metadata.MdException;
import com.sas.plugins.PluginResourceBundle;
import com.sas.workspace.WAPanel;

/**
 * Panel 2
 *
 */
public class Panel2 extends WAPanel {
    /** Property bundle */
    private static PluginResourceBundle bundle =
        new PluginResourceBundle(Panel2.class);

    /** Boolean to turn on a border around this panel */
    protected boolean m_fBorder = false;

    protected JLabel m_label;

```

```

/**
 *
 * @param fBorder - create a border around this panel (true) or not (false)
 */
public Panel2(boolean fBorder) {
    super();
    m_fBorder = fBorder;
    initialize();
    layoutWidgets();
}

/**
 * Initialization routine. Creates and initializes all of the widgets
 * on the panel.
 */
public void initialize() {
    m_label = new JLabel("Put Label here ");
    super.initialize();
} //end public void initialize()

/**
 * Validate the data on the panel.
 */
public boolean validateData() {
    return true;
} //end public boolean validateData()

/**
 * Just like in a property tab, this method is called before the panel is made
 * visible to do the model/view data exchange.
 * @param saveToModel true - move widget values to model values;
 * false - move model values to widgets values
 */
public boolean doDataExchange(boolean bSaveToModel) throws MdException {
    if (bSaveToModel == false) {
    } else
    {
    }
    return true;
} //end public boolean doDataExchange(boolean bsaveToModel) throws MdException

/**
 * Arrange the widgets in displayed panel.
 *
 */
public void layoutWidgets() {
    //Let's layout the button panel
    GridBagLayout gridBagLayout1 = new GridBagLayout();
    GridBagConstraints gbcl = new GridBagConstraints();
    WAPanel myPanel = new WAPanel();
    myPanel.setLayout(gridBagLayout1);
    myPanel.setBorder(new EtchedBorder(EtchedBorder.LOWERED));

    // Add the radio buttons to the panel

```

```

        gbc1.gridx = 0;
        gbc1.gridy = 0;
        gbc1.gridwidth = GridBagConstraints.RELATIVE;
        gbc1.gridheight = 1;
        gbc1.weightx = 1.0;
        gbc1.weighty = 1.0;
        gbc1.anchor = GridBagConstraints.WEST;
        gbc1.fill = GridBagConstraints.HORIZONTAL;
        gbc1.insets = new Insets(0, 5, 0, 0);
        gridBagLayout1.setConstraints(m_label, gbc1);
        myPanel.add(m_label);

        //The Main panel's gridbaglayout stuff
        GridBagLayout gridBagLayout = new GridBagLayout();
        GridBagConstraints gbc = new GridBagConstraints();
        setLayout(gridBagLayout);

        // Add the radio buttons to the panel
        gbc.gridx = 0;
        gbc.gridy = 0;
        gbc.gridwidth = GridBagConstraints.RELATIVE;
        gbc.gridheight = 1;
        gbc.weightx = 1.0;
        gbc.weighty = 1.0;
        gbc.anchor = GridBagConstraints.NORTHWEST;
        gbc.fill = GridBagConstraints.HORIZONTAL;
        gbc.insets = new Insets(0, 0, 0, 0);
        gridBagLayout.setConstraints(myPanel, gbc);
        add(myPanel);

    } //end public void layoutWidgets()

}

```

□ Tab1.class

□ Tab1.java

```

/**
 * Title:          Tab1
 * Description:    Tab1
 * Copyright:      Copyright 2003, SAS Institute Inc * Company:  SAS
 * Institute Inc. * @version 1.0
 */
package plugindir.visuals;

import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.Insets;

import com.sas.plugins.PluginResourceBundle;
import com.sas.workspace.WsDescriptionWizardTab;
/**
 * Tab1
 *
 *

```

```

*/
public class Tab1 extends WsDescriptionWizardTab
{
    /** Property bundle */
    private static PluginResourceBundle bundle =
        new PluginResourceBundle(Tab1.class);

    private PluginResourceBundle m_eda_bundle;

    protected Panell myPanell;
    /**
     * Main constructor
     */
    public Tab1()
    {
        super();
        setHelpTopic("selecttablesbyapplicationareawindow");
        myPanell = new Panell(false);
        initialize();
    }

    /**
     * Initialize the widgets and their layout.
     */
    public void initialize()
    {
        this.setLayout(new GridBagLayout());

        this.add(myPanell, new GridBagConstraints(0, 0, 1, 1, 1.0, 1.0
            ,GridBagConstraints.NORTHWEST, GridBagConstraints.BOTH,
            new Insets(0, 0, 0, 0), 0, 0));

    } //end public void initialize()

    /**
     * Transfer data to and from the model.
     *
     * @param bSaveToModel True if transferring from view to model, false if
     * vice versa
     */
    public boolean doDataExchange( boolean bSaveToModel ) throws
        com.sas.metadata.MdException
    {
        return myPanell.doDataExchange(bSaveToModel);

    } //end public boolean doDataExchange( boolean bSaveToModel ) throws
        com.sas.metadata.MdException

    /**
     * Validate data entered into panel.
     *
     * @return boolean to determine if there is validate data in the panel or not
     */
    public boolean validateData()

```

```

{
    return myPanell.validateData();
} //end public boolean validateData()

/**
 * Run when the Next button is selected.
 */
public void onNext()
{
    super.onNext();
} //end public void onNext()

/**
 * Run when the back button is selected.
 */
public void onBack()
{
    super.onBack();
} //end public void onBack()

/**
 * Create the finish string that shows up in WAWizardFinish
 */
public String createFinishString()
{
    String finishString = "This is the finish string";

    return finishString;
} //end public String createFinishString()

} //

```

□ Tab2.class

□ Tab2.java

```

/**
 * Title:          Tab2
 * Description:    Tab2
 * Copyright:      Copyright 2003, SAS Institute Inc * Company:  SAS
 * Institute Inc. * @author
 * @version       1.0
 */
package plugindir.visuals;

import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.Insets;

import com.sas.metadata.CMetadata;
import com.sas.metadata.MdObjectFactory;
import com.sas.metadata.MdObjectStore;
import com.sas.metadata.PhysicalTable;
import com.sas.metadata.SASLibrary;
import com.sas.plugins.PluginResourceBundle;

```

```

import com.sas.workspace.WAPropertyTab;
import com.sas.workspace.WAWizardDialog;
import com.sas.workspace.Workspace;
import com.sas.workspace.WsDescriptionWizardTab;
/**
 * Tab2
 *
 */
public class Tab2 extends WsDescriptionWizardTab
{
    /** Property bundle */
    private static PluginResourceBundle bundle =
        new PluginResourceBundle(Tab2.class);

    private PluginResourceBundle m_edata_bundle;

    protected Panel2 myPanel2;
    /**
     * Main constructor
     */
    public Tab2()
    {
        super();
        setHelpTopic("Tabber2");
        myPanel2 = new Panel2(false);
        initialize();
    }

    /**
     * Initialize the widgets and their layout.
     */
    public void initialize()
    {
        this.setLayout(new GridBagLayout());

        this.add(myPanel2, new GridBagConstraints(0, 0, 1, 1, 1.0, 1.0
            ,GridBagConstraints.NORTHWEST, GridBagConstraints.BOTH,
            new Insets(0, 0, 0, 0), 0, 0));

    } //end public void initialize()

    /**
     * Transfer data to and from the model.
     *
     * @param bSaveToModel True if transferring from view to model, false
     * if vice versa
     */
    public boolean doDataExchange( boolean bSaveToModel ) throws
    com.sas.metadata.MdException
    {
        if (bSaveToModel == false)
            myPanel2.doDataExchange(bSaveToModel);
        else
            // this is performed after the user has selected FINISH on the

```



```

        wizard screen
    write_metadata();
    return true;

} //end public boolean doDataExchange( boolean bSaveToModel ) throws
    com.sas.metadata.MdException

/**
 * Validate data entered into panel.
 *
 * @return boolean to determine if there is validate data in the panel
 * or not
 */
public boolean validateData()
{
    return myPanel2.validateData();
} //end public boolean validateData()

/**
 * Run when the Next button is selected.
 */
public void onNext()
{
    super.onNext();
} //end public void onNext()

/**
 * Run when the back button is selected.
 */
public void onBack()
{
    super.onBack();
} //end public void onBack()

/**
 * Create the finish string that shows up in WAWizardFinish
 */
public String createFinishString()
{
    String finishString = "This is the finish string for tab2";

    return finishString;
} //end public String createFinishString()

public void write_metadata() throws com.sas.metadata.MdException {
    WAWizardDialog myWizard = (WAWizardDialog) this.getTopLevelAncestor();
    //Get the Right repository to add it to, we hope...
    Workspace workspace = Workspace.getWorkspace();
    CMetadata myRepository = workspace.getDefaultRepository();
    String strID = myRepository.getFQID().substring(9, 17);

    MdObjectStore store =
        (MdObjectStore) myWizard.getWizardData("OBJECTSTORE");

```

```

SASLibrary dbLibrary = (SASLibrary) myWizard.getWizardData("Library");

PhysicalTable newTable =
    (PhysicalTable) MdObjectFactory.createComplexMetadataObject(
        store,
        store,
        "TableName",
        "PhysicalTable",
        strID);
myWizard.setMasterObject(newTable);

//Let's set the attributes for this table
newTable.setIsCompressed(0);
newTable.setIsEncrypted(0);
newTable.setDBMSType("");
newTable.setSASTableName("TableName");
newTable.setTableName("TableName");
newTable.setName("TableName");
newTable.setDesc("Table Description");
newTable.setNumRows(-1);
// we are assuming everything is DATA not View at this point in the game...
newTable.setMemberType("DATA");

for (int i = 0; i < 10; i++) {
    com.sas.metadata.Column newColumn =
        (com.sas.metadata.Column) MdObjectFactory.createComplexMetadataObject(
            store,
            store,
            "Column" + i,
            "Column",
            strID);
    newTable.addElementToChangeList(newColumn);
    newColumn.setSASColumnName("Column" + i);
    newColumn.setSASColumnType("C");
    newColumn.setSASColumnLength(10);
    String format = "$10.";
    newColumn.setSASFormat(format);
    newColumn.setSASInformat("$10.");
    newColumn.setColumnName("ColumnName" + i);
    newColumn.setIsNullable(1);
    newTable.getColumns().addElement(newColumn);
}
newTable.updateMetadataAll();
} //end public void tableDefinition()

} //

```

In the sample source designer programs, the `SourceDesignerPlugin.java` class provides the needed plug-in navigation, where tabs display each panel of the plug-in. For example, public class `Tab1` and public class `Tab2` extend `WsDescriptionWizardTab`. Each tab then references a panel that displays that panel. For example, public class `Panel1` and public class `Panel2` extend `WAPanel`.

Installing and Running the Plug-in

In SAS ETL Studio, you must provide a JAR file that contains all class files, property bundles, and a manifest. Place this JAR file in the plugins subdirectory where SAS ETL Studio is located—for example, **ETLStudioDirectoryLocation\9.1\plugins**.

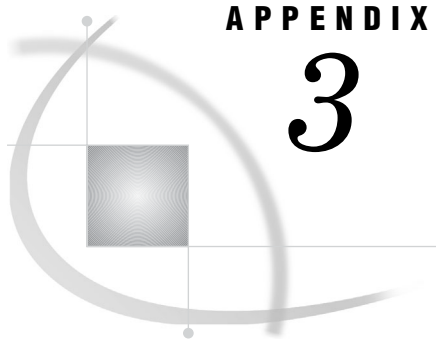
In the JAR file for each plug-in must be a manifest that includes the following information:

```
Manifest-Version: 1.0
Main-Class:      plugindir /* Directory where your plug-in class that */
                  /* extends SourceDesignerInterface resides */
Created-By:      1.3.0 (Sun Microsystems Inc.)
Plugin-Init:     plugindir.SourceDesignerPlugin.class
```

You might also need to set the Class-Path for the plug-in to run properly.

Plug-in Output

The `write_metadata` method in Tab2 shows an example of writing the physical table, `TableName`, with ten columns, named `Column0` through `Column9`. This table shows up in the Custom tree of SAS ETL Studio.



APPENDIX

3

Recommended Reading

Recommended Reading 199

Recommended Reading

Here is the recommended reading list for this title:

- *Communications Access Methods for SAS/CONNECT and SAS/SHARE*
- *Moving and Accessing SAS Files*
- *SAS Intelligence Architecture: Planning and 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, an output data set that is created by applying a scheme to a variable or column that contains character values. The analysis data set identifies clusters of similar values, as well as the value that occurs most frequently in each cluster.

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.

custom repository

in the SAS Open Metadata Architecture, a metadata repository that has dependencies. A custom repository is often used to specify resources that are unique to a particular data collection. A custom repository often depends on (inherits metadata from) a foundation repository. For example, a custom repository could define sources and targets that are unique to a particular data warehouse in a test environment. The custom repository could inherit most server metadata from a foundation repository in the test environment.

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 transformation

in SAS data quality, a data cleansing process that applies a scheme to specified character values. The scheme creates match codes internally in order to create clusters. All values in each cluster are then transformed to the single value that occurs most frequently in 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.

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 does not depend on other repositories. A foundation repository is often used to specify metadata for global resources. For example, a foundation repository could define most of the servers that are used in several data warehouses in a test environment.

global resource

an object, such as a server or a library, that is shared on a network.

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

a version of a character value from which some of the vowels have been removed, insignificant words (if any) have been removed, and the capitalization and formatting of words have been standardized. Match codes are used to identify clusters of similar values in a character variable or column. They can be used to reduce the number of duplicate entries in a data set.

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.

owner

the person who is responsible for the contents of an object such as a table or a library. See also administrator.

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

in the SAS Open Metadata Architecture, a metadata repository that enables a specified person to add or update metadata in a change-managed 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.

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 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 ETL Studio generates code for a job, it uses SAS/CONNECT software to submit code to remote computers. SAS ETL 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 determines the granularity of the clusters that are generated during data analysis and data cleansing.

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.

source

a table, view, or file from which you will extract information. Sources can be in any format that SAS can access, on any supported hardware platform. The metadata for a source is typically an input to a job.

source designer

in applications that support the SAS Open Metadata Architecture, a type of wizard that enables you to enter the metadata for a source.

target

a table, view, or file that contains information that has been extracted from a source. Targets can be in any format that SAS can access, on any supported hardware platform. A target is an output of a job.

target designer

in applications that support the SAS Open Metadata Architecture, a type of wizard that enables you to enter the metadata for a target.

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.

Index

A

administrator setup tasks 43
 aggregations 168
 Analysis folder 109
 authentication domain 179

C

case sensitivity 51, 181
 change-managed metadata repositories 45
 change-management control 21
 cubes 160
 jobs 117
 sources 64
 targets 86
 change management facility 11
 client tier 8
 code
 See also SAS code transformation templates
 See also SAS code transformations
 generated 47, 100
 user-written 101, 118, 169
 column metadata 92
 for targets 93
 selecting 92
 updating 67, 89, 121
 column names 51, 53
 column variables 79
 column width 78
 columns
 configuring 144
 Cube Designer 161, 163
 cube metadata
 updating 161
 cubes 159
 building from star schema 162
 change-management control for 160
 checking in 169
 creating with Cube Designer 161
 examples 162, 169
 prerequisites for 160
 submitting user-written code for 169
 updating 161
 viewing data in 161
 custom formats 49

D

data analysis 12
 data cleansing 12, 28
 data marts 6
 creating 29
 data models
 importing for set of sources 65
 importing for set of targets 86
 data quality transformation templates 12, 53
 data tier 9
 Data Transforms folder 110
 data validation 28
 data warehouses 6, 28
 creating data marts 29
 creating dimensional data 29
 data cleansing 28
 denormalizing data 28
 designing 27
 example 31
 extracting data 28
 loading data 28
 planning 29
 security plan for 30
 validating data 28
 DBMS column names 51
 DBMS name options 52
 DBMS names
 schema names and 181
 DBMS table names 51
 DBMS tables
 accessing 49
 dropping 180
 metadata for tables with keys 65
 re-creating 180
 default SAS application server 46
 code generation and 47
 impact of 47
 interactive data access 47
 SAS/CONNECT server and 47
 selecting 58
 denormalizing data 28
 desktop 15
 dimensional data 29
 dimensions 165
 drill-through reporting 165

E

enterprise applications 50
 error log 56
 ETL Q link 6
 example data warehouse 31
 libraries for 51
 External File source designer 49
 extracting data from external files 66
 External File wizard
 extracting data from flat files 74
 external files
 accessing 49
 extracting data from 66, 74
 extracting data 6, 28
 from external files 66
 from flat files 74

F

flat files
 extracting data from 74
 flows 102
 formats
 accessing custom formats 49

G

generated code 47, 100
 groups
 entering metadata for 45

H

help 4, 14
 hierarchies 165

I

installing SAS ETL Studio 44
 interactive data access 47

J

Java options 55

Java plug-ins

- building 183
- building source designer plug-ins 185
- example 185
- location 56
- shortcut plug-ins 183

Java transformation templates 111

job metadata

- updating 118
- viewing 118

Job Properties window 111

jobs 100

- change-management control for 117
- checking in 117
- creating 114
- creating SAS code transformation templates 122
- creating with source designers 117
- creating with target designers 117
- deploying for scheduling 122
- examples 135, 150
- executing 102
- generated source code with 100
- joining tables 135
- New Job wizard 103
- populating 115
- report generation 135
- retrieving user-written code 118
- running 114, 116, 122
- SAS code transformation templates in 130
- scheduling 13, 102, 122
- source metadata in 120
- target metadata in 120
- transformation metadata in 120
- troubleshooting 116
- updating 115
- user-written source code with 101
- verifying output 116
- viewing 115
- viewing source data 119
- viewing target data 119
- windows for 102

joining tables 135

K

keys

- importing 180
- metadata for DBMS tables with 65
- Source Designer and 180

L

levels 165

libraries

- entering metadata for 48, 50
- for example data warehouse 51
- preassigned 179

loading data 6, 28

local resources 47

Log tab 107

login 179

M

mapping metadata 67, 89, 121

measure details 167

measures 167

member properties 168

menu bar 15

message logging 56

message window 16

metadata 10

- See also* column metadata
- checking out 114
- cube metadata 161
- DBMS tables with keys 65
- entering, for libraries 48, 50
- entering, for servers 46
- entering, for users and groups 45
- exporting 11, 54
- importing 11, 54
- job metadata 118
- mapping metadata 67, 89, 121
- saving changes 180
- source metadata 66, 67, 120
- source tables 68
- table metadata 65, 67, 87
- target metadata 88, 95, 120
- target tables 88, 89
- transformation metadata 120

metadata export wizard 181

metadata import wizard 181

metadata profiles 14

- creating 56
- opening 58

metadata repositories 14

- change-managed 45
- default 57

metadata server 14

mid-tier 9

migration

- SAS/Warehouse Administrator to SAS ETL Studio 177

multi-tier support 12

N

name options

- DBMS 52
- defaults for tables and columns 53
- for individual tables 68

New Job wizard 103

New Library wizard

- accessing DBMS tables 49
- entering metadata for libraries 50

normalized data 28

O

ODBC DB2/zOS pass-through

- accessing tables 179

ODBC Informix library 177

ODBC library

- accessing external files 49
- extracting data from external files 66

online help 4, 14

Open a Metadata Profile window 14

Options window 18

Orion Star Sports & Outdoors 31

Output folder 110

Output tab 107

output verification 116

P

preassigned libraries 179

Process Designer window 16, 105

Process Editor tab 107

process flow diagrams 16

Process Library tree 108

Publish folder 110

Publish to Archive transformation 146

R

remote resources 47

reports

- creating with jobs 135
- drill-through 165

S

SAS application server

- See* default SAS application server

SAS code transformation templates 111

- creating 122
- example 150
- identifying 130
- in jobs 130
- updating 154

SAS code transformations 131

- access control 131
- deleting folders for 132
- exporting 130
- importing 130

SAS/CONNECT servers

- default SAS application server and 47
- required for SAS ETL Studio 44
- signon scripts for 179

SAS ETL Studio 10

- features 10
- installing 44
- migrating from SAS/Warehouse Administrator to 177
- online help 4, 14
- starting 55

SAS Intelligence Architecture 7

SAS Intelligence Value Chain 6

SAS names 181

SAS/Warehouse Administrator

- migrating to SAS ETL Studio 177

schema names

- DBMS names and 181

security plan for data warehouses 30

server tier 9

- servers 44
 - See also* default SAS application server
 - entering metadata for 46
 - metadata server 14
 - SAS/CONNECT servers 44, 47, 179
- setup tasks
 - administrators 43
 - users 55
- shortcut bar 16
- shortcut plug-ins 183
 - installing 184
- signon scripts
 - SAS/CONNECT servers 179
- source code
 - generating 100
 - user-written 101
- Source Designer 180
- source designer plug-ins 185
- source designers
 - creating jobs with 117
 - entering metadata for source tables 68
 - selecting 69
 - specifying metadata for tables 65
- Source Editor
 - submitting user-written code for cubes 169
- Source Editor tab 107
- Source Editor window 17
- source metadata
 - updating 67
 - updating in jobs 120
 - viewing 66
 - viewing in jobs 120
- source tables
 - metadata for 68
- sources 64
 - change-management control for 64
 - examples 68, 74
 - importing data model for set of 65
 - prerequisite tasks 64
 - viewing data in 66
 - viewing job data 119
- special characters 51, 181
- SQL Join transformation 141, 178
- star schema
 - building cubes from 162
- starting SAS ETL Studio 55

- status line 16
- submitting code 169

T

- table metadata
 - specifying 65, 87, 89
 - specifying for source tables 68
 - updating 67, 88
- table names 51, 53
- Table Properties window 112
- tables
 - accessing 48
 - accessing with ODBC DB2/zOS pass-through 179
 - joining 135
 - name options for individual tables 68
 - target tables 88, 89
 - truncating 180
- target designers
 - creating jobs with 117
- target metadata
 - checking in 95
 - saving 95
 - updating in jobs 120
 - viewing 88
 - viewing in jobs 120
- Target Table Designer
 - specifying metadata for tables 87
- target tables
 - metadata for 89
 - updating metadata 88
- targets 85
 - change-management control for 86
 - column metadata for 93
 - example 89
 - importing data model for set of 86
 - physical storage information 94
 - prerequisite tasks 86
 - specifying table metadata 89
 - updating table metadata 88
 - viewing data in 87
 - viewing job data 119
- task flow 21

- toolbar 15
- Transformation Generator wizard 113, 124
- transformation metadata
 - updating in jobs 120
 - viewing in jobs 120
- Transformation Property windows 112
- transformations 6
 - See also* SAS code transformation templates
 - See also* SAS code transformations
 - data quality transformation templates 12, 53
 - Data Transforms folder 110
 - Java transformation templates 111
 - Publish to Archive transformation 146
 - SQL Join transformation 141, 178
- tree view 16
- trees 16
- truncating tables 180

U

- usage notes 23, 177
- user setup tasks 55
- user-written code
 - in jobs 101
 - retrieving in jobs 118
 - submitting for cubes 169
- user-written components 12
- users
 - entering metadata for 45

V

- validating data 28

W

- warehouse design 27
- warehouse project plans 44
- windows 13
 - for jobs 102
 - online help for 14
- wizards 19

Your Turn

If you have comments or suggestions about the *SAS 9.1 ETL Studio: 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
email: 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
email: suggest@sas.com

