## Contents

**What’s New in Data Administration for the SAS 9.4 Intelligence Platform**  v

### Chapter 1 / Overview of Common Data Sources  1

- Overview  1
- Hadoop Data  2
- SAS Data Sets  2
- Shared Access to SAS Data Sets  3
- Local and Remote Access to Data  4
- External Files  5
- XML Data  6
- Message Queues  7
- Relational Database Sources  8
- SAS Scalable Performance Data Server and SAS Scalable Performance Data Engine  11
- SAS Data Surveyors  14
- Change Data Capture  14
- DataFlux Data Management Server and SAS Data Quality Server  15

### Chapter 2 / Managing Table Metadata  17

- Overview of Managing Table Metadata  17
- Creating Table Metadata for a New Library  18
- Assessing Potential Changes in Advance  19
- Updating Your Table Metadata to Match Data in Your Physical Tables  20
- Using the Localize Table Batch Tool  24

### Chapter 3 / Assigning Libraries  31

- What Does It Mean to Assign a Library?  31
- How Do the Different Platform Clients Assign Libraries?  32
- Understanding Pre-Assigned Libraries  33
- Understanding Native Engines and the Metadata LIBNAME Engine  34
- Considerations for SAS Stored Process and SAS Pooled Workspace Servers  36
- Pre-assigning a Library  38
- Verifying Pre-assignments By Reviewing the Logs  43
- Limiting Access to Selected Libraries  43

### Chapter 4 / Connecting to Common Data Sources  45

- Overview of Connecting to Common Data Sources  48
- Overview of Connections to Relational Databases  48
- Using the Folders Tab in SAS Management Console  52
- Establishing Connectivity to a Library of SAS Data Sets  52
- Establishing Shared Access to SAS Data Sets  57
- Establishing Connectivity to a Microsoft Excel File  60
- Establishing Connectivity to a Flat File  61
- Establishing Connectivity to XML Data  64
- Establishing Connectivity to a SAS Information Map  65
- Establishing Connectivity to a Cloud Analytic Services (CAS) Server  68
- Establishing Connectivity to Several Different SAS/ACCESS Servers  71
Establishing Connectivity to a SAS Federation Server .......................... 79
Establishing Connectivity to Hadoop .................................................... 82
Establishing Connectivity to an Impala Server ...................................... 89
Establishing Connectivity to a SAS LASR Analytic Server ....................... 92
Establishing Connectivity to a MySQL Server .................................... 97
Establishing Connectivity to an ODBC Database Using Microsoft
  Windows NT Authentication ............................................................... 100
Establishing Connectivity to a Microsoft Access Database By Using ODBC 107
Establishing Connectivity to an OLE DB Database Using Microsoft
  Windows NT Authentication ............................................................... 111
Establishing Connectivity to an Oracle Database .................................. 117
Establishing Connectivity to a PI System Server .................................. 120
Establishing Connectivity to a Redshift Server .................................... 123
Establishing Connectivity to an SAP Server ......................................... 126
Establishing Connectivity to an SAP HANA Server ................................ 130
Establishing Connectivity to a SAS Scalable Performance Data Server .... 134
Establishing Connectivity to a SASHDAT Library ................................ 138
Establishing Connectivity to a Teradata Database ................................. 141
Registering and Verifying Tables .......................................................... 144
Read-only Access for Reporting Libraries ............................................ 146
Setting UNIX Environment Variables for SAS/ACCESS .......................... 147
Troubleshooting SAS/ACCESS Connections to Relational Databases ........ 147

Chapter 5 / Optimizing Data Storage ...................................................... 149
Overview of Optimizing Data Storage ................................................... 149
Compressing Data ............................................................................... 150
Indexing Data ...................................................................................... 152
Sorting Data ......................................................................................... 154
Buffering Data for Base SAS Tables ..................................................... 156
Buffering Data for SAS/ACCESS Databases ......................................... 157
Using Threaded Reads ......................................................................... 158
Validating SAS Scalable Performance Data Engine Hardware Configuration 158
Setting SAS LIBNAME Options That Affect Performance of SAS Tables .... 159
Setting SAS LIBNAME Options That Affect Performance of SAS/
  ACCESS Databases ............................................................................ 160
Setting SAS LIBNAME Options That Affect Performance of SAS
  SPD Engine Tables ............................................................................. 163
SAS Grid Computing Data Considerations ............................................. 165
Application Response Monitoring ......................................................... 165

Chapter 6 / Managing OLAP Cube Data ................................................... 167
Introduction to Managing OLAP Cube Data ......................................... 167
Data Storage and Access ....................................................................... 167
Exporting and Importing Cubes ......................................................... 168
About OLAP Schemas ......................................................................... 169
Create or Assign an OLAP Schema ...................................................... 169
Building a Cube ................................................................................... 170
Making Detail Data Available to a Cube for Drill-Through ....................... 172
Making Detail Data Available to an OLAP Server for Drill-Through ......... 173
Making Detail Data Available to an Information Map for Drill-Through .... 175
Display Detail Data for a Large Cube .................................................... 176

Chapter 7 / Library Options ..................................................................... 177
Library Options: Base SAS .................................................................... 177
What’s New in Data Administration for the SAS 9.4 Intelligence Platform

Overview

This document focuses on the SAS Intelligence Platform and establishing connectivity to your data sources (and data targets). Here are the changes and enhancements:

- integration of SAS with the DataFlux suite
- support for limiting access to libraries
- new localize table batch tool
- discontinued support for SAS Data Surveyor products
- support for on-disk AES encryption
- documentation enhancements

Integration of SAS with the DataFlux Suite

SAS has now fully integrated the DataFlux suite of data quality, data integration, data governance, and master data management solutions. Incorporating the DataFlux brand into SAS helps you build a more integrated information management approach.

Instead of being licensed separately, DataFlux products are now being combined into SAS software offerings that include other SAS products as well. For example, the SAS Data Management Advanced offering includes Base SAS and the SAS Metadata Server as well as DataFlux Data Management Studio and Data Management Server. These new offerings replace the current SAS Enterprise Data Integration Server offering and enable customers to choose from a broader array of data management products.

Some DataFlux products are changing their names to SAS, and others will in the future. In SAS 9.4, DataFlux Federation Server is now SAS Federation Server, and DataFlux qMDM Solution is now SAS MDM.
Support for Limiting Access to Libraries

In SAS 9.4M1, you can limit the scope and activities of a SAS server by putting it in a locked-down state. The locked-down SAS server can access only specified directories and files.

New Localize Table Batch Tool

The localize table batch tool creates or modifies localization metadata for data tables. There are three locale models:
- row per locale
- table per locale
- master/detail tables

Discontinued Support for SAS Data Surveyor Products

Starting with SAS 9.4, the following products are no longer supported:
- SAS Data Surveyor for Siebel
- SAS Data Surveyor for Salesforce.com
- SAS Data Surveyor for Oracle Applications
- SAS Data Surveyor for Clickstream Data
These products continue to be supported in SAS 9.3 and SAS 9.2.

Support for On-disk AES Encryption

In the second quarter of 2015, the ability to enable on-disk AES encryption for libraries on a Hadoop server was added to the SAS Management Console. You can enable or disable encryption on individual library definitions. You can also specify that the library definitions on this server inherit this encryption setting.
Documentation Enhancements

SAS 9.4M6: Changes and Enhancements

In SAS 9.4M6, the following new sections were added:

- Use SAS Environment Manager 2.6 Administration to Establish Connectivity to SAS LASR Analytic Server
- Establishing Connectivity to an Aster Server
- Establishing Connectivity to Several Different SAS/ACCESS Servers
- Establishing Connectivity to PI System Server

A new chapter entitled "Library Options" was added.

SAS 9.4M5: Changes and Enhancements

In SAS 9.4M5, the following new section was added:

- Establishing Connectivity to a Cloud Analytic Services (CAS) Server

SAS 9.4M4: Changes and Enhancements

In SAS 9.4M4, the following new section was added:

- Establishing Connectivity to a Redshift Server

SAS 9.4M3: Changes and Enhancements

In SAS 9.4M3, the following new sections were added:

- Hadoop Data
- Establishing Connectivity to an Impala Server

In SAS 9.4M3, if SAS Visual Analytics is part of your deployment, you can create and maintain SAS LASR Analytic Servers and SAS LASR Analytic Server libraries using the SAS Environment Manager. The following sections were added:

- Stage 1: Register a SAS LASR Analytic Server Using SAS Environment Manager
SAS 9.4M2: Changes and Enhancements

In SAS 9.4M2, the following sections were added:

- Establishing Connectivity to an ODBC Database Using Microsoft Windows NT Authentication
- Establishing Connectivity to an OLEDB Database Using Microsoft Windows NT Authentication
- Establishing Connectivity to a Teradata Database
- Establishing Connectivity to a SAS LASR Analytic Server
- Establishing Connectivity to a SASHDAT Library

Generic instructions for using the User Manager plug-in within the SAS Management Console were added:

- Define the DBMS User
- Define the DBMS User Group

Starting in SAS 9.4M2, a new support site for Hadoop was added to the SAS Third-Party Software Reference site.
Overview

This chapter describes the features of the most common data sources that you encounter as you perform administrative tasks. In addition, a simple diagram is provided for each data source that shows how the data flows as connections are established between source storage, SAS engines and servers, and SAS applications.
Hadoop Data

Hadoop is an open-source software framework that provides distributed storage and processing of large amounts of data. The data is divided into blocks and stored across multiple connected nodes (computers) that work together. These connected nodes form a cluster. The Hadoop Distributed File System (HDFS) provides distributed data storage and processing. HiveServer2 is a distributed data warehouse component that is built on top of HDFS. HiveServer2 includes the SQL query language HiveQL.

Several SAS products access Hadoop data:

- SAS/ACCESS Interface to Hadoop enables you to access Hadoop data through HiveServer2 and from HDFS as if the data were native SAS data sets.
- SAS Scalable Performance Data (SPD) Engine is a Base SAS engine that enables you to interact with Hadoop through HDFS. You can write data, retrieve data for analysis, and perform administrative functions.
- SAS LASR Analytic Server is a scalable, analytic platform that provides a secure, multi-user environment for concurrent access to in-memory data. The SASHDAT engine adds and deletes SASHDAT files in HDFS.
- SAS Scalable Performance Data (SPD) Server stores and accesses SPD Server tables in HDFS.

For more information, see the following:

- “Establishing Connectivity to Hadoop” on page 82
- “Establishing Connectivity to a SAS LASR Analytic Server” on page 92
- “SAS Scalable Performance Data Server and SAS Scalable Performance Data Engine” on page 11
- “Establishing Connectivity to a SASHDAT Library” on page 138

SAS Data Sets

SAS data sets (tables) are the default SAS storage format. You can use them to store data of any granularity. A SAS table is a SAS file stored in a SAS library that SAS creates and processes. A SAS table contains data values that are organized as a table of observations (rows) and variables (columns) that can be processed by SAS software. A SAS table also contains descriptor information such as the data types and lengths of the columns, as well as which engine was used to create the data. For more information about using default SAS storage, see SAS Language Reference: Concepts and SAS Data Set Options: Reference. The following figure shows how connectivity to SAS data sets is configured.
For a detailed example of a SAS data set connection, see “Establishing Connectivity to a Library of SAS Data Sets” on page 52.

Note: On UNIX systems, SAS data set names must be all uppercase or all lowercase. UNIX does not read mixed case SAS data set names such as MYDATASET.sas7bdat. In this example, the data set name must be either MYDATASET:SAS7BDAT or mydataset.sas7bdat.
architecture from the SAS Application Server, if the site is licensed for that configuration.

**Figure 1.2 Establishing Shared Access to SAS Data Sets**

For a detailed example of a shared SAS data set connection, see “Establishing Shared Access to SAS Data Sets” on page 57.

---

**Local and Remote Access to Data**

To access data, you must register the data as a library in SAS Management Console. The procedures for accessing data and registering data are explained later in this document. However, one of the important details for file-based data, such as SAS data sets, is that you need to specify the file system path to the data. This path is needed so that a SAS Application Server can access it. As shown in the following figure, SAS data sets that are local to the SAS Application Server have a fully qualified path such as `C:\data\sourcetables`.

**Figure 1.3 SAS Workspace Server Accessing Local Data Sets**
Often, file-based data is stored on a host that is remote from the SAS Application Server. When the hosts have a network path for shared directories such as a Windows UNC path or UNIX NFS, that path is used. The following figure shows an example of a SAS Workspace Server accessing a UNC path, `\dataserver\sourcetables`, on a data server.

Note: Data cannot be accessed via mapped drives on the SAS Application Server. You must use the UNC path.

**Figure 1.4** SAS Workspace Server Accessing Remote Data Sets

Note: This figure shows a SAS Workspace Server accessing data over a shared file system. To access data over network connection (without the file system), use SAS/SHARE as described in this document.

---

**External Files**

An external file is a file that is maintained by the machine operating environment or by a software product other than SAS. A flat file with comma-separated values is one example. SAS Data Integration Studio provides three source designer wizards that enable you to create metadata objects for external files:

- the delimited external file wizard for external files in which data values are separated with a delimiter character. This wizard enables you to specify multiple delimiters, nonstandard delimiters, missing values, and multi-line records.
- the fixed-width external file wizard for external files in which data values appear in columns that are a specified number of characters wide. This wizard enables you to specify non-contiguous data.
- the user-written external file wizard for complex external files that require user-written SAS code to access their data.

The external file source designer wizards enable you to do the following:

- display a raw view of the data in the external file
display a formatted view of the data in the external file, as specified in the SAS metadata for that file

display the SAS DATA step and SAS INFILE statement that the wizard generates for the selected file

display the SAS log for the code that is generated by the wizard

specify options for the SAS INFILE statement that is generated by the wizard, such as National Language Support (NLS) encoding

override the generated SAS INFILE statement with a user-written statement

supply a user-written SAS DATA step to access an external file

The following figure shows establishing connectivity to external files:

Figure 1.5 Establishing Connectivity to External Files

For a detailed example of an external file connection, see “Establishing Connectivity to a Flat File” on page 61.

XML Data

The XML LIBNAME engine works in a way similar to other SAS engines. A LIBNAME statement is executed so that a libref is assigned and an engine is specified. That libref is then used throughout the SAS session.

Instead of the libref being associated with the physical location of a SAS library, the libref for the XML engine is associated with a physical location of an XML document. When you use the libref that is associated with an XML document, SAS either translates the data in a SAS data set into XML markup or translates the XML markup into SAS format.

The XML LIBNAME engine can read input streams from a web service input and write an output stream to a web service output. The XML LIBNAME engine supports reading XML files in complex structures using XMLMaps. An XMLMap is a user-defined file that contains XML tags that tell the XML LIBNAME engine how to interpret an XML document. XMLMaps are defined using the SAS XML Mapper.
XML files are written by the XML Writer transformation provided by SAS Data Integration Studio. The XML LIBNAME engine supports Output Delivery System (ODS) tag sets; XMLMaps are not supported for writing. The XML Writer transformation in SAS Data Integration Studio is shipped with a sample ODS tag set, if needed. An output XML document can either be:

- used by a product that processes XML documents
- moved to another host for the XML LIBNAME engine to process by translating the XML markup back to a SAS data set

Because the XML LIBNAME engine is designed to handle tabular data, all the data sent to or from a web service must be in table form.

The following figure shows connectivity to XML files:

Figure 1.6 Establishing Connectivity to XML Files

Message Queues

Message queues are collections of data objects that enable asynchronous communication between processes. These processes are typically applications that run on different computers, and might be configured in a heterogeneous network. Queue management software ensures that messages are transmitted without error. SAS Data Integration Studio can perform messaging jobs to read and write messages to Microsoft MSMQ as well as IBM WebSphere MQ. For more information about administering message queues, see SAS Intelligence Platform: Desktop Application Administration Guide. For more information about creating messaging jobs, see SAS Data Integration Studio: User’s Guide.
Relational Database Sources

SAS/ACCESS

Data can also be stored in third-party hierarchical and relational databases such as DB2, Oracle, Microsoft SQL Server, and Teradata. SAS/ACCESS interfaces provide fast, efficient reading and writing of data to these facilities.

Several of the SAS/ACCESS engines support threaded Reads. You can use this to read entire blocks of data on multiple threads instead of reading data just one record at a time. This feature can reduce input and output bottlenecks so that thread-enabled procedures can read data quickly. These engines can also access database management system (DBMS) data in parallel by using multiple threads to the parallel DBMS server.

Here are some of the SAS/ACCESS engines that support this functionality:

- DB2 under UNIX and PC Hosts
- ODBC
- Oracle
- Sybase
- Teradata

For more information about using the SAS/ACCESS interfaces, see SAS/ACCESS for Relational Databases: Reference. The following figure shows how connectivity to Oracle databases is configured:

![Figure 1.7 Establishing Connectivity to Oracle Databases](image)

For a detailed example of an Oracle connection, see “Establishing Connectivity to an Oracle Database” on page 117.
ODBC Sources

Open database connectivity (ODBC) standards provide a common interface to a variety of databases such as DB2, Microsoft Access, Oracle, and Microsoft SQL Server databases. Specifically, ODBC standards define application programming interfaces (APIs) that enable an application to access a database if the ODBC driver complies with the specification.

**TIP** If a SAS/ACCESS engine is available for a database, then performance is better with the SAS/ACCESS engine rather than with the ODBC Driver interface.

The basic components and features of ODBC include the following:

- ODBC functionality is provided by three components: the client interface, the ODBC driver manager, and the ODBC driver. SAS provides the SAS/ACCESS interface to ODBC, which is the client interface. For PC platforms, Microsoft developed the ODBC Administrator, which is used from the Windows Control Panel to perform software administration and maintenance activities. The ODBC driver manager also manages the interaction between the client interface and the ODBC driver. On UNIX platforms, a default ODBC driver manager does not exist and SAS does not provide a driver manager with SAS/ACCESS to ODBC. For UNIX platforms, you should obtain an ODBC driver manager from your ODBC driver vendor.

- The ODBC administrator defines a data source as the data that is used in an application and the operating system and network that are used to access the data. You create a data source by using the ODBC Administrator in the Windows Control Panel and then selecting an ODBC driver. You then provide the information (for example, data source name, user ID, password, description, and server name) that is required by the driver to make a connection to the desired data. The driver displays dialog boxes in which you enter this information. During operation, a client application usually requests a connection to a named data source, not just to a specific ODBC driver.

- An ODBC Administrator tool is not available in a UNIX environment such as HP-UX, AIX, or Solaris. During an install, the driver creates a generic .odbc.ini file that can be edited to define your own data sources.

The following figure shows how ODBC is used to establish connectivity to Oracle databases:
Figure 1.8 Establishing Connectivity to Oracle Databases By Using ODBC

The following figure shows how ODBC is used to establish connectivity to Access databases:

Figure 1.9 Establishing Connectivity to Access Databases By Using ODBC

For a detailed example of an ODBC-based Access connection, see “Establishing Connectivity to a Microsoft Access Database By Using ODBC” on page 107.
Both the SAS Scalable Performance Data (SPD) Engine and the SAS Scalable Performance Data (SPD) Server are designed for high-performance data delivery. They enable rapid access to SAS data for intensive processing by the application. The SPD Engine and SPD Server deliver data to applications rapidly by organizing the data into a streamlined file format that takes advantage of multiple CPUs and I/O channels to perform parallel input and output functions.

The SPD Engine is a SAS LIBNAME engine that is included with Base SAS software. It is a single-user data storage solution that shares the high-performance parallel processing and parallel I/O capabilities of SPD Server, but it is not a server. It can read and store data in SAS libraries, including data in Hadoop HDFS. For more information about SPD Engine data in HDFS, see SAS SPD Engine: Storing Data in the Hadoop Distributed File System.

The SPD Server is not part of Base SAS. It is a multi-user parallel-processing data server that includes an SPD Server LIBNAME engine and a comprehensive security infrastructure, backup and restore utilities, and sophisticated administrative and tuning options. The SPD Server libraries can be defined using SAS Management Console.

SPD Engine and SPD Server use multiple threads to read blocks of data very rapidly and in parallel. The software tasks are performed in conjunction with an operating system that enables threads to execute on any of the available CPUs. Although threaded I/O is an important function of both products, their real power comes from how the software structures SAS data. They can read and write partitioned files and, in addition, use a specialized file format. This data structure permits threads, running in parallel, to perform I/O tasks efficiently.

Although not intended to replace the default Base SAS engine for most tables that do not span volumes, SPD Engine and SPD Server are high-speed alternatives for processing very large tables. They read and write tables that contain billions of observations.

The SPD Engine and SPD Server performance are boosted in these ways:

- support for terabytes of data
- scalability on symmetric multiprocessing (SMP) machines
- parallel WHERE selections
- parallel loads
- parallel index creation
- partitioned tables
parallel I/O data delivery to applications
implicit sorting on BY statements

Symmetric Multiprocessing

The SPD Server exploits a hardware and software architecture known as symmetric multiprocessing (SMP). An SMP machine has multiple CPUs and an operating system that supports threads. An SMP machine is usually configured with multiple disk I/O controllers and multiple disk drives per controller. When the SPD Server reads a data file, it launches one or more threads for each CPU; these threads then read data in parallel. By using these threads, an SPD Server that is running on an SMP machine provides the quick data access capability that is used by SAS in an application.

For more information about using the SPD Server, see SPD Server and SAS Scalable Performance Data Server: Administrator’s Guide.

The following figure shows how connectivity to SPD Servers is established:

*Figure 1.10 Establishing Connectivity to an SPD Server*

For a detailed example of an SPD Server connection, see “Establishing Connectivity to a SAS Scalable Performance Data Server” on page 134.

Dynamic Clustering

The SPD Server provides a virtual table structure called a clustered data table. A cluster contains a number of slots, each of which contains an SPD Server table. The clustered data table uses a layer of metadata to manage the slots.

This virtual table structure provides the SPD Server with the architecture to offer flexible storage to allow a user to organize tables based on values contained in character and numeric columns, including SAS date, time, or datetime values. This new type of organization is called a dynamic cluster table. Dynamic cluster tables enable parallel loading and selective removal of data from very large tables, making management of large warehouses easier. These unique capabilities provide
organizational features and performance benefits that traditional SPD Server tables cannot provide.

Dynamic cluster tables can load and process data in parallel. Dynamic cluster tables provide the flexibility to add new data or to remove historical data from the table by accessing only the slots affected by the change, without having to access the other slots, thus reducing the time needed for the job to complete. In addition, a complete refresh of a dynamic cluster table requires a fraction of the disk space that would otherwise be needed, and can be divided into parallel jobs to complete more quickly. All of these benefits can be realized using simple SPDO procedure commands to create and alter a cluster.

The two most basic commands are CLUSTER CREATE and CLUSTER UNDO. Two additional commands are CLUSTER ADD and CLUSTER LIST. You execute each of these commands within PROC SPDO.

The CLUSTER CREATE command requires two options:

- the name of the cluster table to create (cluster-table-name)
- a list of SPD Server tables to include in the cluster (using the MEMBER= option)

The following example shows the syntax for PROC SPDO with a CLUSTER CREATE command:

```
PROC SPDO LIBRARY=domain-name;
CLUSTER CREATE cluster-table-name
MEMBER=SPD-Server-table1
MEMBER=SPD-Server-table2
MEMBER=SPD-Server-table3
MEMBER=SPD-Server-table4
MEMBER=SPD-Server-table5
MEMBER=SPD-Server-table6
MEMBER=SPD-Server-table7
MEMBER=SPD-Server-table8
MEMBER=SPD-Server-table9
MEMBER=SPD-Server-table10
MEMBER=SPD-Server-table11
MEMBER=SPD-Server-table12;
QUIT;
```

Here is the syntax for the UNDO command:

```
PROC SPDO
LIBRARY=domain-name;
CLUSTER UNDO sales_hist;
QUIT;
```

This example shows the syntax for the ADD command:

```
PROC SPDO
LIBRARY=domain-name;
CLUSTER ADD sales_hist
MEMBER=2015sales_table1
MEMBER=2015sales_table2
MEMBER=2015sales_table3
MEMBER=2015sales_table4
MEMBER=2015sales_table5
MEMBER=2015sales_table6; 
QUIT;
```

Finally, here is the syntax for the LIST command:

```
```
PROC SPDO
LIBRARY=domain-name;
CLUSTER LIST sales_hist;
QUIT;

These operations run quickly. These features reduce the downtime of the table for maintenance and improve the availability of the warehouse.

SAS Data Surveyors

Data Surveyor for SAP

The Data Surveyor for SAP remains as in previous versions. It contains Java plug-ins to SAS Data Integration Studio and SAS Management Console, plus the required SAS/ACCESS engine necessary to get the information out of the DBMS system. Understanding the metadata of this business application is at the heart of the data surveyor. The SAP Data Surveyor has knowledge about the structure of the tables deployed in SAP. This knowledge contains information that enables you to do the following:

- understand complex data structures
- navigate the large amounts of tables (SAP has more than 20,000)

For a detailed example of an SAP server connection, see “Establishing Connectivity to an SAP Server” on page 126.

Other SAS Data Surveyors

Starting with SAS 9.4, the following products are no longer supported:

- SAS Data Surveyor for Siebel
- SAS Data Surveyor for Salesforce.com
- SAS Data Surveyor for Oracle Applications

These products continue to be supported in SAS 9.3 and SAS 9.2.

Change Data Capture

Data extraction is an integral part of all data warehousing projects. Data is often extracted on a nightly or regularly scheduled basis from transactional systems in bulk and transported to the data warehouse. Typically, all the data in the data warehouse is refreshed with data extracted from the source system. However, an entire refresh involves the extraction and transportation of huge volumes of data and is very expensive in both resources and time. With data volumes now doubling
yearly in some organizations a new mechanism known as change data capture (CDC) is increasingly becoming the only viable solution for delivering timely information into the warehouse to make it available to the decision makers. CDC is the process of capturing changes made at the data source and applying them throughout the enterprise. CDC minimizes the resources required for ETL processes because it deals only with data changes. The goal of CDC is to ensure data synchronicity. SAS offers a number of CDC options.

- Some database vendors (Oracle 10g) provide tables of just changed records. These tables can be registered in SAS Data Integration Studio and used in jobs to capture changes.
- SAS Data Integration Studio allows the user to determine changes and take appropriate action.
- SAS has partnered with Attunity, a company that specializes in CDC. Their Attunity Stream software provides agents that non-intrusively monitor and capture changes to mainframe and enterprise data sources such as VSAM, IMS, ADABAS, DB2, and Oracle. SAS Data Integration Studio provides a dedicated transformation for Attunity.

The Attunity solution does the following:

- moves only CHANGES to the data
- requires no window of operation
- provides higher frequency and reduced latency transfers. It is possible for multiple updates each day, providing near-real-time continuous change flow.
- reduces the performance impact of the following activities:
  - rebuilding of target table indexes
  - recovering from a process failure that happens mid-stream

DataFlux Data Management Server and SAS Data Quality Server

SAS 9.4 includes new software that consists of SAS and SAS DataFlux products, including the Standard and Advanced offerings for SAS Data Quality and SAS Data Management. The SAS Data Quality offerings include the product, SAS Data Quality Server, a Quality Knowledge Base, and other products. This data quality software enables you to analyze, standardize, and transform your data to increase the accuracy and value of the knowledge that you extract from your data.

SAS Data Quality Server consists of SAS language elements and a Quality Knowledge Base that you can customize for your company. The language elements analyze and cleanse data by referencing data definitions in the Quality Knowledge Base. SAS Data Quality Server also provides a SAS language interface to the DataFlux Data Management software.

The SAS Data Management offerings include DataFlux Data Management Server, additional Quality Knowledge Bases, DataFlux Data Management Studio, and other SAS and SAS DataFlux products. DataFlux Data Management Server runs jobs and real-time services that are created in DataFlux Data Management Studio. The jobs and real-time services can be executed by SAS programs that contain the
procedures and functions in SAS Data Quality Server. Among its many capabilities, the DataFlux Data Management Studio software enables you to create jobs and real-time services and customize the data definitions in Quality Knowledge Bases.

SAS Data Integration Studio is part of the SAS Data Integration Server offering, and it provides software that enables data quality applications. SAS provides data quality transformations that enable you to analyze data, cleanse data, or trigger the execution of DataFlux jobs or real-time services on DataFlux Data Management Servers.

The data quality software requires setup and configuration after installation. For administrative information, see “Administering SAS Data Integration Studio” in SAS Intelligence Platform: Desktop Application Administration Guide.
Overview of Managing Table Metadata

As explained in “Registering and Verifying Tables” on page 144, one way to create metadata for the tables in a library is to register the tables with SAS Management Console. You can also create this metadata programmatically by using PROC METALIB. PROC METALIB also provides options for maintaining your table metadata that are not available in SAS Management Console. For example, by default PROC METALIB registers metadata for any physical tables that are not already registered in the metadata and updates the table metadata for tables that have been altered since they were registered.

By using optional statements, you can also use PROC METALIB to perform the following tasks:

- Delete metadata for tables that have been removed from the library.
- Produce a report that lists the changes made by the procedure or the potential changes that will be made when the procedure is executed.
- Operate on a subset of the tables in a library.

Note: For detailed information about the procedure, see "METALIB Procedure" in SAS Language Interfaces to Metadata.
Note: PROC METALIB cannot work with a library that is registered with the Pre-assigned Library resource template. When pre-assigning a library, be sure to choose the resource template specific to the type of data source library that you are creating and select the Library is pre-assigned check box. Do not use the specialized Pre-Assigned Library template. Also, if the library is pre-assigned and you run the PROC METALIB in a Foundation SAS session, you must make sure the library is allocated by either submitting a LIBNAME statement for the library in that SAS session, or by using the META_AUTORESOURCEs option to access the library through a SAS server to which the library is assigned.

The remainder of the chapter presents examples of how PROC METALIB is commonly used. The examples assume that you have set the following metadata server connection options in your SAS session:

```sas
options METAUSER="metadata-server-userid"
   METAPASS="metadata-server-password"
   METAPORT=metadata-server-port
   METASERVER="metadata-server-machine";
```

If you have not set these options, you can use PROC METALIB parameters to specify this information.

---

**Creating Table Metadata for a New Library**

When you first register a SAS library, it has no related table metadata. You can add this metadata by using the Register Tables wizard in SAS Management Console (see “Overview of Managing Table Metadata” on page 17), or by using PROC METALIB. Before you can successfully run PROC METALIB code, you must have ReadMetadata and WriteMetadata access to the library metadata object. You must have ReadMetadata and WriteMemberMetadata access on the metadata folder to use for storing the table metadata. ReadMetadata and WriteMetadata must also be granted in the Default ACT in order to create any new metadata objects.

The following example shows how to use PROC METALIB to create initial table definitions for the tables in a library. The REPORT statement causes the procedure to write information about the table definitions that it creates.

```sas
proc metalib;
   omr (library="lib1" repname="Foundation" );
   report;
run;
```

The report that is generated by the code sample resembles the following:
Before you use PROC METALIB to update existing table metadata, it is a good idea to execute the procedure with the NOEXEC and REPORT statements. The NOEXEC statement tells the procedure not to actually add, update, or delete any metadata. The REPORT statement tells the procedure to create a report that explains what actions it would have taken if the NOEXEC statement had not been present. If you want to make all of the changes that are shown in the report, you can then remove the NOEXEC statement and rerun the procedure to update the metadata.

The following example shows how to use the NOEXEC and REPORT statements to assess potential metadata changes:

```sas
proc metalib;
  oem (library="lib1" repname="Foundation");
  update_rule=(delete);
  noexec;
  report;
run;
```

Note: The UPDATE_RULE statement tells the procedure to delete table definitions for any tables that have been deleted from the library. For more information about this statement, see "Changing the Update Rule" on page 21.

The report that is generated by the code sample resembles the following:
Updating Your Table Metadata to Match Data in Your Physical Tables

Adding and Updating Table Metadata

By default, PROC METALIB creates table definitions for any tables in the library that do not have table definitions and updates any table definition that does not reflect the current structure of the table that it represents. However, it does not delete table metadata.

Use REPORT when you want an output listing that summarizes metadata changes, either before changes are made (by using NOEXEC) or to see afterward what changes were actually made. SAS output is the default.

Example: Default PROC METALIB Behavior

The following example uses the default PROC METALIB behavior. Summary notes are written to the SAS log regardless of whether you request a report. Unlike the
example shown in "Assessing Potential Changes in Advance" on page 19, the summary does not mention any deleted tables.

```sas
proc metalib;
  omr (library="lib1" repname="Foundation");
run;
```

Without the REPORT statement, the options are written to the SAS log. The output resembles the following example:

```
105  proc metalib;
106    omr  (library="lib1" repname="Foundation");
107  run;

NOTE: A total of 3 tables were analyzed for library "lib1".
NOTE: Metadata for 2 tables was updated.
NOTE: Metadata for 1 tables was added.
NOTE: Metadata for 0 tables matched the data sources.
NOTE: 0 other tables were not processed due to error or UPDATE_RULE.
NOTE: PROCEDURE METALIB used (Total process time):
    real time           0.20 seconds
    cpu time            0.06 seconds
```

Changing the Update Rule

By using the optional UPDATE_RULE statement, you can change the default behavior of PROC METALIB. The principal rules that you can specify are shown as follows:

NOADD
- specifies not to add table metadata to the metadata repository for physical tables that have no metadata.

NOUPDATE
- specifies not to update existing table metadata to resolve discrepancies with the corresponding physical tables.

DELETE
- specifies to delete table metadata if a corresponding physical table is not found in the specified library.

Example: Adding, Updating, and Deleting Metadata

The following example shows how to use PROC METALIB to add metadata for new tables, update table definitions where necessary, and also delete table definitions that are no longer valid. (You can also perform these functions using SAS Data Integration Studio.)

```sas
proc metalib;
  omr (library="lib1" repname="Foundation");
  update_rule=(delete);
  report;
run;
```
The following example shows how to use UPDATE_RULE with DELETE, NOADD, and NO UPDATE to delete table definitions that are no longer valid, as well as suppress the default add and update actions:

```sas
proc metalib;
  omr (library="lib1" repname="Foundation");
  update_rule (delete noadd noupdate);
  report;
run;
```

The resulting SAS output resembles the following example:

![Sample Summary Report](image)

**Figure 2.3  Sample Summary Report**

### Specifying Which Tables Are Affected

You can use the optional SELECT or EXCLUDE statements to perform an operation against a subset of the tables in a library. SELECT and EXCLUDE are mutually exclusive, so you should use only one or the other.

When you set the SELECT statement, you can choose the tables for processing:

- For tables, specify their SAS name. If no table definition is found in metadata, it is created in the repository that contains the library object. If a matching table definition is found in metadata, it is compared to the physical table. If differences are found, the table definition is updated in metadata.

- For tables already registered in metadata, specify either the unique metadata identifier or the value in the SASTableName attribute. If you specify the metadata identifier, only the specified table definition is updated, not the first table definition in the association list.

If you use the SASTableName value and there is more than one metadata table object with the same name, then PROC METALIB impacts only the first one that is found. By using the metadata ID instead, you guarantee that PROC METALIB affects the specific table that you want to manage.

You can use EXCLUDE to specify a single table or a list of tables to exclude from processing.
Example: Specifying Tables

The following example shows how to use SELECT to process only a subset of tables:

```sas
proc metalib;
  omr (library="lib1" repname="Foundation");
  select(lima);
  report;
run;
```

The resulting SAS output resembles the following example:

![SAS Output Example](image)

The following example shows how to use EXCLUDE to exclude a specific subset of tables:

```sas
proc metalib;
  omr (library="lib1" repname="Foundation");
  exclude(licj libp);
  noexec;
  report;
run;
```
Using the Localize Table Batch Tool

About the Localize Table Batch Tool (sas-localize-table)

The SAS Intelligence Platform provides the localize table batch tool (sas-localize-table) to create or modify localization metadata for data tables.

The `sas-localize-table` batch tool is located in the following path:


Syntax for the Localize Table Batch Tool

```
sas-localize-table connection-options table-location-and-name
<options>
  connection-options
    <<-? | -- help>>
    <<domain domain-name>
    <<host host-name>
    <<log log-file-name>
    <<password password>
    <<port port-name>
    <<profile file-name>
    <<user user-ID>
  table-location-and-name
    <<key column-name>
    <<locale column-name>
    <<localize column=detail-table-path>
    <<reset>
    <<rowPerLocale>
    <<tablePerLocale>
```

Syntax Descriptions for the Localize Table Batch Tool

`connection-options`

Specifies connection options to log on to a SAS Metadata Server. This is a required argument. You must either specify `-profile`, `-host`, `-port`, `-user`, and `-password` options.
-?  
  --help
      Prints help information.

-domain domain-name
      Specifies the user authentication domain.

-host host-name
      Specifies the SAS Metadata Server host. This option is required if -profile is not set.

-log log-file-name
      Specifies the log file or directory.

-password password
      Specifies the user login password. This option is required if -profile is not set or if the profile does not contain connection credentials.

-port port-name
      Specifies the SAS Metadata Server port. This option is required if -profile is not set.

-profile file-name
      Specifies the SAS Metadata Server connection profile. This option can be used in place of the -host, -port, -user, and -password options.

-user user-ID
      Specifies the user login identity. This option is required if -profile is not set or if the profile does not contain connection credentials.

-table-location-and-name
      Specifies the metadata location (relative to SAS Folders) and name of the table that is being localized. This is a required argument.

-key column-name
      Specifies the key column for a localization detail table.

-locale column-name
      Specifies the locale column for a localization detail table.

-localize column=detail-table-path
      Specifies columns to be localized with a localization detail table. This option can be repeated.

-reset
      Erases all localization metadata for this table.

-rowPerLocale
      Specifies that the table uses the row per locale localization model.

-tablePerLocale
      Specifies that the table uses the table per locale localization model.

Command Examples of sas-localize-table

The following command shows localization of current table:

  sas-localize-table connect-options table-location-and-name

The following command erases all localization metadata of the current table:

  sas-localize-table connect-options table-location-and-name -reset
The following command specifies the columns to localize for the detail table:
```
sas-localize-table connect-options table-location-and-name -localize column-detail-table-path
```

The following command specifies the location of the detail table:
```
sas-localize-table connect-options table-location-and-name -key column-name -locale column-name
```

The following command associates the table with the row per locale model:
```
sas-localize-table connect-options table-location-and-name -rowPerLocale -locale column-name
```

The following command associates the table with the table per locale model:
```
sas-localize-table connect-options table-location-and-name -tablePerLocale
```

Examples for Applying a Metadata Localization Model

Overview

There are three metadata models to use for localizing your tables.
- “Row per Locale Model Example” on page 26
- “Table per Locale Model Example” on page 27
- “Master/Detail Table Model Example” on page 28

Row per Locale Model Example

The row per locale model uses a single table to contain all the localized data. A locale column identifies which rows are associated with a given locale. The table data is filtered for the current locale. The advantage of using the row per locale model is that there are no schema changes required to add a new locale.

Adding new locale data to the table requires adding data to the PRODUCT table. This action forces a rebuild of the indexes on the table. Also, the non-localized data must be repeated for every locale.

Figure 2.4  PRODUCT Localization Detail Table

```
<table>
<thead>
<tr>
<th>PRODUCT</th>
<th>CHAR(20)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRODUCT_ID</td>
<td>CHAR(20)</td>
</tr>
<tr>
<td>PRODUCT_LOCALE</td>
<td>CHAR(5)</td>
</tr>
<tr>
<td>PRODUCT_SKU</td>
<td>CHAR(10)</td>
</tr>
<tr>
<td>PRODUCT_COLOR</td>
<td>CHAR(20)</td>
</tr>
<tr>
<td>PRODUCT_DESC</td>
<td>CHAR(255)</td>
</tr>
</tbody>
</table>
```

The following code creates a single table named PRODUCT, which contains the product information and localized data:
CREATE TABLE PRODUCT (
    PRODUCT_ID CHAR(20) NOT NULL,
    PRODUCT_LOCALE CHAR(5) NOT NULL,
    PRODUCT_SKU CHAR(10),
    PRODUCT_COLOR CHAR(20) character set utf8,
    PRODUCT_DESC CHAR(255) character set utf8);

INSERT INTO PRODUCT VALUES (
    'ball_33', 'en_US', '337718N', 'red', 'paddle ball replacement');
INSERT INTO PRODUCT VALUES (
    'ball_33', 'es_ES', '337718N', 'rojo', 'sustitución de paddle');
INSERT INTO PRODUCT VALUES (
    'ball_33', 'fr_FR', '337718N', 'rouge', 'paddle ball de remplacement');

ALTER TABLE PRODUCT ADD PRIMARY KEY (PRODUCT_ID, PRODUCT_LOCALE);

SELECT PRODUCT_ID, PRODUCT_COLOR, PRODUCT_DESC FROM PRODUCT
WHERE PRODUCT_ID='?' ANDPRODUCT_LOCALE='?';

After you register the PRODUCT table in metadata, use the sas-localize-table tool to
associate the localization in metadata. For example, if the path to the library is '/
Shared Data/Product Data', your sas-localize-table command is as follows:

sas-localize-table <connect-options> '/Shared Data/Product Data/Product'
-rowPerLocale -locale PRODUCT_LOCALE

Table per Locale Model Example

The table per locale model uses a separate data table for each supported locale. Because each locale has a table, new locales can be added without disturbing other data.

Advantages of the table per locale model include the following:

- Character set encoding or sort order can be customized for each locale. For example, order by PRODUCT_DESC could be made locale-sensitive at DDL time.
- Queries that do not involve localized data can go against the default table.
- Code that accesses the table runs normally, even if it is not aware of the localization model.

Disadvantages of the table per locale model include the following:

- Schema changes are required.
- All of the non-localized data must be included in every localized table.
- Queries are more complex because the table names vary by locale.

Figure 2.5  Tables for Multiple Locales
The following code creates a table for each locale:

```sql
CREATE TABLE PRODUCT (PRODUCT_ID CHAR(20) NOT NULL,
    PRODUCT_SKU CHAR(10),
    PRODUCT_COLOR CHAR(20) character set ASCII,
    PRODUCT_DESC CHAR(255) character set ASCII);

CREATE TABLE PRODUCT_ES_ES (PRODUCT_ID CHAR(20) NOT NULL,
    PRODUCT_SKU CHAR(10),
    PRODUCT_COLOR CHAR(20) character set ISO8859_1,
    PRODUCT_DESC CHAR(255) character set ISO8859_1);

CREATE TABLE PRODUCT_FR_FR (PRODUCT_ID CHAR(20) NOT NULL,
    PRODUCT_SKU CHAR(10),
    PRODUCT_COLOR CHAR(20) character set ISO8859_1,
    PRODUCT_DESC CHAR(255) character set ISO8859_1);

INSERT INTO PRODUCT VALUES ('ball_33', '337718n', 'red', 'paddle ball replacement');
INSERT INTO PRODUCT_ES_ES VALUES ('ball_33', '337718n', 'red', 'sustitución de paddle');
INSERT INTO PRODUCT_FR_FR VALUES ('ball_33', '337718n', 'rojo', 'paddle ball de remplacement');

ALTER TABLE PRODUCT ADD PRIMARY KEY (PRODUCT_ID);
ALTER TABLE PRODUCT_ES_ES ADD PRIMARY KEY (PRODUCT_ID);
ALTER TABLE PRODUCT_FR_FR ADD PRIMARY KEY (PRODUCT_ID);

SELECT PRODUCT_ID, PRODUCT_COLOR, PRODUCT_DESC FROM ? WHERE PRODUCT_ID=?;
```

After you register the PRODUCT table and the localized tables in metadata, use the `sas-localize-table` tool to associate the localized table in metadata. For example, if the path to the library is '/Shared Data/Product Data', your `sas-localize-table` command is as follows:

```
sas-localize-table <connect-options> '/Shared Data/Product Data/Product'
-localePerLocale
```

### Master/Detail Table Model Example

The localization detail table must have three columns:

1. key column
2. locale column
3. value column

The detail tables model uses separate detail tables for each column that requires localization. The data column contains a locale invariant key value that maps to a key column in the detail table. The detail table also contains a locale column and one or more value columns.

A full view of the data table is generated by performing a left join of the data table to the detail table and filtering based on the current locale. The localized value columns from the detail table are joined to the unlocalized columns in the original data table.
Advantages of using separate detail tables include the following:

- No schema changes are required to add a new locale.
- Non-localized data is not repeated.
- Localization details can be shared across multiple data tables.
- Assuming that a left join is used, missing values, instead of missing rows, are provided for data that is not localized.
- Queries that do not involve localized data do not require any special processing.

**Figure 2.6  Master/Detail Table**

The models create a detail table entitled PRODUCT. The key column (PRODUCT_ID) is matched to the column value in the original data table. The locale column (PRODUCT_LOCALE) value specifies the locale for a given row. Once you find the row in the localization detail table with the key value matching your original data and the appropriate locale, you can use the value column to get the localized value.

When you use the sas-localize-table batch tool to set up a localization detail table, you are doing the following:

- marking the table as a localization detail table
- identifying the key column
- identifying the locale column

You do not need to identify the value column because any column that is not a key or locale column is assumed to be a value column.

The next step is to associate a localization detail table with a specific column in a data table. The specified column in the data table has values that can be looked up in the key column of the detail table.

The following code creates two tables:

- a master table containing the non-localized information
- a detail table containing the localized information

```sql
CREATE TABLE PRODUCT (  
    PRODUCT_ID CHAR(20) NOT NULL PRIMARY KEY,  
    PRODUCT_Sku CHAR(10) );
```
CREATE TABLE PRODUCT_DETAIL (  
    PRODUCT_ID CHAR(20) NOT NULL REFERENCES PRODUCT(PRODUCT_ID),  
    LOCALE_CD CHAR(5) NOT NULL,  
    PRODUCT_COLOR CHAR(20) character set UTF8,  
    PRODUCT_DESC CHAR(255) character set UTF8);  

ALTER TABLE PRODUCT_DETAIL ADD PRIMARY KEY (  
    PRODUCT_ID, LOCALE_CD);  

INSERT INTO PRODUCT VALUES (  
    'ball_33', '337718N');  

INSERT INTO PRODUCT_DETAIL VALUES (  
    'ball_33', 'en_US', 'red', 'paddle ball replacement');  
INSERT INTO PRODUCT_DETAIL VALUES (  
    'ball_33', 'es_ES', 'rojo', 'sustitución de paddle');  
INSERT INTO PRODUCT_DETAIL VALUES (  
    'ball_33', 'fr_FR', 'rouge', 'paddle ball de remplacement');  

SELECT P.PRODUCT_ID, PRODUCT_COLOR, PRODUCT_DESC  
FROM PRODUCT P  
INNER JOIN PRODUCT_DETAIL PD  
    ON P.PRODUCT_ID=PD.PRODUCT_ID  
WHERE PRODUCT_ID=? ;  

After you register the PRODUCT table and each PRODUCT_DETAIL table in metadata, use the sas-localize-table tool to associate the localized table in metadata. For example, if the path to the table is '/Shared Data/Product Data', then use the following sas-localize-table commands to create the detail table and associate it with the PRODUCT table in metadata:  

sas-localize-table <connect-options> '/Shared Data/Product Data/Product_Detail'  
    -key PRODUCT_ID -locale LOCALE_CD  
  
sas-localize-table <connect-options> '/Shared Data/Product Data/Product'  
    -localize PRODUCT_ID='/Shared Data/Product Data/Product Detail'  

Use the following command to view the localization that is applied to the PRODUCT table:  

sas-localize-table <connect-options> '/Shared Data/Product Data/PRODUCT'
Assigning Libraries

What Does It Mean to Assign a Library? ................................................................. 31
How Do the Different Platform Clients Assign Libraries? ............................... 32
Understanding Pre-Assigned Libraries .............................................................. 33
Understanding Native Engines and the Metadata LIBNAME Engine .................. 34
Considerations for SAS Stored Process and SAS Pooled Workspace Servers ....... 36
Pre-assigning a Library ........................................................................ 38
Overview of Pre-assigning a Library ................................................................. 38
Stage 1: Flag the Library as Pre-assigned ......................................................... 38
Stage 2: Edit the Configuration Files .............................................................. 40
Pre-assigning Libraries to Use the Metadata Engine ........................................ 41
Pre-assigning Libraries in an Autoexec File ..................................................... 41
SAS Application Server Autoexec Files .......................................................... 42
Verifying Pre-assignments By Reviewing the Logs .......................................... 43
Limiting Access to Selected Libraries ............................................................ 43

What Does It Mean to Assign a Library?

In “Overview of Connecting to Common Data Sources” on page 48, you learn how to register libraries in metadata and assigned the libraries to SAS servers. These libraries represent data such as SAS data sets in a directory or the tables in a database schema. By assigning a library to a SAS server, you accomplish the following:

- enable the SAS server to access the library
- make the library visible to users of the SAS server
- control which SAS engine is used by the SAS server to access data, if you pre-assign the library

Assigning a library to a SAS server means letting the SAS server session know that a libref—a shortcut name—is associated with the information that a SAS session needs to access a data library. SAS Intelligence Platform clients such as SAS Data Integration Studio, SAS OLAP Cube Studio, and SAS Information Map Studio generate SAS code that uses librefs. Before the generated code can access data from a library, the library must be assigned to the SAS server that executes the code. After the SAS server determines that a library is assigned to it, the next step is to determine how the SAS server should access the data in the library.
There are two ways in which a server can determine how to access data in a library. One way is for you, as the administrator, to configure the environment so that the SAS server finds out about the libref and the SAS engine to use for data access at server start-up. This approach is referred to as pre-assigning the library, because the libref is established before any code that uses that libref is submitted. The other way is to let the client application define the libref and set the SAS engine when it generates code for submission to that server.

How Do the Different Platform Clients Assign Libraries?

By default, newly created libraries are not pre-assigned. When a library is not pre-assigned, the library is accessed using the SAS engine that is most appropriate for the client application and its intended user base. For example, if you do not pre-assign the library, SAS Data Integration Studio creates a libref that uses the native engine that is specified in metadata, such as BASE. This approach is a best practice, because it is assumed that in most cases SAS Data Integration Studio developers are building processes that create or update tables. In this case, the native engine is the only engine that should be used for data-populating tasks.

The following table shows which SAS engine is used by many of the platform clients for libraries that are not pre-assigned.

<table>
<thead>
<tr>
<th>Application</th>
<th>Pre-assigned</th>
<th>Library Engine Used</th>
<th>Minimum Metadata Authorizations Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAS Add-In for Microsoft Office</td>
<td>No</td>
<td>META</td>
<td>Library: ReadMetadata</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Table: ReadMetadata and Read</td>
</tr>
<tr>
<td>SAS Enterprise Guide</td>
<td>No</td>
<td>META</td>
<td>Library: ReadMetadata</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Table: ReadMetadata and Read</td>
</tr>
<tr>
<td>SAS Data Integration Studio</td>
<td>No</td>
<td>native engine</td>
<td>Library: ReadMetadata</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Table: ReadMetadata</td>
</tr>
<tr>
<td>SAS OLAP Cube Studio</td>
<td>No</td>
<td>native engine</td>
<td>Library: ReadMetadata</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Table: ReadMetadata</td>
</tr>
<tr>
<td>SAS Information Map Studio</td>
<td>No</td>
<td>native engine</td>
<td>Library: ReadMetadata</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Table: ReadMetadata</td>
</tr>
</tbody>
</table>
When libraries are not pre-assigned, each SAS platform client accesses data with the SAS engine that makes the most sense for the client. Allowing each client to choose the SAS engine that it deems appropriate for its user base results in a security model that might match data access requirements. The clients that are typically used for data building use the native engine. The clients that are typically used for queries and reporting are designed to use the metadata engine. An example of such an environment is one with clients running at least SAS Enterprise Guide and SAS Data Integration Studio. In this environment, SAS Data Integration Studio processes update tables that are in turn used in ad hoc analysis within SAS Enterprise Guide. The SAS Data Integration Studio processes need to specify tables in the library as target tables (output), whereas the SAS Enterprise Guide user's activities largely involve querying and analyzing chunks of data (input).

Because SAS Data Integration Studio processes typically update or create target tables, it is designed to use the native engine instead of the metadata engine. It accesses the tables using the engine that is specified in metadata for the library. Because SAS Data Integration Studio works with tables that are registered in the metadata repository, you can control access to tables by granting ReadMetadata, WriteMetadata, and CheckInMetadata permissions on the library and table metadata objects.

**Note:** The metadata authorization layer supplements operating system and RDBMS security. It does not replace it. Operating system and RDBMS authorization should always be used as the first means of securing access to tables.

On the other hand, the SAS Add-In for Microsoft Office and SAS Enterprise Guide use the metadata engine by default. For these clients, the data-level authorizations of Read, Write, Create, and Delete that are specified in metadata, are enforced.

If defining libraries so that they are not pre-assigned seems like a potential option for your environment, then you should also learn how to ensure that these libraries are available to server processes that do not receive direct requests from client applications. For example, you need to know how to assign the library in server processes such as the stored process server and DATA Step Batch Server (if present). For more information, see “Considerations for SAS Stored Process and SAS Pooled Workspace Servers” on page 36.

---

**Understanding Pre-Assigned Libraries**

Deciding whether to pre-assign a library or not has important consequences. One factor to keep in mind is that pre-assigning an excessive number of libraries can slow the execution of SAS jobs for all users. Other factors are described in “Understanding Native Engines and the Metadata LIBNAME Engine” on page 34. SAS clients and stored processes can access a library using one of two engines:
the native engine that is specified in the library's metadata or a file such as an autoexec file. In this case, the Base SAS engine is used for libraries of SAS data sets, the ORACLE engine is used for Oracle libraries, and so on.

the Metadata LIBNAME Engine.

Which engine you use affects security and determines what operations are possible.

Note: If you are defining a pre-assigned DBMS library, do not use the Pre-Assigned Library resource template. Register the library using the appropriate DBMS library template.

If you pre-assign libraries, then you control which engine is used to access the data.

If you choose to pre-assign libraries with LIBNAME statements in an autoexec file, then you should still register the library in metadata. Registering the library in metadata makes it available for use by SAS clients that read from metadata, such as SAS Data Integration Studio.

Having the server process assign libraries upon start-up results in library assignments that are identical and guaranteed across all SAS client applications and servers. Some environments where this approach to assigning libraries is desirable include the following:

- environments where users are executing stored processes, and you do not want programmers having to manage library assignments in their code or in autoexec files.
- environments where the DATA Step Batch Server is used to execute jobs created by SAS Data Integration Studio, and library assignments for these jobs should be identical to assignments used when the process was created.
- environments where SAS Enterprise Guide or SAS Add-In for Microsoft Office users are running tasks that need to create tables in the library that is registered in metadata. When you register a client-assigned library (a library that is not pre-assigned), SAS Enterprise Guide and SAS Add-In for Microsoft Office assign the library to use the metadata engine by default. Metadata engine libraries do not update metadata after changes to the underlying data source. Metadata can be updated to reflect changes to the underlying data source with PROC METALIB or with the register tables function of SAS Management Console.

When libraries are assigned by the client application, each application can assign the library in a way that is most suitable for its intended user base, and library connections are established only if needed. When libraries are assigned by the server, each library is available to all server processes and each library is allocated the same way for all client applications. A mixture of some server-assigned and some client application-assigned libraries is typically required to meet the needs of all the users in your environment.

Understanding Native Engines and the Metadata LIBNAME Engine

As mentioned previously, when you access the data in a data library, you can use the native engine stored in the metadata definition of the library, the definition in a configuration file, or you can use the metadata engine. As shown in the following
figure, the metadata engine invokes the Base SAS engine that is identified in the metadata.

*Figure 3.1* Metadata Engine Invocation of the Base SAS Engine

SAS metadata provides a metadata authorization layer that enables you to control which users can access which metadata objects, such as SASLibrary, PhysicalTable, and LogicalServer. You manage the access to metadata by granting ReadMetadata and WriteMetadata permissions on the object or on the repository.

As depicted in the previous figure, when a user accesses a library that uses the metadata engine, the engine sends a request to the SAS Metadata Server to determine the user’s metadata permissions on the tables in the library. The tables for which the user has ReadMetadata access are returned and are available to the user. If the user then performs an action, such as opening a table, the metadata engine queries the metadata server for the user’s metadata permission on the table. If the user has Read access to the table, directly or through group membership, the metadata engine then calls upon the engine specified in metadata to handle the request. The table is then opened into the client application for reading.

In contrast, for clients such as SAS Data Integration Studio that surface only the tables that are registered in metadata and accessible to the user, if a user accesses a library that uses a native engine, the client application contacts the metadata server and requests access to the metadata object as the user. The metadata server then queries the metadata authorization layer to determine whether the user has ReadMetadata access on the tables in the library. In contrast to the metadata engine, there is no query to check for Read access on the table. When a library uses a native engine, the data-level authorizations of Read, Write, Create, and Delete are not checked.

If you want to use the metadata authorization layer to control Read, Write, Create, and Delete permissions, then pre-assign the library. Follow the steps described in
When you use the Pre-assignment Type menu, select **By metadata library engine**.

One drawback to using the metadata engine is that it does not automatically create, update, or delete metadata when changes are made to the tables.

---

**Considerations for SAS Stored Process and SAS Pooled Workspace Servers**

In the SAS Intelligence Platform, a stored process is a SAS program that is stored on a server and can be executed as requested by users that have ReadMetadata access to the stored process program's metadata. If a library is not pre-assigned, it is the responsibility of the stored process program's author or the SAS administrator to ensure that the library is assigned. Assigning the library can be done with SAS code that is written in the stored process program or that is linked to the stored process with an %INCLUDE statement from an external file.

It is important to understand how SAS Stored Process Servers and SAS Pooled Workspace Servers access data in libraries that are pre-assigned or not pre-assigned, and how they use the metadata engine or a native engine. The following table describes how these interactions affect data access.

**Table 3.2  SAS Engine Assignment Interactions for SAS Stored Process and SAS Pooled Workspace Servers**

<table>
<thead>
<tr>
<th>Metadata Engine</th>
<th>Native Engine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Library is pre-assigned</td>
<td>Library is pre-assigned</td>
</tr>
<tr>
<td>The library is assigned to use the metadata engine at server initialization by the SAS Spawned Servers account (sassrv).</td>
<td>The library is assigned to use the native engine for the library type at server initialization by the SAS Spawned Servers account (sassrv).</td>
</tr>
<tr>
<td>Metadata authorization layer permissions for the SAS Spawned Servers account are checked at server initialization and for each data access request.</td>
<td>Metadata authorization layer permissions are checked for the SAS Spawned Servers account. The SAS Spawned Servers account must have ReadMetadata access to the library. (The access is typically inherited through a grant to the SAS General Servers group.) The metadata authorization layer permissions for the user ID that uses the server are not used.</td>
</tr>
<tr>
<td>Metadata authorization layer permissions for the user ID that uses the server are not used.</td>
<td>These two items apply to pre-assigned libraries, regardless of the SAS engine that is used for data access.</td>
</tr>
</tbody>
</table>

- If the Base SAS engine is used, then the SAS Spawned Servers account (sassrv) operating system permissions for the path to the physical library are checked at server initialization.
- If a SAS/ACCESS engine is used, then database credentials must be stored in metadata. The SAS Spawned Servers account must access the credentials stored in metadata at server initialization.
The library is not assigned at server initialization. It is assigned to use the metadata engine by the SAS code that is submitted to the server.

The metadata authorization layer and data-level permissions are checked for the user ID that submits the code.

The SAS Spawned Servers account metadata authorization layer permissions are not used.

The library is not assigned at server initialization. It is assigned to use a native engine by the SAS code that is submitted to the server.

Metadata authorization layer permissions are not used. They are never checked for the SAS Spawned Servers account or the user ID that uses the server.

* If you change metadata permissions on a pre-assigned library or its tables, you must restart the SAS Object Spawner. This is needed so that the SAS server initializes again.

**TIP** Regardless of the engine used and whether the library is pre-assigned or not pre-assigned, if the library is a Base SAS library, then the SAS Spawned Servers account must have operating system permissions for the path to the physical library. The operating system permissions for the user ID that submitted the code are not used.

The following are errors that might be written to the SAS log.

**ERROR: Libname libref is not assigned.**

If the library is not pre-assigned, then check the syntax for your LIBNAME statement. If the library is pre-assigned to use the metadata engine, then make sure that the SAS General Servers identity has ReadMetadata permission to the library. If the library is pre-assigned to use a native engine, then make sure that the user ID accessing the data has ReadMetadata permission to the library.

**ERROR: The Read permission is required to read data in table/view libref.table.DATA.**

This message is generated for libraries that use the metadata engine only. If the table belongs to a pre-assigned library, then grant Read permission to the SAS General Servers identity on the table. If the table does not belong to a pre-assigned library, then grant Read permission to the user identity on the table.

**ERROR: File libref.table.DATA does not exist.**

**ERROR: The metadata for the table/view libref.table does not exist.**

This message is generated for libraries that use the metadata engine only and means that the table is not registered in metadata. If the library is pre-assigned to use the metadata engine, then disable that setting temporarily while you register the table. If you do not disable the pre-assignment, then be aware that the library is assigned to use the metadata engine with the METAOUT=ALL option. This METAOUT option does not permit registering tables.
Pre-assigning a Library

Overview of Pre-assigning a Library

Pre-assigning a large number of libraries can have a negative impact on the execution time of SAS programs for all users. Therefore, be judicious in deciding whether to pre-assign a library.

Pre-assigning a library can be a two-stage process:

1. Flag the library as pre-assigned.
2. Edit the configuration files.

Note: Keep in mind that pre-assigning the library does not mean that you do not need to register tables in metadata. You must register the tables so that applications that read tables from metadata, such as SAS Data Integration Studio, can work with the tables.

Stage 1: Flag the Library as Pre-assigned

A library can be configured to be assigned by the server process by either selecting the Library is pre-assigned advanced option when the library is being registered or by modifying the library’s properties after the fact. To pre-assign a library, perform the following steps:

1. From SAS Management Console, select Data Library Manager ➔ Libraries ➔ library-name ➔ Properties.
2. Select the Options tab.
3. Click the Advanced Options button.
4. Select the Library is Pre-Assigned check box on the Pre-Assign tab, and select an option from the Pre-Assignment Type menu.
Figure 3.2  Library Is Pre-assigned Option

**TIP** Select **By external configuration** when you want to pre-assign the library in a configuration file, such as an autoexec.

5. Click **OK** on the Advanced Options dialog box.
6. Click the **Assign** tab on the library properties window.
7. Select the servers that you want to pre-assign the libraries to. Click **OK** when you are finished.
Stage 2: Edit the Configuration Files

In the previous stage, the library was flagged as pre-assigned. You also selected which servers can retrieve the library definitions from metadata. In this stage, you edit the `sasv9_usermods.cfg` file for the same servers that were selected in the previous stage and add a `METAAUTORESOURCES` SAS system option. This option is used so that the servers read the library definitions from metadata as they start.

**Note:** This stage is not needed for workspace servers, pooled workspace servers, stored process servers, SAS/SHARE servers, or OLAP servers. Those server types automatically read metadata when they start and assign the libraries.

To edit the configuration files, perform the following steps:

1. For each SAS/CONNECT server, edit the following file:
   
   ```
   SAS-configuration-directory\Lev1\SASApp\ConnectServer\sasv9_usermods.cfg
   ```
   
   Add the following SAS system option:
   
   ```
   -metaautoresources 'SASApp'
   ```

2. For each DATA Step Batch server, edit the following file:
   
   ```
   SAS-configuration-directory\Lev1\SASApp\BatchServer\sasv9_usermods.cfg
   ```
   
   Add the following SAS system option:
   
   ```
   -metaautoresources 'SASApp'
   ```
Pre-assigning Libraries to Use the Metadata Engine

If You Do Not Need to Create or Delete Tables

Pre-assigning a library to use the metadata engine is available by selecting the **By metadata library engine** option for a library on the Advanced Options dialog box. Using this option results in using the metadata engine with the METAOUT=ALL option. This LIBNAME option specifies that you can read, create, update, and delete observations in physical tables that already exist and are registered in metadata. You cannot create or delete entire physical tables.

If You Need to Create or Delete Tables

Other METAOUT= LIBNAME options exist. For example, the METAOUT=DATA option permits reading, creating, updating, and deleting physical tables. The METAOUT=DATAREG setting permits reading, updating, and deleting tables registered in metadata as well as creating new physical tables. Be aware that new physical tables cannot be read until they are registered in metadata. For more information, see the “METAOUT= Argument” in *SAS Language Interfaces to Metadata*.

If the METAOUT=ALL setting does not meet your business needs, but the data-level authorizations do meet your business needs, you can pre-assign the library to use the metadata engine in a configuration file. To do so, perform the following high-level steps:

1. Follow all the steps in “Stage 1: Flag the Library as Pre-assigned”. When you set the pre-assignment type in step 4, select **By external configuration**.
   
   **Note:** Do not choose **By metadata library engine**. In order to use a different METAOUT= option, you must specify the LIBNAME statement in an external configuration file.

2. Construct a LIBNAME statement that uses the metadata engine and your preferred METAOUT= option. For an example, see **Example Code 3.1**.

3. Determine which autoexec file to use for the LIBNAME statement. For more information, see “SAS Application Server Autoexec Files” on page 42.

The metadata engine can reference a library by the library name. For a library that is registered in metadata with a name of Accounting tables and a libref of lib1, the following LIBNAME statement accesses the tables with the metadata engine:

**Example Code 3.1 Metadata LIBNAME Engine Statement**

```
libname lib1 meta library="Accounting tables" metaout=datareg;
```

Pre-assigning Libraries in an Autoexec File

Pre-assigning libraries in an autoexec file is not a recommended as a routine practice because library assignments are recorded in two places, the autoexec file and metadata. Having configuration information in two places increases
maintenance. An autoexec file is a text file that contains SAS statements that are executed when the server process starts. If an autoexec file is used in your environment, it is important to note that the LIBNAME statements in the autoexec file take precedence over same-named libraries assigned by the server in metadata. For example, if ORGOLD is registered in the metadata to be pre-assigned, and ORGOLD is also defined in an autoexec for the same server, the ORGOLD library is assigned using the LIBNAME information from the autoexec file. Simply put, the library assignment in the autoexec file always takes precedence.

To pre-assign a library in an autoexec file, perform the following high-level steps:

1. Follow all the steps in “Stage 1: Flag the Library as Pre-assigned”. When you set the pre-assignment type in step 4, select By external configuration.

2. Construct a LIBNAME statement.

3. Determine which autoexec file to use for the LIBNAME statement. For more information, see “SAS Application Server Autoexec Files” on page 42.

The following display shows LIBNAME statements that pre-assign libraries in an autoexec file. The comments in the display acknowledge how pre-assigning a library in an autoexec file and registering the same library in metadata can cause unexpected results when accessing data.

Figure 3.4 Library Assignment in an Autoexec File

```sas
/* Notice this LIBNAME maps ORGOLD to a different location */
/* than the location in the metadata. This can cause unexpected */
/* results and definite failures in metadata-dependent applications. */
LIBNAME ORGOLD "C:\Oracle\NotGold\access=readonly;
LIBNAME TABLES "C:\SAS\Lev\SASApp\Data\access=readonly;"
LIBNAME MISC "C:\SAS\Config\Lev\SASApp\Data";
LIBNAME GLIMMACRO
  "C:\SAS\Config\Lev\SASApp\SASEnvironment\SASMacro\GoldProgrammers"
  access=readonly;
OPTION imtsearch=(MISC, MISC, MISC) nostorerr;
OPTIONS NOSTORE GLMSTORE=GLIMMACRO;
/* initialize some macros */
%let dmemptions=;
%let dbitempnames=;
%let dbitemps=;
```

SAS Application Server Autoexec Files

During the configuration process, the SAS Deployment Wizard created a single file named `appserver_autoexec_usermods.sas` that controls all component servers of the SAS Application Server and files named `autoexec_usermods.sas` for each of the component servers of the application server:

- `appserver_autoexec_usermods.sas`

  Use this file if you want all the SAS Application Server components registered to the application server, such as an OLAP server, a workspace server, and so on, to access the library with the same SAS engine.

- `autoexec_usermods.sas`

  Use this file to modify one of the SAS Application Server components, such as the workspace server, to use the engine specified in the LIBNAME statement,
but to leave the other server components unchanged. If this is your choice, then note that the autoexec_usermods.sas file is located within a subdirectory, such as C:sas\Config\Lev1\SASApp\WorkspaceServer.

Verifying Pre-assignments By Reviewing the Logs

After you specify that a library is to be pre-assigned by the server, the SAS server process starts as follows:

1. Connect to the metadata server.
2. Retrieve library metadata.
3. Assign the library using the engine specified in the library metadata.

For example, if the Orion Gold Customers library is pre-assigned, then the library assignment is equivalent to a SAS programmer submitting a LIBNAME statement such as the following:

LIBNAME ORGOLD BASE "D:\OrionStar\Gold";

In the case of an IOM server, such as the workspace server, you can verify the pre-assignment of this library by the server process by enabling logging and observing the note generated from the first GetMetadata method call in the server's log, as in the following sample log:

NOTE: Libref ORGOLD successfully assigned from logical server.

For non-IOM servers using the METAUTORESOURCES option, a note like the following is written to its log file:

NOTE: Libref ORGOLD successfully assigned from logical server.

For information about enabling the logging, see SAS Logging: Configuration and Programming Reference.

Limiting Access to Selected Libraries

In “Overview of Connecting to Common Data Sources” on page 48, you learned how to register libraries in metadata and assigned the libraries to SAS servers. If you are running in a client/server environment (for example, you use SAS Enterprise Guide), you can create an environment where your SAS client has access to a
specific set of directories and files. All other directories and files are inaccessible. In order to limit access to certain libraries and files, you can limit the scope and activities of a SAS server by putting it in a locked-down state.

The locked-down SAS server can access only specified directories and files. For example, a user who connects to a locked-down workspace server from SAS Enterprise Guide can access only those directories and files that are included in that server’s list of permitted resources. Regardless of host-layer permissions, LIBNAME and FILENAME statements that users submit through a locked-down server are rejected, unless the target resource is included in the server’s lockdown paths list.

The following types of servers can be locked down:

- workspace (standard)
- pooled workspace (server-side pooling)
- stored process
- batch
- grid
- SAS/CONNECT

For more information about locked-down servers, see the SAS Intelligence Platform: Security Administration Guide.
## Connecting to Common Data Sources

- **Overview of Connecting to Common Data Sources**: 48
- **Overview of Connections to Relational Databases**: 48
  - Process Overview: 48
  - Define the DBMS User: 49
  - Define the DBMS User Group: 50
  - Register the DBMS Server: 50
  - Register the DBMS Library: 51
- **Using the Folders Tab in SAS Management Console**: 52
- **Establishing Connectivity to a Library of SAS Data Sets**: 52
  - Overview of Establishing Connectivity to a Library of SAS Data Sets: 52
  - Register the Base SAS Library Using SAS Management Console: 53
  - Register the Base SAS Library Using SAS Environment Manager: 54
  - 2.5 Administration: 54
  - Working with User-Defined Formats: 55
- **Establishing Shared Access to SAS Data Sets**: 57
  - Overview of Establishing Shared Access: 57
  - Create a SAS/SHARE Remote Engine Library Using SAS Management Console: 58
  - Create a SAS/SHARE Remote Engine Library Using SAS Environment Manager: 59
- **Establishing Connectivity to a Microsoft Excel File**: 60
  - Overview of Establishing Connectivity to an Excel File: 60
- **Establishing Connectivity to a Flat File**: 61
- **Establishing Connectivity to XML Data**: 64
- **Establishing Connectivity to a SAS Information Map**: 65
  - Overview of Establishing Connectivity to a SAS Information Map: 65
  - Special Considerations for Information Map Tables: 67
- **Establishing Connectivity to a Cloud Analytic Services (CAS) Server**: 68
  - Overview of Establishing Connectivity to CAS: 68
  - Stage 1: Register the CAS Server: 69
  - Stage 2: Register the Cloud Analytic Services (CAS) Library: 70
- **Establishing Connectivity to Several Different SAS/ACCESS Servers**: 71
  - Overview of Establishing Connectivity to a Greenthumb, Hawq, Netessa, Sybase IQ, Postgres, or Vertica Server: 71
  - Register the DBMS Server: 72
### Establishing Connectivity to an ODBC Database Using Microsoft Windows NT Authentication

Stage 2: Register the ODBC Library

Overview of Establishing Connectivity to an ODBC Database Using Microsoft Windows NT Authentication

Stage 2: Register the ODBC Server Library

Stage 2: Register the DBMS Server

Stage 1: Configure an ODBC Data Source Using Microsoft Windows NT Authentication

Stage 2: Create a Library By Submitting a LIBNAME Statement with the ODBC Engine

Stage 3: Register a User

Stage 4: Register the ODBC Server

Stage 5: Register the ODBC Library

### Establishing Connectivity to an Aster Server

Overview of Establishing Connectivity to an Aster Server

Stage 2: Register the Aster Server Library

Stage 2: Register the ODBC Server

Stage 2: Register the Aster Server Library

### Establishing Connectivity to a SAS Federation Server

Overview of Establishing Connectivity to a SAS Federation Server

Stage 1: Register the SAS Federation Server

Stage 2: Register the SAS Federation Server Library

Stage 2: Register the SAS Federation Server Library

Special Considerations for SAS Federation Server Libraries

### Establishing Connectivity to Hadoop

Overview of Establishing Connectivity to Hadoop

Stage 1: Register the Hadoop Server

Stage 2: Register the Hadoop via Hive Library

Stage 2: Register the SAS/ACCESS Server Library

Stage 2: Create a Library By Submitting a LIBNAME Statement

Stage 2: Register the Hadoop via Hive Library

Special Considerations for Hadoop via Hive Tables

### Establishing Connectivity to a SAS LASR Analytic Server

Overview of Establishing Connectivity to a SAS LASR Analytic Server

Use SAS Environment Manager 2.6 Administration to Establish Connectivity to SAS LASR Analytic Server

Use SAS Environment Manager 2.5 Administration to Establish Connectivity to SAS LASR Analytic Server

Use SAS Management Console to Establish Connectivity to SAS

### Establishing Connectivity to a SAS Federation Server

Overview of Establishing Connectivity to a SAS Federation Server

Stage 2: Register the SAS Federation Server Library

### Establishing Connectivity to a MySQL Server

Overview of Establishing Connectivity to a MySQL Server

Register the DBMS Server

Stage 2: Register the MySQL Server Library

### Establishing Connectivity to an ODBC Database Using Microsoft Windows NT Authentication

Overview of Establishing Connectivity to an ODBC Database

Using Microsoft Windows NT Authentication

Prerequisites

Stage 1: Configure an ODBC Data Source Using Microsoft Windows NT Authentication

Stage 2: Create a Library By Submitting a LIBNAME Statement with the ODBC Engine

Stage 3: Register a User

Stage 4: Register the ODBC Server

Stage 5: Register the ODBC Library

### Establishing Connectivity to a Microsoft Access Database By Using ODBC

Overview of Establishing Connectivity to a Microsoft Access Database By Using ODBC

Stage 1: Define the ODBC Data Source

Stage 2: Register the ODBC Server

Stage 3: Register the ODBC Library
<table>
<thead>
<tr>
<th>Content</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Establishing Connectivity to an OLE DB Database Using</strong></td>
<td></td>
</tr>
<tr>
<td>Microsoft Windows NT Authentication</td>
<td>111</td>
</tr>
<tr>
<td>Overview of Establishing Connectivity to an OLE DB Database</td>
<td></td>
</tr>
<tr>
<td>Using Microsoft Windows NT Authentication</td>
<td>111</td>
</tr>
<tr>
<td>Prerequisites</td>
<td>111</td>
</tr>
<tr>
<td>Stage 1: Register the OLE DB Server</td>
<td>112</td>
</tr>
<tr>
<td>Stage 2: Register the OLE DB Library</td>
<td>114</td>
</tr>
<tr>
<td>Stage 3: Verify That the SQL Server Tables Are Listed in the SAS Explorer</td>
<td>116</td>
</tr>
<tr>
<td>Stage 4: Register the SQL Server Tables</td>
<td>116</td>
</tr>
<tr>
<td><strong>Establishing Connectivity to an Oracle Database</strong></td>
<td>117</td>
</tr>
<tr>
<td>Overview of Establishing Connectivity to an Oracle Database</td>
<td></td>
</tr>
<tr>
<td>Prerequisites</td>
<td>117</td>
</tr>
<tr>
<td>Stage 1: Register the Oracle Server</td>
<td>118</td>
</tr>
<tr>
<td>Stage 2: Register the Oracle Library</td>
<td>119</td>
</tr>
<tr>
<td><strong>Establishing Connectivity to a PI System Server</strong></td>
<td>120</td>
</tr>
<tr>
<td>Overview of Establishing Connectivity to a PI System Server</td>
<td></td>
</tr>
<tr>
<td>Register the DBMS Server</td>
<td>121</td>
</tr>
<tr>
<td>Stage 2: Register the PI System Server Library</td>
<td>122</td>
</tr>
<tr>
<td><strong>Establishing Connectivity to a Redshift Server</strong></td>
<td>123</td>
</tr>
<tr>
<td>Overview of Establishing Connectivity to Redshift</td>
<td></td>
</tr>
<tr>
<td>Stage 1: Register the Redshift Server</td>
<td>124</td>
</tr>
<tr>
<td>Stage 2: Register the Redshift Library</td>
<td>126</td>
</tr>
<tr>
<td><strong>Establishing Connectivity to an SAP Server</strong></td>
<td>126</td>
</tr>
<tr>
<td>Overview to Establishing Connectivity to an SAP Server</td>
<td></td>
</tr>
<tr>
<td>Stage 1: Register the SAP Server</td>
<td>127</td>
</tr>
<tr>
<td>Stage 2: Register the SAP Library</td>
<td>129</td>
</tr>
<tr>
<td>Stage 3: Extract SAP Metadata</td>
<td>129</td>
</tr>
<tr>
<td>Special Considerations for SAP</td>
<td>130</td>
</tr>
<tr>
<td><strong>Establishing Connectivity to an SAP HANA Server</strong></td>
<td>130</td>
</tr>
<tr>
<td>Overview to Establishing Connectivity to an SAP HANA Server</td>
<td></td>
</tr>
<tr>
<td>Stage 1: Register the SAP HANA Server</td>
<td>132</td>
</tr>
<tr>
<td>Stage 2: Register the SAP HANA Library</td>
<td>133</td>
</tr>
<tr>
<td><strong>Establishing Connectivity to a SAS Scalable Performance Data Server</strong></td>
<td>134</td>
</tr>
<tr>
<td>Overview of Establishing Connectivity to a SAS Scalable Performance Data Server</td>
<td></td>
</tr>
<tr>
<td>Stage 1: Configure the libnames.parm File</td>
<td>134</td>
</tr>
<tr>
<td>Stage 2: Register the SAS Scalable Performance Data Server</td>
<td>135</td>
</tr>
<tr>
<td>Stage 3: Register the SAS Scalable Performance Data Server Library</td>
<td>137</td>
</tr>
<tr>
<td><strong>Establishing Connectivity to a SASHDAT Library</strong></td>
<td>138</td>
</tr>
<tr>
<td>Overview of Establishing Connectivity to a SASHDAT Library</td>
<td></td>
</tr>
<tr>
<td>Register a SASHDAT Library</td>
<td>138</td>
</tr>
<tr>
<td><strong>Establishing Connectivity to a Teradata Database</strong></td>
<td>141</td>
</tr>
<tr>
<td>Overview of Establishing Connectivity to a Teradata Server</td>
<td></td>
</tr>
<tr>
<td>Stage 1: Register the Teradata Server</td>
<td>141</td>
</tr>
<tr>
<td>Stage 2: Register the Teradata Library</td>
<td>143</td>
</tr>
<tr>
<td><strong>Registering and Verifying Tables</strong></td>
<td>144</td>
</tr>
<tr>
<td>Overview to Registering Tables</td>
<td></td>
</tr>
<tr>
<td>Stage 1: Register the Tables</td>
<td>144</td>
</tr>
<tr>
<td>Stage 2: Verify Access to the Data in a SAS Application</td>
<td>145</td>
</tr>
<tr>
<td><strong>Read-only Access for Reporting Libraries</strong></td>
<td>146</td>
</tr>
</tbody>
</table>
Overview of Connecting to Common Data Sources

This chapter consists of detailed examples for establishing a connection to each of the common data sources introduced in Chapter 1, “Overview of Common Data Sources,” on page 1. Some of the connection processes covered in this chapter have common elements that might be applied to similar data sources. For example, the description of the process of using SAS/ACCESS to connect to an Oracle database might be useful when you connect to other relational databases such as DB2, Sybase, and Informix. Also, the description of ODBC connections to a Microsoft Access database and the account of the connection to an SAP source can be helpful when you connect to similar data sources.

In order to perform the procedures for registering libraries, you must have ReadMetadata and WriteMetadata permission for the repository and the SAS Application Servers to which the data library is assigned. You must also have WriteMemberMetadata permission for the folder into which the library and its tables are registered.

This chapter also explains the process that registers tables as metadata from the data sources. Registering a table in metadata enables you to view the data in a SAS application. For more information about managing table metadata, see Chapter 2, “Managing Table Metadata,” on page 17.

Overview of Connections to Relational Databases

Process Overview

This section provides generic instructions for using SAS Management Console to configure access to a database. SAS/ACCESS must be licensed and configured before using SAS Management Console to register the library that holds the tables. The generic task has three stages:

1. Define the DBMS user or user group. Defining users and user groups is the same for all databases.

2. Register the DBMS server. These steps can vary per SAS/ACCESS interface. The SAS/ACCESS interfaces that vary have their own section.
3 Register the DBMS library. These steps can vary per SAS/ACCESS interface. The SAS/ACCESS interfaces that vary have their own section.

Define the DBMS User

User definitions can be set up using SAS Management Console or SAS Environment Manager.

The User Manager is a SAS Management Console plug-in that provides functions to manage metadata definitions for users and user groups. Each user and user group that accesses secure resources on a SAS Metadata Server must be represented by an identity on the server. Individual users are represented by Person objects, and groups are represented by Identity Group objects.

To define a user using SAS Management Console, perform the following steps:

1 Start SAS Management Console using your administrator account.

2 From SAS Management Console navigation tree, right-click the User Manager icon. Then select New ⇒ User, which displays the New User Properties dialog box.

3 On the General tab, enter your name in the Name field. Note that you can enter any user name that you want to use (for example, a single name, your full name, and so on). Display Name, Job Title, and Description are optional.

4 Click the Accounts tab and click New to display the New Login Properties dialog box.

5 Enter your Windows domain and user ID (WindowsDomain\user) and the password.

6 Select DefaultAuth as the authentication domain.

7 Click OK to add the login to the user.

8 Click OK to close the New User Properties dialog box.

9 Click User Manager to verify that the name appears in the user name list.

The SAS Environment Manager Users module supports some of the user administration tasks that are provided by the User Manager plug-in to SAS Management Console, including the following:

- creation and maintenance of users, groups, and roles
- management of group and role memberships
- management of logins and internal accounts

For more information about the SAS Environment Manager Users module, see “Managing User Access” in SAS® Environment Manager User’s Guide.
Define the DBMS User Group

User group definitions can be set up using SAS Management Console or SAS Environment Manager.

To define a user group using SAS Management Console, perform the following steps:

1. Start SAS Management Console using your administrator account.
2. From SAS Management Console navigation tree, right-click the User Manager icon. Then select New Group, which displays the New User Properties dialog box.
3. Click the General tab, enter the group name in the Name field. Note that you can enter any group name that you want to use. Display Name and Description are optional.
4. Click the Groups and Roles tab and add the appropriate users and groups.
5. Click the Accounts tab and click New to display New Login Properties dialog box.
6. Enter your domain and user ID (WindowsDomain\user) and the password.
7. Select DefaultAuth as the authentication domain. Click OK.
8. Click the Members tab and add all of the users that need to access this library. To provide access for all registered users, give logon access to the SASUSERS group by moving it from the Available Identities box to the Current Members box. Click OK.
9. Click User Manager to verify that the name appears in the user group name list.

The SAS Environment Manager Users module supports some of the user administration tasks that are provided by the User Manager plug-in to SAS Management Console, including the following:

- creation and maintenance of users, groups, and roles
- management of group and role memberships
- management of logins and internal accounts

For more information about the SAS Environment Manager Users module, see “Managing User Access” in SAS® Environment Manager User’s Guide.

Register the DBMS Server

To register a DBMS server, perform the following steps:

1. Right-click Server Manager and select the New Server option to access the New Server wizard.
2. Select the database server type from the Database Servers list. Click Next.
3 Enter an appropriate server name in the **Name** field. The **Description** is optional. Click **Next**.

4 Accept the defaults for the server properties. Click **Next**.

5 Specify the database vendor-specific values on the connection properties page. If the user credentials for the database are different from the credentials used to log on to SAS, then you must create an Authentication domain to store valid database credentials. For more information, see “How to Store Passwords for a Third-Party Server” in *SAS Intelligence Platform: Security Administration Guide*. Click **Next**.

6 Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the wizard settings.

---

### Register the DBMS Library

To register a DBMS library, perform the following steps:

1 In SAS Management Console, expand **Data Library Manager**. Right-click **Libraries** and select the **New Library** option to access the New Library wizard.

2 Select the database type from the **Database Data** list. Click **Next**.

3 Enter an appropriate library name in the **Name** field. The **Description** field is optional. **Location** is the metadata folder where the library object is saved. You can keep the default location or change it. Click **Next**.

4 Select an application server from the list, and use the right arrow to assign the application server. This step makes the library available to the server and makes the library visible to users of the server. Click **Next**.

5 Specify a libref on the library properties page. You can also click **Advanced Options** to perform tasks such as pre-assignment. Pre-assigning a library is valuable if your clients include SAS Enterprise Guide or SAS Add-In for Microsoft Office. Some SAS solutions require that libraries accessed from their client applications be pre-assigned (check the documentation for your specific solution for details). For more information, see Chapter 3, “Assigning Libraries,” on page 31. Click **Next** to access the next page of the wizard.

6 On the server and connection page, select the database server from the previous stage. Contact your database administrator if you are unsure of the correct value for the schema field. Click **Next**.

7 Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the library settings. At this point, register tables as explained in “Registering and Verifying Tables” on page 144.
Using the Folders Tab in SAS Management Console

All SAS folders that the user has permission to view are displayed in the Folders tab of SAS Management Console and in the Folders view of the Administration tab in SAS Environment Manager. Most other client applications display SAS folders only if they contain content that is relevant to the application, subject to the user’s permissions. For more information, see “Working with SAS Folders” in SAS Intelligence Platform: System Administration Guide.

Establishing Connectivity to a Library of SAS Data Sets

Overview of Establishing Connectivity to a Library of SAS Data Sets

The following figure provides a logical view of accessing a library of SAS data sets.

*Figure 4.1  Establishing Shared Access to SAS Data Sets*

After you have installed the required SAS software, you need to set up a connection from a SAS server to a SAS data set. This connection requires that you register the Base SAS library with the SAS Metadata Server. In addition, you must import any user-defined formats that have been created for the data set in order to view or operate on the data. Assume that the SAS software has already been loaded by
using the standard installation wizard and that the data set is stored in a location that can be accessed.

Register the library by using SAS Management Console or SAS Environment Manager. This metadata enables your SAS applications to access the data sets that you need to work with. For this example, the data set contains information about customers of the Orion Gold enterprise.

---

Register the Base SAS Library Using SAS Management Console

To register a Base SAS library, perform the following steps:

1. In SAS Management Console, expand **Data Library Manager**. Right-click **Libraries**. Then, select the **New Library** option to access the first page of the wizard.

2. Select **Base SAS Library** from the **SAS Data** list. Click **Next**.

3. Enter an appropriate library name in the **Name** field (for example, **Orion Gold Customers**). The **Description** field is optional. **Location** is the metadata folder where the library object is saved. You can keep the default location or change it. Click **Next**.

4. Select the appropriate server from the **Available servers** list and move it to the **Selected servers** list. For this example, use **SASApp**. Click **Next**.

5. Enter the following library properties:

   **Table 4.1 Library Properties**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>ORGOLD</td>
</tr>
<tr>
<td>Engine</td>
<td>BASE</td>
</tr>
<tr>
<td>Path Specification</td>
<td>Select from the <strong>Available items</strong> or click <strong>New</strong> to enter a path (for example, C:\SAS\Config \Lev1\SASApp\Data). (Enter the fully qualified path to the library. This path is specified differently in different operating systems. Make sure that the appropriate path is displayed in the <strong>Selected items</strong> field.)</td>
</tr>
</tbody>
</table>

   You can also click **Advanced Options** to perform tasks such as pre-assignment and setting host-specific and LIBNAME options. Click **Next** to access the next page of the wizard.

6. Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the wizard settings.

At this point, register tables as explained in “Registering and Verifying Tables” on page 144. If registering tables fails, check that the user account has host access permission to the path.
Register the Base SAS Library Using SAS Environment Manager 2.5 Administration

To register a Base SAS library, perform the following steps:

1. Click the Administration tab.
2. Click the Side Menu icon in the upper left corner of the window.
3. Click Libraries.
4. Click the New Library icon.
5. Enter a library Name (for example, Orion Gold). The name cannot contain a forward slash or a backslash. If necessary, change the Location. Click SAS Base Library.
6. In the Options dialog box, enter the Libref.

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>Enter an identifier of your choice (for example, OrGold). The libref name cannot contain spaces. The first character must be a letter or an underscore, and all other characters can be either letters, underscore, or numbers.</td>
</tr>
<tr>
<td>Engine</td>
<td>Click the down arrow at the right of the field. From the resulting drop-down list, select an engine. For this example, the value (Base) is the engine name.</td>
</tr>
<tr>
<td>Enable WebDAV support</td>
<td>If you need WebDAV support, select the check box.</td>
</tr>
<tr>
<td>Specify the path to the library</td>
<td>For this example, select the /install/data/SAS check box.</td>
</tr>
</tbody>
</table>

Click OK.

7. Examine the page to ensure that the proper values have been entered. Enter the Description. Click the Save icon to save the new library.

Register the Base SAS Library Using SAS Environment Manager 2.6 Administration

To register a Base SAS library, perform the following steps:

1. Click the Side Menu icon in the upper left corner of the window.
2. Click the Administrator tab.
2 Click the Libraries icon.

3 Click the New Library icon 🔄.

4 Enter a library Name (for example, Orion gold). The name cannot contain a forward slash or a backslash. If necessary, change the Location. Click SAS Base Library. The Options dialog box appears.

5 In the Options dialog box, enter the Libref.

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>Enter an identifier of your choice (for example, OrGold). The libref name cannot contain spaces. The first character must be a letter or an underscore and all other characters can be either letters, underscores, or numbers.</td>
</tr>
<tr>
<td>Engine</td>
<td>Click the down arrow at the right of the field. From the resulting drop-down list, select an engine. For this example, the value (Base) is the engine name.</td>
</tr>
<tr>
<td>Enable WebDAV support</td>
<td>If you need WebDAV support, select the check box.</td>
</tr>
<tr>
<td>Specify the path to the library</td>
<td>For this example, select the /install/data/SAS check box.</td>
</tr>
</tbody>
</table>

6 Click the Save icon 🔄 to save the new library.

For more information, see "Library Options: Base SAS" on page 177.

Working with User-Defined Formats

Choices for Working with User-Defined Formats

If you have existing SAS data sets, you might also have a catalog of user-defined formats and informats. You have two options for making these formats available to applications such as SAS Data Integration Studio and SAS Information Map Studio:

- Give the format catalog a standard name and place it in an expected location. This is the preferred method.
- Create a user-defined formats configuration file, and use the FMTSEARCH system option to point to the format catalog.

Use a Standard Name and Location for the Format Catalog

To make the format catalog available, in the preferred method, perform the following steps:

1 Name the format catalog formats.sas7bcat.
2 Place the catalog in the directory `SAS-configuration-directory\Lev1\SASApp\SASEnvironment\SASFormats`.

**Create a User-Defined Formats Configuration File**

Alternatively, you can create a user-defined formats configuration file in which you point to the location of the formats catalog.

To make a format catalog available using the alternative method on Windows and UNIX systems, perform the following steps:

1 To the SAS configuration file `SAS-configuration-directory\Lev1\SASApp\sasv9_usermods.cfg`, add the CONFIG system option, and use it to point to the user-defined formats configuration file:
   
   ```
   -config "SAS-configuration-directory\Lev1\SASApp\userfmt.cfg"
   ```

2 Then, use the FMTSEARCH system option in the same SAS configuration file to point to the format catalog:
   
   ```
   -set fmtlib1 "SAS-configuration-directory\Lev1\Data\orformat"
   -insert fmtsearch (fmtlib1.orionfmt)
   ```

   In this example, `SAS-configuration-directory\Lev1\Data\orformat` is the location of the format catalog, and `orionfmt` (filename `orionfmt.sas7bcat`) is the name of the format catalog. If you have more than one catalog to list, leave a space between each catalog name.

   **Note:** On UNIX systems, you must enter the environment variable name in uppercase. For example, you enter `FMTLIB1` instead of `fmtlib1`.

To make a format catalog available using the alternative method on z/OS systems, perform the following steps:

1 Add the AUTOEXEC system option to the SAS launch command as shown in the following example:

   ```
   SAS-configuration-directory/Lev1/SASApp/startsas.sh
   o("autoexec=./WorkspaceServer/userfmt.sas")
   ```

   In this example, `startsas.sh` is your SAS launch command script, and `userfmt.sas` is the name of the SAS autoexec file. When you enter the command, you must enter it all on one line.

2 In the autoexec file, use the LIBNAME statement to assign the format library and the OPTIONS statement to set the FMTSEARCH system option. For example, you might specify the following statements:

   ```
   LIBNAME fmtlib1
   'SAS-configuration-directory/Lev1/Data/orformat';
   options fmtsearch=(fmtlib1.orionfmt);
   ```
Establishing Shared Access to SAS Data Sets

Overview of Establishing Shared Access

The following figure provides a logical view of accessing SAS data sets through a SAS/SHARE server.

*Figure 4.2 Establishing Shared Access to SAS Data Sets*

Base SAS libraries allow the following access:

- Any number of users can read data.
- A single user can write or update data.

This access can be extended through the use of the SAS/SHARE server. A SAS/SHARE server permits multiple users to update the same items in a SAS library.

You can share access to a library of existing SAS data sets by using a SAS/SHARE server to manage access to the data. Assume that the SAS/SHARE software has already been loaded by using the standard installation wizard, and that you have registered a SAS/SHARE server in metadata (for example, SHAREServer) that was created by the wizard. Configuring shared access is a two-stage process:

1. **Create a SAS/SHARE REMOTE Engine Library.** This library is assigned to a SAS Application Server, as shown in the previous figure.

2. While creating the SAS/SHARE REMOTE Engine Library, choose the option to register a new library to the SAS/SHARE server. This is shown in the previous figure as a Base SAS library. It is very important to pre-assign this library and to assign it to the SAS/SHARE server.
Create a SAS/SHARE Remote Engine Library Using SAS Management Console

To create a SAS/SHARE Remote Engine library, perform the following steps:

1. In SAS Management Console, expand Data Library Manager. Right-click Libraries. Then, select the New Library option to access the New Library wizard.

2. Select SAS/SHARE REMOTE Engine Library from the SAS Data list. Click Next.

3. Enter an appropriate library name in the Name field (for example, SharedAccessToOrionGold). The description is optional. Location is the metadata folder where the library object is saved. You can keep the default location or change it. Click Next.

4. Select the appropriate server from the Available servers list and move it to the Selected servers list. For this example, use SASApp. Click Next.

5. Enter a value for Libref and click Next.

6. Enter the following library properties:

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAS/SHARE Server</td>
<td>SHAREServer</td>
</tr>
<tr>
<td>SAS/SHARE Server Library</td>
<td>Click New to register a new library such as a Base SAS library. Assign the new library to the SAS/SHARE Server and set the library as pre-assigned.</td>
</tr>
<tr>
<td>Connection</td>
<td></td>
</tr>
<tr>
<td>Default Login</td>
<td>(None) (This default login is used to resolve conflicts between multiple logins to an authentication domain. In such cases, the default login is used.)</td>
</tr>
</tbody>
</table>

Click Next.

7. Examine the final page of the wizard (for the SAS/SHARE REMOTE Engine Library) to ensure that the proper values have been entered. Click Finish to save the wizard settings.

8. Restart the SAS/SHARE server.

At this point, register tables as explained in “Registering and Verifying Tables” on page 144.
Create a SAS/SHARE Remote Engine Library Using SAS Environment Manager

To create a SAS/SHARE Remote Engine library, perform the following steps:

1. Click the Administration tab.
2. Click the Side Menu icon in the upper left corner of the window.
3. Click Libraries.
4. Click the New Library icon.
5. Enter a library Name (for example, SharedAccessToOrionGold). The name cannot contain a forward slash or a backslash. Change the Location using Browse to /Shared Data. Click SAS Base Library. The Options dialog box appears.
6. In the Options dialog box, enter the Libref.

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>Enter an identifier of your choice (for example, ShareLib). The libref name cannot contain spaces. The first character must be a letter or an underscore and all other characters can be either letters, underscore, or numbers.</td>
</tr>
<tr>
<td>Engine</td>
<td>Click the down arrow at the right of the field. From the resulting drop-down list, select an engine. For this example, the value (Base) is the engine name.</td>
</tr>
<tr>
<td>Enable WebDAV support</td>
<td>If you need WebDAV support, select the check box.</td>
</tr>
<tr>
<td>Specify the path to the library</td>
<td>For this example, select the /install/data/SAS check box.</td>
</tr>
</tbody>
</table>

Click OK.

7. Examine the page to ensure that the proper values have been entered. Enter the Description. Click the Save icon to save the new library.
Establishing Connectivity to a Microsoft Excel File

Overview of Establishing Connectivity to an Excel File

The following figure provides a logical view of using an Excel file as a data source.

*Figure 4.3 Establishing Connectivity to an Excel File*

The Excel file must be stored in a location that can be accessed by a Windows machine. This example focuses on a file that is local to a SAS server, but a Windows UNC path such as `\datasrv\sales\firstquarter.xls` is also acceptable.

**Note:** Mapped drives are not accessible to SAS Application Servers.

To establish connectivity to an Excel file, perform the following steps:

1. In SAS Management Console, expand **Data Library Manager**. Right-click **Libraries**. Then, select the **New Library** option to access the New Library wizard.

2. Select **Microsoft Excel Library** from the **Database Data** list. Click **Next**.

3. Enter a value for **Libref** and click **Next**.

4. Enter an appropriate library name in the **Name** field (for example, `FirstQuarterSales`). The **Description** field is optional. **Location** is the metadata folder where the library object is saved. You can keep the default location or change it. Click **Next**.

5. Select the appropriate server from the **Available servers** list and move it to the **Selected servers** list. For this example, use **SASApp**. Click **Next**.
6 On the server and connection page, click **New** in the **Server Details** group box to specify a new database server.

The New Server wizard is displayed.

7 Enter a name such as `FirstQuarterSalesFolder` in the **Name** field. Click **Next**.

8 Click **Next** on the Server Properties page.

9 On the Connection Properties page, enter the path to the Excel file. Enclose the value in quotation marks (for example, "C:\sales\firstquarter.xls"). Click **Next**.

10 Examine the final page of the New Server wizard to ensure that the proper values have been entered. Click **Finish** to save the server settings.

The New Server wizard closes and returns you to the final page of the New Library wizard.

11 Examine the final page of the New Library wizard to ensure that the proper values have been entered. Click **Finish** to save the library settings.

At this point, register tables as explained in "Registering and Verifying Tables" on page 144.

The Microsoft Excel Library wizard in SAS Management Console generates the metadata to construct a LIBNAME statement for the EXCEL LIBNAME engine. For more information about the EXCEL LIBNAME engine and supported options, see the **SAS/ACCESS Interface to PC Files: Reference**.

---

**Establishing Connectivity to a Flat File**

The following figure provides a logical view of using an external file as a data source.
You can connect to a flat file using the External File Source Designer in SAS Data Integration Studio.

Assume that the SAS software has already been loaded by using the standard installation wizard, and that the flat file is stored in a location that can be accessed. This example focuses on a comma-delimited flat file. A similar process is used for other types of flat files, but some steps are different.

To establish a connection to a flat file, perform the following steps:

1. Open SAS Data Integration Studio. Then, select **File** ➔ **New** ➔ **External File** ➔ **Delimited** to access the **New Delimited External File** wizard.

2. Enter a name for the external file and click **Next**.

3. Enter the fully qualified path to the file in the **File name** field (for example, `SAS-configuration-directory\sources\customer_data.dat`). Click **Next**.

4. On the **Delimiters and Parameters** page of the wizard, deselect the **Blank** option in the **Delimiters** group box. Then, select the **Comma** option. Click **Next** to access the **Column Definitions** page of the wizard.

5. To define the columns, perform the following steps:
   a. Click **Refresh** to view the data from the flat file in the **File** tab in the view pane at the bottom of the page.
   b. Click **Auto Fill** to access the Auto Fill Columns dialog box. Change the value entered in the **Start record** field in the **Guessing records** group box to 2. This setting is based on the assumption that the first data record of the flat file contains header information and that the record is unique because it holds the column names for the file. Therefore, excluding the first data record from the guessing process yields more accurate preliminary data because it is excluded when the guessing algorithm is run.
6 Click **Import** to access the Import Column Definitions dialog box. The following four methods are provided for importing column definitions:

- Get the column definitions from other existing tables or external files.
- Get the column definitions from a format file.
- Get column definitions from a COBOL format file.
- Get the column names from column headings in the file.

In most cases, you get the column definitions from an external format file or from the column headings in the external file. Here is an example of a format file:

```
# Header follows
Name,SASColumnType,SASColumnName,SASColumnLength,SASInformat,SASFormat,Desc,ReadFlag
# Column definition records records follow
Make,C,Make,13,$char13.,Manufacturer name column,y
Model,C,Model,40,$char40.,Model name column,y
# Comma within quotation marks below is not a delimiter
Description,C,Description,32,$char32.,'Description, Comments, Remarks',y
```

A sample of the output is shown in the following figure:

For this example, select the **Get the column names from column headings in the file** radio button. Keep the default settings for the fields underneath it.

**Note:** If you select **Get the column names from column headings in the file**, the value in the **Starting record** field in the **Data** tab of the view pane in the Column Definitions dialog box is automatically changed. The new value is one greater than the value in the **The column headings are in file record** field in the Import Column Definitions dialog box.

7 Click **OK** to return to the Column Definitions page.

8 The preliminary data for the external file object is displayed in the columns table at the top of the page. The **Informat** and **Format** columns for the rows in the table are based on the values that are included in the sample data that is processed by the guessing function. The results are accurate for this particular set of records, but you should still examine them to make sure that they are representative of the data in the rest of the flat file. Edit the values by clicking directly on the cells in the column table and making the necessary changes.

9 Click the **Data** tab at the bottom of the Column Definitions page. Then, click **Refresh**. The data should be properly formatted. If not, edit the cells in the column table and check the results by refreshing the **Data** tab. You can repeat this process until you are satisfied. You can review the SAS log for more details.

**Note:** To view the code that is generated for the external file, click the **Source** tab. To view the SAS log for the generated code, click the **Log** tab. The code that
is displayed in the **Source** tab is the code that is generated for the current external file.

10 Click **Next**.

11 Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the library settings. The file is ready for use.

---

**Establishing Connectivity to XML Data**

The following figure provides a logical view of using XML files as a data source.

*Figure 4.5 Establishing Connectivity to XML Files*

The following steps describe how to specify a SAS XML library in SAS Management Console. Assume that the XML library points to an XML file that contains climate information *(climate.xml)*. The XML file is in generic format, as defined for the SAS XML LIBNAME engine. For more information, see the *SAS XMLV2 and XML LIBNAME Engines: User’s Guide*.

To register an XML library, perform the following steps:

1. In SAS Management Console, expand **Data Library Manager**. Right-click **Libraries**. Then, select the **New Library** option to access the New Library wizard.

2. Select **SAS XML library** from the **SAS Data** list. Click **Next**.

3. Enter an appropriate library name in the **Name** field (for example, *XML Lib*). The **Description** field is optional. **Location** is the metadata folder where the library object is saved. You can keep the default location or change it. Click **Next**.
Select the appropriate server from the **Available servers** list and move it to the **Selected servers** list. For this example, use **SASApp**. Click **Next**.

Enter information about the library, such as the following:

**Table 4.3  Library Properties**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>xmlllib</td>
</tr>
<tr>
<td>Engine</td>
<td>XML</td>
</tr>
<tr>
<td>XML File Location</td>
<td>C:\sources\xml\climate.xml</td>
</tr>
<tr>
<td>XML Type</td>
<td>GENERIC</td>
</tr>
<tr>
<td>Library Access</td>
<td>READONLY</td>
</tr>
</tbody>
</table>

Click **Next**.

Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the wizard settings.

---

**Establishing Connectivity to a SAS Information Map**

**Overview of Establishing Connectivity to a SAS Information Map**

A SAS Information Map is a business metadata layer on top of another data source. When an information map is registered as a table in a SAS Information Map library, it can also be used as the data source for other information maps. For more information about creating information maps, see either the **SAS Information Map Studio Help** or **Base SAS Guide to Information Maps**. Information map tables are expected to be used primarily by SAS Information Map Studio.
To register a SAS Information Map library, perform the following steps:

1. In SAS Management Console, expand **Data Library Manager**. Right-click **Libraries**. Then, select the **New Library** option to access the New Library wizard.

2. Select **SAS Information Map Library** from the **SAS Data** list. Click **Next**.

3. Enter an appropriate library name in the **Name** field (for example, **InfoMapLib**). The **Description** field is optional. **Location** is the metadata folder where the library object is saved. You can keep the default location or change it. Click **Next**.

4. Select the appropriate server from the **Available servers** list and move it to the **Selected servers** list. For this example, use **SASApp**. Click **Next**.

5. Enter information about the library, such as the following:

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>maplib</td>
</tr>
<tr>
<td>Engine</td>
<td>SASIOIME</td>
</tr>
<tr>
<td>Metadata server</td>
<td>select a metadata server from the list</td>
</tr>
<tr>
<td>Default login</td>
<td>(None)</td>
</tr>
<tr>
<td>Information map location</td>
<td>/Shared Data/SASInfoMaps</td>
</tr>
</tbody>
</table>

   Click **Next**.

6. Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the library settings. At this point, register tables as explained in “Registering and Verifying Tables” on page 144.
Special Considerations for Information Map Tables

- When registering the tables with SAS Management Console, ensure that the check box for the option **Enable special characters within DBMS object names** option is selected. The New Library wizard uses data item IDs from the source information map as column names for the table. These data item IDs sometimes contain special characters.

- The data item IDs of a source information map are used as column names when the information map is registered as an information map table. If you change the data item ID or table column name after the table is registered, then you will not be able to run queries on the table.

- By default, the values retrieved from an information map table are the detail values from the source information map. If you want to retrieve aggregated values, then you must set an aggregation option. You can set the AGGREGATE= options for tables and libraries in Data Library Manager in SAS Management Console. After you add a table as a data source for an information map, you can also set the aggregation option for the data source in the Table Properties dialog box in SAS Information Map Studio. Selecting the **Use the detail values from the data source** radio button is equivalent to setting the AGGREGATE= option to NO, and selecting the **Use the aggregated values from the data source** radio button is equivalent to setting the AGGREGATE= option to YES. This setting overrides the aggregation settings on the library or the table.

- When referenced by an information map table, measure data items that reference other measure data items or that use aggregate functions in their expressions can produce aggregated values only. If a source information map contains one of these data items, then set the AGGREGATE= option to YES on the library. Otherwise, these data items are not registered as columns when you register the information map as a table.

- If the source information map for an information map table has measure data items that reference other measure data items or that use aggregate functions in their expressions, then the combination of setting the AGGREGATE= option to YES on the library and setting the AGGREGATE= option to NO on the table or on the data source (within SAS Information Map Studio) causes the table data to be inaccessible.

- Normally, when an information map table is registered, its columns get their formats from the associated source data items. However, when a source data item is a measure data item and has a data type of character, then if the AGGREGATE= option is set to NO on the library, the format is not set to the format of the source data item. The format is set to the format of the column that the source data item is based on.

- Because an information map acts as a reference to underlying data, ReadMetadata permission must be granted to a user for the information map table, the source information map, and the table used by the source information map. Read permission is also needed on the source information map.

- Stored processes and prefilters associated with a source information map are applied to the information map table. (Filters that are not used as prefilters are not applied.)
If a stored process is associated with a source information map and the stored process uses prompts with default values, the stored process is applied to the information map table. If the prompts do not have default values, the stored process does not affect the information map table.

Information maps created from OLAP cubes cannot be registered as information map tables.

The source information map name must be 32 bytes or less.

Establishing Connectivity to a Cloud Analytic Services (CAS) Server

Overview of Establishing Connectivity to CAS

The following figure provides a logical view of connecting to a CAS Server.

To define a user or user group using SAS Management Console, see “Define the DBMS User” on page 49 and “Define the DBMS User Group” on page 50.

To define a user or user group using SAS Environment Manager, see “Managing User Access” in SAS® Environment Manager User’s Guide.

Setting up a connection from SAS to a CAS Server is a two-stage process:

1. Register the CAS server.
2. Register the CAS library.
Stage 1: Register the CAS Server

To register the CAS server, perform the following steps:

1. Open SAS Management Console application.
2. Right-click Server Manager, and select the New Server option to access the New Server wizard.
3. Select Cloud Analytic Services Server from the SAS Servers list. Click Next.
4. Specify a name for the new CAS server (for example, casServer). The description is optional.
5. Enter the following server properties. A SAS Viya 3.3 installation is used as the example:

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Major Version Number</td>
<td>3</td>
</tr>
<tr>
<td>Minor Version Number</td>
<td>3</td>
</tr>
<tr>
<td>Software Version</td>
<td>3.3</td>
</tr>
<tr>
<td>Vendor</td>
<td>SAS Institute</td>
</tr>
<tr>
<td>Associated Machine</td>
<td>Defaults to your machine, click New to specify a different machine.</td>
</tr>
</tbody>
</table>

   Click Next.

6. On this page, enter the connection properties. Click the Server Information button. Click Options. For this example, enter the connection properties. Server and Port information are optional. For Authentication Type, select None from the drop-down menu.

   Click OK.

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication Type</td>
<td>Select None.</td>
</tr>
<tr>
<td>Authentication Domain</td>
<td>The default is DefaultAuth.</td>
</tr>
</tbody>
</table>

   Click Next.

7. Examine the final page of the wizard to ensure that the proper values have been entered. Click Finish to save the wizard settings.
Stage 2: Register the Cloud Analytic Services (CAS) Library

After you have registered the server, you can register the library. To register the library, perform the following steps:

1. In SAS Management Console, expand *Data Library Manager*. Right-click *Libraries*. Then, select the *New Library* option to access the New Library wizard.

2. From the *SAS Data* list, select *Database Data* ⇒ *CAS Library*. Click *Next*.

3. Enter an appropriate library name in the *Name* field (for example, *CASLibrary*). The *Description* field is optional. *Location* is the metadata folder where the library object is saved. You can keep the default location or use *Browse* to change it. Click *Next*.

4. Select an application server from the list, and use the right arrow to assign the application server. This step makes the library available to the server and makes the library visible to users of the server. Click *Next*.

5. Enter the following library properties:

   *Table 4.7  Library Properties*

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>CasRef</td>
</tr>
<tr>
<td>Engine</td>
<td>The default is CAS</td>
</tr>
<tr>
<td>CASLIB</td>
<td>Enter the CAS library location.</td>
</tr>
<tr>
<td>SESSREF</td>
<td>Enter the CAS session reference.</td>
</tr>
</tbody>
</table>

   You can also click *Advanced Options* to identify *Options to be appended*. Click *Next*.

6. Enter the following settings:

   *Table 4.8  Server and Connection Information*

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Server</td>
<td><em>CAS Server</em> (Use the server that you created in the New Server wizard.) Click <em>New</em> to change the server.</td>
</tr>
<tr>
<td>Connection</td>
<td>Use the default value of Connection: <em>server_name</em>. Click <em>New</em> to change the server.</td>
</tr>
<tr>
<td>Default Login</td>
<td>Use the default value of <em>(None)</em>.</td>
</tr>
</tbody>
</table>
Click **Next**.

7 Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the library settings.

---

**Establishing Connectivity to Several Different SAS/ACCESS Servers**

**Overview of Establishing Connectivity to a Greenthumb, Hawq, Netessa, Sybase IQ, Postgres, or Vertica Server**

The following instructions are for several different SAS/ACCESS servers. The following figure provides a logical view of connecting to several different SAS/ACCESS Servers and accessing tables from multiple data sources registered there.

*Figure 4.7 Establishing Connectivity to Several SAS/ACCESS Servers*

After you define a user or user group using SAS Management Console, follow this two-stage process to set up a connection from a SAS session to a server:

1. Register the SAS/ACCESS Server.
2. Register the SAS/ACCESS Server library.
Register the DBMS Server

The following instructions are for several SAS/ACCESS servers.

1. Right-click Server Manager and select the New Server option to access the New Server wizard.

2. Select the appropriate database server from the Database Servers list. Click Next.

3. Enter an appropriate server name in the Name field. The Description is optional. Click Next.

4. Enter the Major version and Minor version numbers. The Software version is the default for the server. The example uses 1.1 release. Enter a value in the Associated Machine field. Click Next.

5. You are required to select at least one field. Enter the appropriate information.

Figure 4.8 SAS/ACCESS Connection Properties

If you choose ODBC datasource, click Options. Enter the ODBC datasource. Enter Authentication Information if it is different from the default information.
Establishing Connectivity to Several Different SAS/ACCESS Servers

Stage 2: Register the SAS/ACCESS Server Library

After you have registered the SAS/ACCESS Server, you can register the SAS/ACCESS Server library.

1. In SAS Management Console, expand Data Library Manager. Then, right-click Libraries and select the New Library option to access the New Library wizard.

2. Select the SAS/ACCESS Library from the SAS Data list. Click Next.

3. Enter an appropriate library name in the Name field (for example, Greenthumb Library). The Description is optional. Database is the metadata folder where the library object is saved. You can keep the default location or change it. You can also click Advanced Options to perform tasks such as pre-assignment and optimization.

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>ACCESSLIB</td>
</tr>
<tr>
<td>Engine</td>
<td>Name of product</td>
</tr>
<tr>
<td>Database</td>
<td>Enter the metadata database name.</td>
</tr>
</tbody>
</table>
Click **Next**.

4. Select an application server from the list and use the right arrow to assign the application server. Click **Next**.

5. Enter the **Server Details** and **Connection Details**. Some of the server and connection information might be entered automatically.

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Server:</td>
<td>Enter the database server.</td>
</tr>
<tr>
<td>Database Schema Name:</td>
<td>Enter the database schema.</td>
</tr>
<tr>
<td>Connection:</td>
<td>Enter the connection string.</td>
</tr>
<tr>
<td>Default Login:</td>
<td>Enter the default login.</td>
</tr>
</tbody>
</table>

Click **Next**. Verify that the information is correct.

6. Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the library settings.

---

**Establishing Connectivity to an Aster Server**

**Overview of Establishing Connectivity to an Aster Server**

The following figure provides a logical view of connecting to an Aster Server and accessing tables from multiple data sources registered there.
To define a user or user group using SAS Management Console, see “Define the DBMS User” on page 49 and “Define the DBMS User Group” on page 50.

To define a user or user group using SAS Environment Manager, see "Managing User Access" in SAS Environment Manager User's Guide.

After you define a user or user group, follow this two-stage process to set up a connection from a SAS session to a server:

1. Register the Aster Server.
2. Register the Aster Server library.

Register the DBMS Server

The following instructions use Aster.

1. Right-click Server Manager and select the New Server option to access the New Server wizard.

2. Select the Aster database server from the Database Servers list. Click Next.

3. Enter an appropriate server name in the Name field. Description is optional. Click Next.

4. Enter the Major version and Minor version numbers. The Software version is the default for the server. The example uses 1.1 release. Enter a value in the Associated Machine field. Click Next.
Figure 4.12  Aster Server Properties

5 You are required to select at least one field.
If you choose **Server Information**, then click **Options**. Add the server name. Change the port number if it is not correct. Click **OK**.

If you choose **ODBC Datasource**, then click **Options**. Add the data source name. Change the port number if it is not correct. Click **OK**.

If you choose **Connection String**, then click **Options**. Add the connection string name. Change the port number if it is not correct. Click **OK**.

Click **Next**.

6. Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the wizard settings.
Stage 2: Register the Aster Server Library

After you have registered the Aster Server, you can register the Aster Server library.

1. In SAS Management Console, expand Data Library Manager. Then, right-click Libraries and select the New Library option to access the New Library wizard.

2. Select Aster Library from the SAS Data list. Click Next.

3. Enter an appropriate library name in the Name field (for example, Aster Library). The Description is optional. Database is the metadata folder where the library object is saved. You can keep the default location or change it. You can also click Advanced Options to perform tasks such as pre-assignment and optimization.

   Table 4.11  Library Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>LIBASTER</td>
</tr>
<tr>
<td>Engine</td>
<td>ASTER</td>
</tr>
<tr>
<td>Database</td>
<td>Enter the metadata database name.</td>
</tr>
</tbody>
</table>

   Click Next.

4. Select an application server from the list and use the right arrow to assign the application server.
   Click Next.

5. Enter the Server Details and Connection Details. Some of the server and connection information might be entered automatically.

   Table 4.12  Library Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Server:</td>
<td>Enter the Aster database server.</td>
</tr>
<tr>
<td>Database Schema:</td>
<td>Enter the Aster database schema.</td>
</tr>
<tr>
<td>Connection:</td>
<td>Enter the connection string.</td>
</tr>
<tr>
<td>Default Login:</td>
<td>Enter the default login/</td>
</tr>
</tbody>
</table>

   Click Next. Verify that the information is correct.

6. Examine the final page of the wizard to ensure that the proper values have been entered. Click Finish to save the library settings.
Establishing Connectivity to a SAS Federation Server

Overview of Establishing Connectivity to a SAS Federation Server

The following figure provides a logical view of connecting to a SAS Federation Server and accessing tables from multiple data sources registered there, using a federated DSN that contains multiple data source names.

Figure 4.15 Establishing Connectivity to a SAS Federation Server

To define a user or user group using SAS Management Console, see "Define the DBMS User" on page 49 and "Define the DBMS User Group" on page 50.

To define a user or user group using SAS Environment Manager, see "Managing User Access" in SAS® Environment Manager User’s Guide.

After defining a user or user group, setting up a connection from a SAS session to a SAS Federation Server is a two-stage process:

1. Register the SAS Federation Server.

2. Register the SAS Federation Server library.

If you do not know the DSN or connection string to use for accessing the data, then contact the server administrator for that information.
Stage 1: Register the SAS Federation Server

To register the SAS Federation Server, perform the following steps:

1. Open the SAS Management Console application.

2. Right-click Server Manager and select the New Server option to access the New Server wizard.

3. Select SAS Federation Server from the SAS Servers list. Click Next.

4. Enter an appropriate server name in the Name field (for example, Federation Server). Note that you can supply an optional description if you want. Click Next.

5. Enter the following server properties:

   Table 4.13 Server Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Major Version Number</td>
<td>3</td>
</tr>
<tr>
<td>Minor Version Number</td>
<td>2</td>
</tr>
<tr>
<td>Software Version</td>
<td>3.2</td>
</tr>
<tr>
<td>Vendor</td>
<td>SAS Institute</td>
</tr>
</tbody>
</table>

   Click Next.

6. Leave the connection type set to SAS Connection. Click Next.

7. Enter the following connection properties:

   Table 4.14 Connection Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port number</td>
<td>21032</td>
</tr>
<tr>
<td>Host name</td>
<td>Specify the host name for the SAS Federation Server.</td>
</tr>
<tr>
<td>Authentication type</td>
<td>User/Password</td>
</tr>
</tbody>
</table>
Stage 2: Register the SAS Federation Server Library

After you have registered the SAS Federation Server, you can register the SAS Federation Server library.

To register the library, perform the following steps:

1. In SAS Management Console, expand **Data Library Manager**. Then, right-click **Libraries** and select the **New Library** option to access the New Library wizard.

2. Select **SAS Federation Server Library** from the **SAS Data** list. Click **Next**.

3. Enter an appropriate library name in the **Name** field (for example, **Federated Data Library**). The **Description** field is optional. **Location** is the metadata folder where the library object is saved. You can keep the default location or change it. Click **Next**.

4. Select an application server from the list and use the right arrow to assign the application server. Click **Next**.

5. Enter the following library properties:

   **Table 4.15  Library Properties**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>FEDREF</td>
</tr>
<tr>
<td>Engine</td>
<td>FEDSVR</td>
</tr>
<tr>
<td>Identify Connection Information</td>
<td>Select the radio button for <strong>Data Source Name</strong> and specify the DSN. If you prefer to specify a connection string, then select the radio button for <strong>Connection String</strong>.</td>
</tr>
</tbody>
</table>
You can also click **Advanced Options** to perform tasks such as pre-assignment and optimization. Click **Next** to access the next page of the wizard.

6. **Enter the following settings:**

   **Table 4.16  Server and Connection Information**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Server</td>
<td>Federation Server (Use the server that you created in the New Server wizard.)</td>
</tr>
<tr>
<td>Database Schema Name</td>
<td>See your administrator for the correct value.</td>
</tr>
<tr>
<td>Connection</td>
<td>Use the default value of Connection: server_name.</td>
</tr>
<tr>
<td>Default Login</td>
<td>Use the default value of (None).</td>
</tr>
</tbody>
</table>

   Click **Next**.

7. **Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the library settings. At this point, register tables as explained in “Registering and Verifying Tables” on page 144.**

---

**Special Considerations for SAS Federation Server Libraries**

A federated DSN can be configured with several schemas so that data from several data sources can be federated under a single DSN. However, when you register tables for a library that uses a federated DSN, you can register the tables from the first schema only.

For more information about how the LIBNAME engine for SAS Federation Server supports access to one data source at a time, see “Data Source Connection” in *SAS LIBNAME Engine for SAS Federation Server: User’s Guide*. 

---

**Establishing Connectivity to Hadoop**

---

**Overview of Establishing Connectivity to Hadoop**

The following figure provides a logical view of using the SAS/ACCESS Interface to Hadoop to access a Hive Server. The Hive Server is shown running on the same machine as the Hadoop NameNode.
Figure 4.16  Establishing Connectivity to Hadoop Servers

Establishing Connectivity to Hadoop Servers

The example shows the process for establishing a SAS connection to a Hive Server. In order for the SAS/ACCESS interface to connect with the Hive Server, the machine that is used for the SAS Workspace Server must be configured with several JAR files. These JAR files are used to make a JDBC connection to the Hive Server. The following prerequisites have been satisfied:

- installation of SAS/ACCESS Interface to Hadoop. For more information, see the configuration guide for your operating system and SAS version at SAS 9.4 Install Center Documentation.
- installation of the Hadoop JAR files required by SAS. For more information, see the “Hadoop JAR Files” in SAS Hadoop Configuration Guide for Base SAS and SAS/ACCESS or the configuration guide for your operating system and SAS version at SAS 9.4 Install Center Documentation.
- setting the SAS_HADOOP_JAR_PATH environment variable. See the “SAS Environment Variables for Hadoop” appendix in SAS Hadoop Configuration Guide for Base SAS and SAS/ACCESS.

To define a user or user group using SAS Management Console, see “Define the DBMS User” on page 49 and “Define the DBMS User Group” on page 50.

To define a user or user group using SAS Environment Manager, see “Managing User Access” in SAS® Environment Manager User's Guide.

Setting up a connection from SAS to a Hadoop Server is a two-stage process:

1. Register the Hadoop Server.
2. Register the Hadoop via Hive library.

This section describes the steps that are used to access data in Hadoop as tables through a Hive Server. SAS Data Integration Studio offers a series of transformations that can be used to access the Hadoop Distributed File System (HDFS), submit Pig code, and submit MapReduce jobs.

Hadoop Configuration Resources

Starting in the second maintenance release of SAS 9.4, a new support site for Hadoop has been added to SAS Third-Party Software Reference. This site includes a new guide, SAS Hadoop Configuration Guide for Base SAS and SAS/ACCESS,
Stage 1: Register the Hadoop Server

To register the Hadoop Server, perform the following steps:

1. Open SAS Management Console application.
2. Right-click **Server Manager** and select the **New Server** option to access the New Server wizard.
3. Select **Hadoop Server** from the **Cloud Servers** list. Click **Next**.
4. Enter an appropriate server name in the **Name** field (for example, **HadoopServer**). You can supply an optional description. Click **Next**.
5. In the second quarter of 2015, an Encryption Options section was added to SAS Management Console.

Enter the following server properties:

<table>
<thead>
<tr>
<th>Table 4.17 Server Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Field</strong></td>
</tr>
<tr>
<td>Major Version Number</td>
</tr>
<tr>
<td>Minor Version Number</td>
</tr>
<tr>
<td>Software Version</td>
</tr>
<tr>
<td>Vendor</td>
</tr>
<tr>
<td>Associated Machine</td>
</tr>
<tr>
<td>Enable Encryption</td>
</tr>
<tr>
<td><strong>Note:</strong> Enabling encryption does not affect existing non-encrypted tables. Move any non-encrypted tables to a backup location before enabling encryption on an existing server definition. After you enable encryption, move the table back to the server. The tables are encrypted as they are written to the server.</td>
</tr>
</tbody>
</table>

| **New Encrypt Key:** | If you enabled encryption, specify the encryption key. |
| **Confirm Encrypt Key:** | Confirm the encryption key. |
Click **Next**.

6 Enter the following connection properties:

**Table 4.18 Connection Properties**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>HiveService Node</td>
<td>Specify the host name of the machine that is running the HiveServer2. For this example, use hdp21d1.</td>
</tr>
<tr>
<td>Port Number</td>
<td>Specify the network port number for the Hive Service. For this example, use the default 10000.</td>
</tr>
<tr>
<td>Environment Install Location</td>
<td>/opt/TKGrid (For deployments that use SAS Visual Analytics with Hadoop, specify the path to TKGrid on the machines in the cluster.)</td>
</tr>
<tr>
<td>LASR Authorization Service</td>
<td>Choose whether to select the <strong>Use LASR authorization service</strong> check box. If you select this check box, you must also specify the location of the authorization service. This option applies only to SASHDAT libraries that use this server. This check box is automatically selected and cannot be cleared if you enabled on-disk AES encryption for libraries on this server.</td>
</tr>
<tr>
<td>Authentication type</td>
<td>Use the default value of <em>(none)</em> for this example.</td>
</tr>
<tr>
<td>Field</td>
<td>Sample Value</td>
</tr>
<tr>
<td>------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Authentication domain</td>
<td>DefaultAuth (You might need to create a new authentication domain. For more information, see “How to Store Passwords for a Third-Party Server” in SAS Intelligence Platform: Security Administration Guide.) Click New to access the New Authentication Domain dialog box. Then enter the appropriate value in the Name field and click OK to save the setting.</td>
</tr>
</tbody>
</table>

**Configuration**

For this example, use the default.

**NameNode HTTP Address**

For this example, use the default.

**Secondary NameNode HTTP Address**

For this example, use the default.

**Job Tracker HTTP Address**

For this example, use the default.

---

7 Click **Next**. Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the wizard settings.
Stage 2: Register the Hadoop via Hive Library

After you have registered the Hadoop Server, register the library. To register the **Hadoop via Hive Library**, perform the following steps:

1. In SAS Management Console, expand **Data Library Manager**. Right-click **Libraries**. Then, select the **New Library** option to access the New Library wizard.

2. Select **Hadoop via Hive Library** from the **Database Data** list. Click **Next**.

3. Enter an appropriate library name in the **Name** field (for example, **Hive Library**). The **Description** field is optional. **Location** is the metadata folder where the library object is saved. You can keep the default location or change it. Click **Next**.

4. Select a SAS server from the list and use the right arrow to assign the SAS server. This step makes the library available to the server and makes the library visible to users of the server. Click **Next**.

5. Enter the following library properties:

   **Table 4.19  Library Properties**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>HIVEREF</td>
</tr>
<tr>
<td>Engine</td>
<td>HADOOP</td>
</tr>
</tbody>
</table>

6. Click **Advanced Options**. Then, click the **Connection** tab.

7. Enter the following library properties:

   **Table 4.20  Library Properties**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hive Service</td>
<td>For this example, use the default HIVE2.</td>
</tr>
<tr>
<td>Hive Properties</td>
<td>Leave blank for this example.</td>
</tr>
<tr>
<td>Hive Kerberos Principal</td>
<td>For this example, use hive/HiveServer2Host@YOUR-REALM.COM. See the following figure.</td>
</tr>
</tbody>
</table>
8 Click OK. Click Next to access the next page of the wizard.

9 Enter the following settings:

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Server</td>
<td>HadoopServer (Use the Hadoop Server that you created in the New Server wizard.)</td>
</tr>
<tr>
<td>Database Schema Name</td>
<td>See your Hadoop administrator for the correct value.</td>
</tr>
<tr>
<td>Connection</td>
<td>Use the default value of Connection: server_name.</td>
</tr>
<tr>
<td>Default Login</td>
<td>Use the default value of None.</td>
</tr>
</tbody>
</table>

Click Next.

10 Examine the final page of the wizard to ensure that the proper values have been entered. Click Finish to save the library settings. At this point, register tables as explained in “Registering and Verifying Tables” on page 144.
Special Considerations for Hadoop via Hive Tables

Hadoop via Hive tables can be registered in metadata with clients such as SAS Management Console and SAS Data Integration Studio. However, table metadata cannot be updated after the table is registered in metadata.

Establishing Connectivity to an Impala Server

Overview of Establishing Connectivity to Impala

The following figure provides a logical view of using the SAS/ACCESS Interface to Impala.

To define a user or user group using SAS Management Console, see “Define the DBMS User” on page 49 and “Define the DBMS User Group” on page 50.

To define a user or user group using SAS Environment Manager, see “Managing User Access” in SAS® Environment Manager User’s Guide.

Setting up a connection from SAS to an Impala Server is a two-stage process:

1. Register the Impala server.
2. Register the Impala library.
Stage 1: Register the Impala Server

To register the Impala server, perform the following steps:

1. Open SAS Management Console application.

2. Right-click **Server Manager** and select the **New Server** option to access the New Server wizard.

3. Select **Impala Server** from the **Database Servers** list. Click **Next**.

4. Specify a name for the new Impala server. The **Description** is optional.

5. Enter the following server properties. An Impala 1.0 installation is used as the example:

   **Table 4.22 Server Properties**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Major Version Number</td>
<td>1</td>
</tr>
<tr>
<td>Minor Version Number</td>
<td>0</td>
</tr>
<tr>
<td>Software Version</td>
<td>1.0</td>
</tr>
<tr>
<td>Vendor</td>
<td>Cloudera Inc.</td>
</tr>
<tr>
<td>Associated Machine</td>
<td>Defaults to your machine, Click <strong>New</strong> to specify a different machine.</td>
</tr>
</tbody>
</table>

   Click **Next**.

6. On this page, enter the connection properties. Click the **Server Information** button. Click **Options**. For this example, enter the following connection properties:

   **Figure 4.20 Impala Server Option**

   ![Server Information Options](image)

   Click **OK**.
### Table 4.23 Connection Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Server Information:</strong></td>
<td></td>
</tr>
<tr>
<td>ODBC datasource information:</td>
<td></td>
</tr>
<tr>
<td><strong>Authentication Type</strong></td>
<td>The default is <strong>User/Password</strong>. Select a value from the drop-down list.</td>
</tr>
<tr>
<td><strong>Authentication Domain</strong></td>
<td><strong>DefaultAuth</strong> (You might need to create a new authentication domain. For more information, see “How to Store Passwords for a Third-Party Server” in SAS Intelligence Platform: Security Administration Guide.) Click <strong>New</strong> to access the New Authentication Domain dialog box. Then enter the appropriate value in the <strong>Name</strong> field and click <strong>OK</strong> to save the setting.</td>
</tr>
</tbody>
</table>

7. Click **Next**.

8. Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the wizard settings.

---

**Stage 2: Register the Impala Library**

After you have registered the server, you can register the library. To register the library, perform the following steps:

1. In SAS Management Console, expand **Data Library Manager**. Right-click **Libraries**. Then, select the **New Library** option to access the New Library wizard.

2. Select **Impala Library** from the **Database Data** list. Click **Next**.

3. Enter an appropriate library name in the **Name** field (for example, **Impala Library**). The **Description** field is optional. **Location** is the metadata folder where the library object is saved. You can keep the default location or change it. Click **Next**.

4. Select an application server from the list and use the right arrow to assign the application server. This step makes the library available to the server and makes the library visible to users of the server. Click **Next**.

5. Enter the following library properties:

   **Table 4.24 Library Properties**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>ImpLib</td>
</tr>
</tbody>
</table>
Click **Next**.

6. Select the Impala server that you entered in the **Database Server** field of the New Server wizard (for example, *ImpalaServer*) by using the **Database Server** drop-down list. Click **Next**.

7. Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the library settings.

---

### Establishing Connectivity to a SAS LASR Analytic Server

#### Overview of Establishing Connectivity to a SAS LASR Analytic Server

The following figure provides a logical view of connecting to a SAS LASR Analytic Server as a data source.

*Figure 4.21  Establishing Connectivity to a SAS LASR Analytic Server*

You can set up a connection using SAS Management Console, SAS Environment Manager Administration 2.5, SAS Environment Manager Administration 2.6, and SAS LASR Analytics.
Use SAS Environment Manager 2.6 Administration to Establish Connectivity to SAS LASR Analytic Server

Important: If SAS Visual Analytic Server 7.5 is installed, go to SAS Environment Manager 2.6 Administration.

Use SAS Environment Manager 2.6 Administration to Define a SAS LASR Analytic Server User or User Group

If other users can access this table, add them as users. For detailed information, see “Adding Users” in SAS Visual Analytics: Administration Guide.

Use SAS Environment Manager 2.6 Administration to Define a SAS LASR Analytic Server

For detailed information, see “Add a LASR Server” in SAS Visual Analytics: Administration Guide.

Use SAS Environment Manager 2.6 Administration to Define a SAS LASR Analytic Server Library

For detailed information, see “Add a LASR Library” in SAS Visual Analytics: Administration Guide.

Use SAS Environment Manager 2.5 Administration to Establish Connectivity to SAS LASR Analytic Server

Use SAS Environment Manager 2.5 Administration to Define a SAS LASR Analytic Server User or User Group


3. To add a user group, return to the main menu and click “Add a Custom Group” in SAS Environment Manager Administration: User’s Guide. Follow the steps.

Use SAS Environment Manager 2.5 Administration to Define a SAS LASR Analytic Server

2 Click “Create a New Server Definition” in SAS Environment Manager Administration: User’s Guide. Follow the steps.

Use SAS Environment Manager 2.5 Administration to Define a SAS LASR Analytic Server Library


2 Click “Creating a New Library Definition” in SAS Environment Manager Administration: User’s Guide. Follow the steps.

Use SAS Management Console to Establish Connectivity to SAS LASR Analytic Server

Use SAS Management Console to Define a SAS LASR Analytic Server User or User Group

To define a user or user group using SAS Management Console, see “Define the DBMS User” on page 49.

Use SAS Management Console to Define a SAS LASR Analytic Server


2 In the New Server wizard, select SAS Servers ➔ SAS LASR Analytic Server. Click Next.

3 Enter a name for the server. Click Next.

4 Enter the following server properties:

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single machine server</td>
<td>For a distributed server, select No. For a non-distributed server, select Yes.</td>
</tr>
<tr>
<td>High-Performance Analytics environment install location</td>
<td>Specify the host path where files that define the cluster are located (for example, /opt/TKGrid). This field is applicable to a distributed server only.</td>
</tr>
<tr>
<td>Number of machines to use</td>
<td>Accept the default value (ALL). This field is applicable to a distributed server only.</td>
</tr>
</tbody>
</table>
Click the **Advanced Options** button.

5 In the Advanced Options window, click the **Additional Options** tab. Make sure that **Signature files location on server** field references a directory that has appropriate host protection.

When you are finished setting advanced options, click **OK** in the Advanced Options window. In the wizard, click **Next**.

6 Enter the following connection properties:

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Port number</strong></td>
<td>Enter a unique port number.</td>
</tr>
<tr>
<td><strong>High-Performance Analytics environment host</strong></td>
<td>Enter the fully qualified machine name of the host (for example, va.abc.com).</td>
</tr>
<tr>
<td><strong>Use LASR authorization service</strong></td>
<td>Leave this check box selected.</td>
</tr>
</tbody>
</table>

7 Click **Next**.

8 If you want to add explicit grants of the Administer permission on the server, then move identities from one list to the other. Click **Next**.

   **Note**: Only users who have the Administer permission for the server can stop the server. The server inherits settings from the repository ACT (default ACT), so it might not always be essential to add explicit grants.

9 Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the wizard settings.

Use **SAS Management Console to Define a SAS LASR Analytic Server Library**

To register a SAS LASR Analytic Server library, perform the following steps:

1 On the **Plug-ins** tab in SAS Management Console, expand **Data Library Manager**. Right-click **Libraries**, and select **New Library**.

2 In the New Library wizard, select **High-Performance Analytics ➞ SAS LASR Analytic Server Library**. Click **Next**.

3 Enter a name (for example, **Sales LASR**). The **Description** field is optional. **Location** is the metadata folder where the library object is saved. You can keep the default location or change it. Click **Next**.

4 (Optional) Assign the library to one or more SAS Application Servers. Click **Next**.

5 Enter the following library properties:
Table 4.27  Library Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>Enter an identifier of your choice (for example, SALESLIB).</td>
</tr>
<tr>
<td>Engine</td>
<td>This field is not editable. The value SASIOLA is the engine name for a SAS LASR Analytic Server Library.</td>
</tr>
</tbody>
</table>
| Server tag   | If the data for the SAS LASR Analytic Server Library is from co-located HDFS or NFS-mounted MapR, the server tag must be the source path in dot-delimited format.  
  Note: Each server tag must be unique. |
| Data Provider Library | Use the default value (None).                                      |

6 Click Next.

7 Assign the library to a SAS LASR Analytic Server by entering the following settings:

Table 4.28  Connection Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Server</td>
<td>Select a server from the drop-down list.</td>
</tr>
<tr>
<td>Connection</td>
<td>Use the pre-selected value (which prepends the selected server name with the string Connection:).</td>
</tr>
<tr>
<td>Default Login</td>
<td>This field is not editable. The value is (None).</td>
</tr>
</tbody>
</table>

8 Click Next.

9 If you want explicit grants of the Administer permission on the library, move identities from one list to the other. Click Next.

Note: Only users who have the Administer permission for the library can load new tables to memory. The library inherits settings from its parent folder, so it might not be essential to add any explicit grants.

10 Examine the final page of the wizard to ensure that the proper values have been entered. Click Finish to save the wizard settings.
Establishing Connectivity to a MySQL Server

Overview of Establishing Connectivity to MySQL Server

The following instructions are for a MySQL server. The following figure provides a logical view of connecting to a MySQL server and accessing tables from multiple data sources registered there.

*Figure 4.22 Establishing Connectivity to a MySQL Server*

After defining a user or user group, setting up a connection from a SAS session to a MySQL server is a two-stage process:

1. Register the MySQL Server.
2. Register the MySQL Server library.

Register the DBMS Server

The following instructions are for a MySQL server. Follow these steps:

1. Right-click **Server Manager** and select the **New Server** option to access the New Server wizard.
2. Select the MySQL database server from the **Database Servers** list. Click **Next**.
3. Enter an appropriate server name in the **Name** field. **Description** is optional. Click **Next**.
4. Enter the **Major version** and **Minor version** numbers. The **Software version** is the default for the server. The example uses 1.1 release. Enter a value in the **Associated Machine** field. Click **Next**.

5. Enter the appropriate information in the **MySQL Database Information** section. If the **Authentication Information** is incorrect, change it.

*Figure 4.23  MySQL Connection Properties*

If needed, click on **Advanced Options**. Enter the appropriate information. Click **OK**.

*Figure 4.24  MySQL Advanced Options*

Click **Next**.

6. Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the wizard settings.
Stage 2: Register the MySQL Server Library

After you have registered the MySQL server, you can register the MySQL server library.

To register the library, perform the following steps:

1. In SAS Management Console, expand **Data Library Manager**. Then, right-click **Libraries** and select the **New Library** option to access the New Library wizard.

2. Select the **MySQL Library** from the **SAS Data** list. Click **Next**.

3. Enter an appropriate library name in the **Name** field (for example, **MySQL Library**). The **Description** is optional. You can also click **Advanced Options** to perform tasks such as pre-assignment and optimization.

   - **Table 4.29 Library Properties**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>SQLLIB</td>
</tr>
<tr>
<td>Engine</td>
<td>Name of product</td>
</tr>
</tbody>
</table>

   Click **Next**.

4. Select an application server from the list and use the right arrow to assign the application server.

   Click **Next**.

5. Enter the **Server Details** and **Connection Details**. Some of the server and connection information might be entered automatically.

   - **Table 4.30 Library Properties**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Server:</td>
<td>Enter the database server.</td>
</tr>
<tr>
<td>Database Schema Name:</td>
<td>Enter the database schema.</td>
</tr>
<tr>
<td>Connection:</td>
<td>Enter the connection string.</td>
</tr>
<tr>
<td>Default Login:</td>
<td>Enter the default login.</td>
</tr>
</tbody>
</table>

   Click **Next**. Verify that the information is correct.

6. Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the library settings.
Establishing Connectivity to an ODBC Database Using Microsoft Windows NT Authentication

Overview of Establishing Connectivity to an ODBC Database Using Microsoft Windows NT Authentication

The following figure provides a logical view of using ODBC with SAS/ACCESS software.

*Figure 4.25 Establishing Connectivity to an ODBC Database*

This example explains how to define an Open Database Connectivity (ODBC) library in SAS Management Console using Windows authentication to access Microsoft SQL Server data. In this section, the SAS server is running in a Windows operating environment. Once you define the ODBC library, you can use the library with SAS Enterprise Guide and SAS Data Integration Studio.

Prerequisites

The following prerequisites are necessary before you can define an ODBC library in SAS Management Console.

- Make sure that "Login as batch" user rights are assigned for your user ID (entered in the WindowsDomain\user). For details about assigning user rights, see your Windows operating system documentation.
SAS/ACCESS Interface to ODBC and the ODBC driver for the Microsoft SQL Server must be installed and configured on the same machine where your SAS runs.

Setting up a connection from SAS to an ODBC data source using Microsoft Windows NT authentication is a five-stage process.

2. Verify a library by submitting a LIBNAME statement with the ODBC engine.
3. Register a user.
4. Register the ODBC Server.
5. Register the ODBC library.

Stage 1: Configure an ODBC Data Source Using Microsoft Windows NT Authentication

1. Select Start ⇒ Control Panel.

2. In the Control Panel window, select System and Security ⇒ Administrative Tools ⇒ Data Sources (ODBC) to open the ODBC Data Source Administrator dialog box.

3. Click the System DSN tab. On that tab, click Add to open the Create New Data Source dialog box.

4. Select SQL Server in the Create New Data Source dialog box.

5. Click Finish, and the Create a New Data Source to SQL Server dialog box appears. Enter the name of your data source and the name of your server. This example uses the values SqlServerNT and BLUEFIN, respectively.
6 Click Next. On this page, specify the following items:

- the method for verifying the login ID
- the network library that is used to communicate with the SQL Server (via the Client Configuration button)
- a connection to the SQL Server

How should SQL Server verify the authenticity of the login ID?

- With Windows NT authentication using the network login ID.
- With SQL Server authentication using a login ID and password entered by the user.

To change the network library used to communicate with SQL Server, click Client Configuration.

7 Click Next to make further server-configuration changes, as necessary.

8 Click Finish. The application displays a summary page.

At this point, you should click Test Data Source to run a test connection and ensure that your configuration is valid. If the test is successful, the Test Results page is displayed.

9 Click OK to exit.

10 Click OK again to exit the ODBC Data Source Administrator dialog box.
Stage 2: Create a Library By Submitting a LIBNAME Statement with the ODBC Engine

1. Invoke SAS.

2. Submit a LIBNAME statement that contains the ODBC engine in the SAS Enhanced Editor. The LIBNAME statement should be similar to the following:

   ```
   libname odbclib odbc noprompt="dsn=SqlServerNT;Trusted_Connection=yes"
   schema=DBO;
   ```

3. After you submit this LIBNAME statement, make sure that you can see your SQL Server tables in the SAS Explorer window. You can also submit a DATASETS procedure similar to the following to display the tables in the SAS log:

   ```
   proc datasets lib=odbclib;
   quit;
   ```

   The previous steps confirm that the ODBC data source was created correctly and that you can access the tables in the Microsoft SQL Server. If another user has trouble accessing the tables from the registered library (described in the following stages), then ask the user to try these steps or request assistance from your Windows administrator.

Stage 3: Register a User

1. Open SAS Management Console and log on with the user ID or with the ID of any other unrestricted user. The SAS Management Console window is displayed.

2. Right-click User Manager in the left pane. Select New ⇒ User. This selection opens the New User Properties dialog box.

3. Click the General tab in the New User Properties dialog box, and add the user name in the Name text box. Display Name, Job Title, and Description are optional.

4. Click the Accounts tab. Click New to open the New Login Properties dialog box.

5. Add the Windows user ID (in the form WindowsDomain\user) and password in the New Login Properties dialog box. Click OK. The user ID and password appear in the New Users Properties dialog box.

6. Click OK to return to the main SAS Management Console window.

To define a user or user group using SAS Environment Manager, see “Managing User Access” in SAS® Environment Manager User’s Guide.
Stage 4: Register the ODBC Server

1. Right-click **Server Manager** in the SAS Management Console window.

2. Select **New Server** from the menu. This selection invokes the New Server wizard.

3. Select **ODBC Server** from the Database Servers list in the New Server wizard.

4. Click **Next**. Specify a name for the new ODBC server, such as `sql_server_ntauth`. The **Description** field is optional.

5. Click **Next** to continue to the next page, which lists the default property values for the server.

   Note: Do not change these default values. Just click **Next** to continue to the next page in the wizard.

   **Table 4.31 Server Properties**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Major Version Number</td>
<td>6</td>
</tr>
<tr>
<td>Minor Version Number</td>
<td>10</td>
</tr>
<tr>
<td>Data Source Type</td>
<td>ODBC — Other Database</td>
</tr>
<tr>
<td>Software Version</td>
<td>6.10</td>
</tr>
<tr>
<td>Vendor</td>
<td>Defaults to vendor name</td>
</tr>
<tr>
<td>Associated Machine</td>
<td>Defaults to your machine name</td>
</tr>
</tbody>
</table>

6. Click **Next**.

7. Select the **Connection String** radio button.
Enter the following string in the **Connection String** text box:

```
"dsn=SQlServerNT;Trusted_Connection=yes"
```

**Note:** The value for the DSN= parameter is the name of the ODBC-system data source that you set up in stage 1.

Select **None** for **Authentication type**. Select **DefaultAuth** for the **Authentication domain**.

**Note:** If you do not want to create a data source name, you can connect without using the DSN= parameter. Instead, submit the following LIBNAME statement:

```
libname test odbc noprompt='server=bluefin.your-domain.com;driver=sql
server;Trusted Connection=yes';
```

In this case, you enter the following as the connection string in the **Connection String** text box:

```
'server=bluefin.your-domain.com;driver=sql server;
Trusted Connection=yes'
```

8 **Click Next.**

9 Examine the final page of the wizard to ensure that the proper values have been entered. **Click Finish** to save the wizard settings.
Stage 5: Register the ODBC Library

1. Expand Data Library Manager in SAS Management Console.
2. Right-click Libraries and select New Library to open the New Library wizard.
3. Select ODBC Library from the Database Data list. Click Next.
4. Enter the library name (for example, sql_server_ntauth) in the Name field. The Description field is optional. Location is the metadata folder where the library object is saved. You can keep the default location or change it. Click Next.
5. Select the appropriate server from the Available servers list and move it to the Selected servers list. For this example, use SASApp.
6. Click Next.
7. Enter a libref in the Libref text box, such as Odbclib.

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>Odbclib</td>
</tr>
<tr>
<td>Engine</td>
<td>ODBC</td>
</tr>
</tbody>
</table>
8. Click Next.
9. Specify the schema name (DBO) and the database server (sql_server_ntauth).
10. Click Next.
11. Examine the final page of the wizard to ensure that the proper values have been entered. Click Finish to save the wizard settings.

At this point, your library is defined. However, to access data, you might need to perform extra steps (for example, pre-assigning the library or registering tables). See “Registering and Verifying Tables” on page 144.
Establishing Connectivity to a Microsoft Access Database By Using ODBC

Overview of Establishing Connectivity to a Microsoft Access Database By Using ODBC

The following figure provides a logical view of using Microsoft Access as a data source and connecting to the database with a SAS/ACCESS ODBC interface.

Figure 4.26  Establishing Connectivity to Access Databases By Using ODBC

To define a user or user group using SAS Management Console, see “Define the DBMS User” on page 49 and “Define the DBMS User Group” on page 50.

To define a user or user group using SAS Environment Manager, see “Managing User Access” in SAS® Environment Manager User’s Guide.

Setting up a connection from SAS to a Microsoft Access database by using ODBC is a three-stage process:

1. Define an ODBC data source.
2. Register the database server.
3. Register the database library.

This example shows the process for establishing a SAS connection to an Access database. It assumes that the software for the database has already been loaded with the standard installation wizard for the database client. In addition, SAS/ACCESS Interface to ODBC must be installed on the SAS server that accesses the Access database.
Stage 1: Define the ODBC Data Source

First, you must define the ODBC data source. To define the ODBC data source on Windows systems, perform the following steps:

1. Open the Windows Control Panel. Click **Systems and Security** ➔ **Administrative Tools** ➔ **Data Sources (ODBC)** to access the ODBC Data Source Administrator dialog box.

2. Click **Add** to access the Create New Data Source dialog box. Click the Microsoft Access driver listed in the window (for example, **Microsoft Access Driver [*.mdb]**). Click **Finish** to access the ODBC Microsoft Access Setup dialog box.

   **Note:** System data sources and user data sources store information about how to connect to the indicated data provider. A system data source is visible to all users with access to the system, including Windows services. A user data source is visible only to a particular user, and it can be used on the current machine only.

3. Enter the following configuration settings:

   **Table 4.33 Configuration Settings**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>MS Access</td>
</tr>
<tr>
<td>Database</td>
<td>Click <strong>Select</strong> to browse for your Access database file, such as Northwinds.mdb in the Microsoft Office Samples directory.</td>
</tr>
</tbody>
</table>

4. Click **OK** to save the configuration settings and return to the ODBC Data Source Administrator dialog box. Click **OK** to save the data source.

Stage 2: Register the ODBC Server

To register the database server, perform the following steps:

1. Open the SAS Management Console application.

2. Right-click **Server Manager** and select the **New Server** option to access the New Server wizard.

3. Select **ODBC Server** from the **Database Servers** list. Click **Next**.

4. Enter an appropriate server name in the **Name** field (for example, **MS Access Server**). One server is required for each DSN. Note that you can supply an optional description if you want. Click **Next**.

5. Enter the following server properties:
### Table 4.34 Server Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Major Version Number</td>
<td>3</td>
</tr>
<tr>
<td>Minor Version Number</td>
<td>7</td>
</tr>
<tr>
<td>Data Source Type</td>
<td>ODBC - Microsoft Access</td>
</tr>
<tr>
<td>Software Version</td>
<td>3.7.0</td>
</tr>
<tr>
<td>Vendor</td>
<td>Microsoft</td>
</tr>
<tr>
<td>Associated Machine</td>
<td>newserver.na.sas.com</td>
</tr>
</tbody>
</table>

This is the server where the database is running. (Select this value from the drop-down list. If the value that you need is not available, click **New** to access the New Machine dialog box. Then enter the appropriate value in the **Host Name** field.)

Click **Next**.

#### 6 Enter the following connection properties:

### Table 4.35 Connection Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datasrc</td>
<td><em>MS Access</em> (Use the value entered in the <strong>Data Source Name</strong> field in the ODBC Data Source Administrator dialog box.)</td>
</tr>
<tr>
<td>Authentication type</td>
<td>User/Password</td>
</tr>
<tr>
<td>Authentication domain</td>
<td>ODBCAuth (You might need to create a new authentication domain. For more information, see “How to Store Passwords for a Third-Party Server” in SAS Intelligence Platform: Security Administration Guide.) Click <strong>New</strong> to access the New Authentication Domain dialog box. Then enter the appropriate value in the <strong>Name</strong> field and click <strong>OK</strong> to save the setting.</td>
</tr>
</tbody>
</table>

Click **Next**.

#### 7 Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the wizard settings.

## Stage 3: Register the ODBC Library

After you have registered the database server, you can register the database library. To register the database library, perform the following steps:
1. In SAS Management Console, expand Data Library Manager. Then, right-click Libraries and select the New Library option to access the New Library wizard.

2. Select ODBC Library from the Database Data list. Click Next.

3. Enter an appropriate library name in the Name field (for example, MS Access Library). The Description field is optional. Location is the metadata folder where the library object is saved. You can keep the default location or change it. Click Next.

4. Select an application server from the list and use the right arrow to assign the application server. Click Next.

5. Enter the following library properties:

   Table 4.36  Library Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>ACCESREF</td>
</tr>
<tr>
<td>Engine</td>
<td>ODBC</td>
</tr>
</tbody>
</table>

   You can also click Advanced Options to perform tasks such as pre-assignment and optimization. Click Next to access the next page of the wizard.

6. Enter the following settings:

   Table 4.37  Server and Connection Information

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Server</td>
<td>MS Access Server (Use the database server that you created in the New Server wizard.)</td>
</tr>
</tbody>
</table>

   Click Next.

7. Examine the final page of the wizard to ensure that the proper values have been entered. Click Finish to save the library settings. At this point, register tables as explained in “Registering and Verifying Tables” on page 144.
Establishing Connectivity to an OLE DB Database Using Microsoft Windows NT Authentication

Overview of Establishing Connectivity to an OLE DB Database Using Microsoft Windows NT Authentication

The following figure provides a logical view of using OLE DB with SAS/ACCESS software.

**Figure 4.27  Establishing Connectivity to an OLE DB Database**

This section explains how to define an OLE DB library in SAS Management Console in order to access the Microsoft SQL Server using Microsoft Windows NT authentication.

Prerequisites

The following prerequisites are necessary before you begin defining an OLE DB library:

- Make sure that “Login as batch” rights are assigned for your user ID (entered in the WindowsDomain\user). For details about assigning user rights, see your Windows operating system documentation.

- SAS/ACCESS Interface to OLE DB and the OLE DB provider for the Microsoft SQL Server must be installed on the same machine where SAS runs.
Before you configure the library in SAS Management Console, make sure that you can submit a LIBNAME statement successfully. Here is an example of a LIBNAME statement:

```hs
libname sqlole oledb
   init_string="Provider=SQLOLEDB.1;
   Integrated Security=SSPI;
   Persist Security Info=True;
   Initial Catalog=Northwind;
   Data Source=bluefin.na.sas.com" schema=DBO;
```

Note: If you cannot see SQL Server tables in the SAS Explorer window, then you need to submit the following SQL procedure to find the table names and the corresponding schema for each table.

```sql
proc sql;
   connect to oledb;
   select table_name, table_schema from connection
to oledb(OLEDB::Tables);
quit;
```

Stage 1: Register the OLE DB Server

To define a user or user group using SAS Management Console, see “Define the DBMS User” on page 49 and “Define the DBMS User Group” on page 50.

To define a user or user group using SAS Environment Manager, see “Managing User Access” in SAS® Environment Manager User’s Guide.

Setting up a connection from SAS to an OLE DB database using Microsoft Windows NT authentication is a four-stage process.

1 Register the OLE DB server.
2 Register the OLE DB library.
3 Verify that the SQL Server tables are listed in the SAS Explorer.
4 Register the SQL Server tables.
Table 4.38  Server Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Major Version Number</td>
<td>4</td>
</tr>
<tr>
<td>Minor Version Number</td>
<td>0</td>
</tr>
<tr>
<td>Data Source Type</td>
<td>OLE DB — Other Database</td>
</tr>
<tr>
<td>Software Version</td>
<td>4.0</td>
</tr>
<tr>
<td>Vendor</td>
<td>Microsoft Corp.</td>
</tr>
<tr>
<td>Associated Machine</td>
<td>Defaults to your machine</td>
</tr>
</tbody>
</table>

Note: The Associated Machine property refers to the SAS Application Server - not the SQL Server machine.

6 Click Next.

7 Enter the server name "bluefin.na.sas.com" (in quotation marks) for the SQL Server in the Datasource field. Enter the Provider.

Table 4.39  Connection Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datasource</td>
<td>Enter &quot;bluefin.na.sas.com&quot; (in quotation marks) from the example LIBNAME statement.</td>
</tr>
<tr>
<td>Provider</td>
<td>Enter SQLOLEDB.1 from the example LIBNAME statement.</td>
</tr>
<tr>
<td>Prompt</td>
<td>The default is NO.</td>
</tr>
<tr>
<td>OLE DB Connection</td>
<td></td>
</tr>
<tr>
<td>Information Options</td>
<td></td>
</tr>
</tbody>
</table>
| Authentication type    | User/Password
Specify the credentials (in this example, user and password) that are retrieved from metadata and that correspond to the selected authentication domain, OledbAuth. |
| Authentication Domain  | OledbAuth
Specify the domain (in this example, OledbAuth) that is used to authenticate logon attempts to the OLE DB server. (You might need to create a new authentication domain. For more information, see "How to Store Passwords for a Third-Party Server" in SAS Intelligence Platform: Security Administration Guide.) Click New to access the New Authentication Domain dialog box. Then enter the appropriate value in the Name field and click OK to save the setting. |
Click the **OLE DB Connection Information Options** button to display the OLE DB Connection Information Options dialog box.

8 Enter the following text in the **Init String** field. Enter the entire string in quotation marks (" "), all on one line.

*Provider=SQLOLEDB.1;*
Integrated Security=SSPI;
Persist Security Info=True;
Initial Catalog=Northwind;
Data Source=bluefin.na.sas.com"

9 Click **OK**.

10 Specify **Authentication Type** and **Authentication Domain**.

11 Click **Next**.

12 Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the wizard settings.

13 In SAS Management Console, click the **Server Manager** icon.

The new OLE DB server name, *sqlserver_oledb_ntauth*, appears in the Name list.

---

**Stage 2: Register the OLE DB Library**

1 From the navigation tree in SAS Management Console, expand **Data Library Manager**.

2 Under **Data Library Manager**, right-click **Libraries** and select **New Library** to start the New Library wizard.

3 In the wizard, expand the **Libraries** ⇒ **Database Data** folder and select **OLE DB Library**. Click **Next**.

4 Enter the library name (for example, *sqlserver_oledb_ntauth_lib*) in the **Name** field. The **Description** field is optional. **Location** is the metadata folder where the library object is saved. You can keep the default location or change it.

5 Click **Next**.
6. Select the appropriate server from the **Available servers** list and move it to the **Selected servers** list on the right. For this example, use **SASApp**. Click **Next**.

7. Enter the libref name (for example, **SqlOledb**) in the **Libref** field.

   **Table 4.40 Library Properties**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>SqlOledb</td>
</tr>
<tr>
<td>Engine</td>
<td>OLEDB</td>
</tr>
</tbody>
</table>

8. Click **Next**.

9. In the **Database Server** field, select the OLE DB server (**sqlserver_oledb_ntauth**).

   In the **Database Schema Name** field, enter a name (for example, **DBO**) for the new OLE DB schema.

   **Note:** This is the same database schema name specified in the LIBNAME statement:

   ```
   libname sqlole oledb init_string="Provider=SQLOLEDB.1; Integrated Security=SSPI; Persist Security Info=True; Initial Catalog=Northwind; Data Source=bluefin.na.sas.com" schema=DBO;
   ```

   **Note that the database schema name must be uppercase in the LIBNAME statement.**

   **Table 4.41 Server and Connection Information**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Server</td>
<td>Enter sqlserver_oledb_ntauth from the LIBNAME statement example. (Use the database server that you created in the New Server wizard.)</td>
</tr>
<tr>
<td>Database Schema Name</td>
<td>Enter DBO from the LIBNAME statement example.</td>
</tr>
<tr>
<td>Connection</td>
<td>Use the default value.</td>
</tr>
<tr>
<td>Default Login</td>
<td>Use the default value of (<strong>None</strong>)</td>
</tr>
</tbody>
</table>

   Click **Next**.

10. Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the wizard settings. The new library (**sqlserver_oledb_ntauth_lib**) appears in the list of libraries.
Stage 3: Verify That the SQL Server Tables Are Listed in the SAS Explorer

Perform the following steps to check the accuracy of the LIBNAME statement that was generated in the previous section:

1. Close SAS Management Console and restart it using the same Windows ID (WindowsDomain\user) that you previously used.
2. From the navigation tree in SAS Management Console, right-click Data Library Manager.
3. Under Data Library Manager, right-click the library that you just created (sqlserver_oledb_ntauth_lib) and select Display LIBNAME Statement. The Display LIBNAME Statement message window appears with the LIBNAME statement that is generated for your server.
4. Copy the LIBNAME statement from the Display LIBNAME Statement window and paste it into your SAS Program Editor.
5. Submit the LIBNAME statement and verify that the appropriate tables are listed in the SAS Explorer window.

Stage 4: Register the SQL Server Tables

1. From the navigation tree in SAS Management Console, expand Data Library Manager.
2. In the Data Library Manager, right-click sqlserver_oledb_ntauth_lib and select Register Tables.
3. In the SAS Library field, click the down arrow at the right of the field. From the resulting drop-down list, select your application server (SASApp, in this example), and click Next.
4. Click Next again, and the Define Tables dialog box is displayed with a list of SQL Server tables.
5. In the list, click the table name for which you want to import metadata.
6. Click Next, and the application displays the table that you selected.
7. Click Finish, and you are able to use the specified table in any SAS BI client.

For more information, see “Relational Database Sources” on page 116.
Establishing Connectivity to an Oracle Database

Overview of Establishing Connectivity to an Oracle Database

The following figure provides a logical view of using Oracle as a data source.

Figure 4.28 Establishing Connectivity to an Oracle Database

This example shows the process for establishing a SAS connection to an Oracle database. It assumes that the software for the database has already been loaded by using the standard installation wizard for the database client.

Prerequisites

The following prerequisites are necessary before you begin defining an Oracle library:

- installation of SAS/ACCESS Interface to Oracle. For configuration information, see SAS 9.4 Install Center Documentation and use the operating system and SAS version to locate the appropriate SAS Foundation Configuration Guide.
- installation of a supported Oracle Database Client.
- validation that the Oracle client can communicate with the Oracle server.
- (UNIX only) configuration of SAS/ACCESS environmental variables. For more information, see “Setting UNIX Environment Variables for SAS/ACCESS” on page 147.

Submit a LIBNAME statement, such as the following example:
Stage 1: Register the Oracle Server

To register the Oracle database server, perform the following steps:

1. Open SAS Management Console application.
2. Right-click **Server Manager** and select the **New Server** option to access the New Server wizard.
3. Select **Oracle Server** from the **Database Servers** list. Click **Next**.
4. Enter an appropriate server name in the **Name** field (for example, `oracle_server_on_alien`). You can supply an optional description. Click **Next**.
5. The server properties that are displayed in the window are default values and should not be changed. To change the **Associated Machine** property, click the down arrow at the right of the field and select the appropriate server from the drop-down list.

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Major Version Number</td>
<td>0</td>
</tr>
<tr>
<td>Minor Version Number</td>
<td>0</td>
</tr>
<tr>
<td><strong>Software Version</strong></td>
<td></td>
</tr>
<tr>
<td>Vendor</td>
<td>Oracle Corporation</td>
</tr>
<tr>
<td>Associated Machine</td>
<td>Defaults to your machine</td>
</tr>
</tbody>
</table>

Click **Next**.

6. Enter the following connection properties:
Table 4.43  Connection Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>NEWSERVER10G (This value is contained in the tnsnames.ora file generated during the Oracle installation. The file is stored in an Oracle installation directory such as /opt/oracle/app/oracle/product/10.2.0/db_1/network/admin/tnsnames.ora. The alias for the connection information is contained in this file. See the following figure.)</td>
</tr>
<tr>
<td>Authentication type</td>
<td>The default is User/Password. Select a value from the drop-down list.</td>
</tr>
<tr>
<td>Authentication domain</td>
<td>OraAuth (You might need to create a new authentication domain. For more information, see &quot;How to Store Passwords for a Third-Party Server&quot; in SAS Intelligence Platform: Security Administration Guide.) Click New to access the New Authentication Domain dialog box. Then enter the appropriate value in the Name field and click OK to save the setting.</td>
</tr>
</tbody>
</table>

7 Examine the final page of the wizard to ensure that the proper values have been entered. Click Finish to save the wizard settings.

Stage 2: Register the Oracle Library

After you have registered the database server, register the database library. To register the Oracle database library, perform the following steps:

1 In SAS Management Console, expand Data Library Manager. Right-click Libraries. Then, select the New Library option to access the New Library wizard.

2 Select Oracle Library from the Database Data list. Click Next.

3 Enter an appropriate library name in the Name field (for example, oracle_lib_on_alien). The Description field is optional. Location is the metadata folder where the library object is saved. You can keep the default location or change it. Click Next.

4 Select a SAS server from the list and use the right arrow to assign the SAS server. This step makes the library available to the server and makes the library visible to users of the server. This step is optional. Click Next.

5 Enter the following library properties:

Table 4.44  Library Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>ORAREF</td>
</tr>
</tbody>
</table>
You can also click **Advanced Options** to perform tasks such as pre-assignment and optimization. Click **Next** to access the next page of the wizard.

6 Enter the following settings:

**Table 4.45 Server and Connection Information**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engine</td>
<td>ORACLE</td>
</tr>
<tr>
<td><strong>Database Server</strong></td>
<td>oracle_server_on_alien (Use the database server that you created in the New Server wizard.)</td>
</tr>
<tr>
<td><strong>Database Schema Name</strong></td>
<td>See your database administrator for the correct value. Note: Enter the schema name using the exact casing that is used in the database schema name.</td>
</tr>
<tr>
<td><strong>Default Login</strong></td>
<td>Use the default value of <em>(None)</em>.</td>
</tr>
</tbody>
</table>

Click **Next**.

7 Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the library settings. At this point, register tables as explained in “Registering and Verifying Tables” on page 144.

---

**Establishing Connectivity to a PI System Server**

**Overview of Establishing Connectivity to a PI System Server**

The following instructions are for a PI System server. The following figure provides a logical view of connecting to a PI System server and accessing tables from multiple data sources registered there.
After you define a user or user group using SAS Management Console, use this two-stage process to set up a connection from a SAS session to a PI System server:

1. Register the PI System server.
2. Register the PI System server library.

Register the DBMS Server

The following instructions are for a PI System server.

1. Right-click **Server Manager** and select the **New Server** option to access the New Server wizard.
2. Select the PI System server from the **Database Servers** list. Click **Next**.
3. Enter an appropriate server name in the **Name** field. **Description** is optional. Click **Next**.
4. Enter the **Major version** and **Minor version** numbers. The **Software version** is the default for the server. The example uses the 1.1 release. Enter the **Associated machine** value. Click **Next**.
5. Enter the **PI System server name** and the **PI System Web API REST host**. Click **Next**.
Stage 2: Register the PI System Server Library

After you have registered the PI System server, you can register the PI System server library.

1. In SAS Management Console, expand **Data Library Manager**. Then, right-click **Libraries** and select the **New Library** option to access the New Library wizard.

2. Select the PI System **Library** from the **SAS Data** list. Click **Next**.

3. Enter an appropriate library name in the **Name** field (for example, **PI System Library**). The **Description** is optional. **Database** is the metadata folder where the library object is saved. You can keep the default location or change it. Click **Next**.

4. Select an application server from the list and use the right arrow to assign the application server. Click **Next**.

6. Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the wizard settings.
5 Table 4.46 Library Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>PISYSLIB</td>
</tr>
<tr>
<td>Engine</td>
<td>Name of product</td>
</tr>
<tr>
<td>Database</td>
<td>Enter the metadata database name.</td>
</tr>
</tbody>
</table>

You can also click Advanced Options to perform tasks such as pre-assignment and optimization. Click Next to access the next page of the wizard.

6 Enter the Server Details and Connection Details. Some of the server and connection information might be entered automatically.

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Server:</td>
<td>Enter the database server.</td>
</tr>
<tr>
<td>Database Schema Name:</td>
<td>Enter the database schema.</td>
</tr>
<tr>
<td>Connection:</td>
<td>Enter the connection string.</td>
</tr>
<tr>
<td>Default Login:</td>
<td>Enter the default login.</td>
</tr>
</tbody>
</table>

Click Next. Verify that the information is correct.

7 Examine the final page of the wizard to ensure that the proper values have been entered. Click Finish to save the library settings.

Establishing Connectivity to a Redshift Server

Overview of Establishing Connectivity to Redshift

The following figure provides a logical view of connecting to a Redshift Server.
To define a user or user group using SAS Management Console, see “Define the DBMS User” on page 49 and “Define the DBMS User Group” on page 50.

To define a user or user group using SAS Environment Manager, see “Managing User Access” in SAS® Environment Manager User’s Guide.

Setting up a connection from SAS to a Redshift Server is a two-stage process:

1. Register the Redshift server.
2. Register the Redshift library.

Stage 1: Register the Redshift Server

To register the Redshift server, perform the following steps:

1. Open SAS Management Console application.
2. Right-click Server Manager, and select the New Server option to access the New Server wizard.
3. Select Redshift Server from the Database Servers list. Click Next.
4. Specify a name for the new Redshift server (for example, RedshiftServer). The description is optional.
5. Enter the following server properties. A Redshift 1.0 installation is used as the example:

Table 4.48 Server Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Major Version Number</td>
<td>1</td>
</tr>
<tr>
<td>Minor Version Number</td>
<td>0</td>
</tr>
<tr>
<td>Software Version</td>
<td>1.0</td>
</tr>
</tbody>
</table>
Establishing Connectivity to a Redshift Server

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vendor</td>
<td>Amazon</td>
</tr>
<tr>
<td>Associated Machine</td>
<td>Defaults to your machine, Click New to specify a different machine.</td>
</tr>
</tbody>
</table>

Click **Next**.

6 On this page, enter the connection properties. Click the **Server Information** button. Click **Options**. For this example, enter the following connection properties:

*Figure 4.31  Redshift Server Options*

Click **OK**.

*Table 4.49  Connection Properties*

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server Information:</td>
<td></td>
</tr>
<tr>
<td>ODBC datasource</td>
<td></td>
</tr>
<tr>
<td>Information:</td>
<td></td>
</tr>
<tr>
<td>Authentication Type</td>
<td>The default is User/Password. Select a value from the drop-down list.</td>
</tr>
<tr>
<td>Authentication Domain</td>
<td>DefaultAuth (You might need to create a new authentication domain. For more information, see &quot;How to Store Passwords for a Third-Party Server&quot; in SAS Intelligence Platform: Security Administration Guide.) Click New to access the New Authentication Domain dialog box. Enter the appropriate value in the Name field, and click OK to save the setting.</td>
</tr>
</tbody>
</table>

7 Click **Next**.

8 Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the wizard settings.
Stage 2: Register the Redshift Library

After you have registered the server, you can register the library. To register the library, perform the following steps:

1. In SAS Management Console, expand Data Library Manager. Right-click Libraries. Then, select the New Library option to access the New Library wizard.

2. Select Redshift Library from the Database Data list. Click Next.

3. Enter an appropriate library name in the Name field (for example, Redshift Library). The Description field is optional. Location is the metadata folder where the library object is saved. You can keep the default location or change it. Click Next.

4. Select an application server from the list and use the right arrow to assign the application server. This step makes the library available to the server and makes the library visible to users of the server. Click Next.

5. Enter the following library properties:

   Table 4.50  Library Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>RedLib</td>
</tr>
<tr>
<td>Engine</td>
<td>SASIORST (Accept the value that is populated automatically.)</td>
</tr>
</tbody>
</table>

   Click Next.

6. Select the Redshift server that you entered in the Database Server field of the New Server wizard (for example, RedshiftServer) by using the Database Server drop-down list. Click Next.

7. Examine the final page of the wizard to ensure that the proper values have been entered. Click Finish to save the library settings.

Establishing Connectivity to an SAP Server

Overview to Establishing Connectivity to an SAP Server

The following figure provides a logical view of connecting to an SAP Server as a data source.
Stage 1: Register the SAP Server

To register the SAP server, perform the following steps:

1. Open the SAS Management Console application.
2. Right-click Server Manager and select the New Server option to access the New Server wizard.
3. Select SAP Server from the Enterprise Applications Servers list. Click Next.
4. Enter an appropriate server name in the Name field (for example, SAPServer). Note that you can supply an optional description if you want. Click Next.
5. Enter the following server properties. An SAP 4.6 installation is used as the example:
Table 4.51  Server Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Major Version Number</td>
<td>4</td>
</tr>
<tr>
<td>Minor Version Number</td>
<td>6</td>
</tr>
<tr>
<td>Software Version</td>
<td>4.6</td>
</tr>
<tr>
<td>Vendor</td>
<td>SAP AG</td>
</tr>
</tbody>
</table>

Click Next.

6 Enter the following connection properties:

Table 4.52  Connection Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication Domain</td>
<td>SAPAuth (You might need to create a new authentication domain. For more information, see &quot;How to Store Passwords for a Third-Party Server&quot; in SAS Intelligence Platform: Security Administration Guide.) Click New to access the New Authentication Domain dialog box. Then enter the appropriate value in the Name field and click OK to save the setting.</td>
</tr>
<tr>
<td>Client</td>
<td>800 (This value is obtained from your SAP administrator.)</td>
</tr>
<tr>
<td>Language</td>
<td>EN (This value is obtained from your SAP administrator.)</td>
</tr>
</tbody>
</table>

Note: An embedded RFC server is not available for z/OS. For z/OS, click the Advanced Options button and enter "host=rfc-server port=rfc-port" in the Other option(s) to be appended text field. Also, select the Batch Mode check box.

7 Select Application Server and click Options to access the Application Server Host dialog box.

TIP Instead of the application server, you might choose other options, as well, including: SAPGUI Logical Name, SAPRFC.INI Logical Name, and Message Servers.

8 Enter the fully qualified name of the server host that was supplied by the SAP administrator (for example, sapsrv.na.sas.com) in the Application Server Host field. Enter the system number that was supplied by the SAP administrator (for example, 12) in the System Number field. The default access mode is direct access. In order to run in batch mode, click the Advanced Options tab. Select the Batch Mode check box. In the Other options field, enter a value for destgroup such as destgroup="SDSTEST". For batch mode on z/OS, follow the instructions in Installation Instructions for SAS/ACCESS Interface to R/3 Software. Click OK to return to the New Server wizard.
9 Click Next.

10 Examine the final page of the wizard to ensure that the proper values have been entered. Click Finish to save the wizard settings.

Stage 2: Register the SAP Library

After you have registered the server, you can register the library. To register the library, perform the following steps:

1 In SAS Management Console, expand Data Library Manager. Right-click Libraries. Then, select the New Library option to access the New Library wizard.

2 Select SAP Library from the Enterprise Applications Data list. Click Next.

3 Enter an appropriate library name in the Name field (for example, SAP Library). The Description field is optional. Location is the metadata folder where the library object is saved. You can keep the default location or change it. Click Next.

4 Select an application server from the list and use the right arrow to assign the application server. This step makes the library available to the server and makes the library visible to users of the server. Click Next.

5 Enter the following library properties:

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>SAPLib</td>
</tr>
<tr>
<td>Engine</td>
<td>SASIOSR3 (Accept the value that is populated automatically.)</td>
</tr>
</tbody>
</table>

Click Next.

6 Select the SAP server that you entered in the Name field of the New Server wizard (for example, SAP Server) by using the Database Server drop-down list. Click Next.

7 Examine the final page of the wizard to ensure that the proper values have been entered. Click Finish to save the library settings.

Stage 3: Extract SAP Metadata

If SAS Data Surveyor for SAP is installed, then you can extract metadata about your SAP objects to SAS data sets. Once you have created the SAS data sets, the tables in your SAP System are available for use in jobs with clients like SAS Data Integration Studio and SAS Enterprise Guide.
The tools for extracting the SAP metadata are provided as a plug-in to SAS Management Console and access to the tool is controlled with role-based access. To enable the extraction tool for role-based access and to extract the SAP metadata, perform the following steps:

1. Using an unrestricted account, select Tools ➡ Plug-in Manager From the SAS Management Console.

2. On the Plug-in Manager window, select the ExtractionTool check box. Click OK.

3. Assign the ExtractionTool capability to a role with the User Manager plug-in to SAS Management Console, and then associate users or groups with the role. The following list provides two choices:
   - Assign the ExtractionTool capability to an existing role such as Management Console: Advanced.
   - Create a new role, assign the ExtractTool capability to it, and then associate users and groups with the new role.

For more information about roles, see “Change a Role’s Capabilities” in SAS Management Console: Guide to Users and Permissions.

4. Important: Log on to SAS Management Console with an account that has access to the Foundation repository and that is not an unrestricted account.

5. Select Tools ➡ Extract from BW or Tools ➡ Extract from R/3. For information about using the tools, click Help.

Special Considerations for SAP

For z/OS operating environments, when you specify the language value on the New Server wizard, use uppercase letters and enclose the value in quotation marks (for example, “EN”).

Establishing Connectivity to an SAP HANA Server

Overview to Establishing Connectivity to an SAP HANA Server

The following figure provides a logical view of connecting to an SAP HANA Server as a data source.
To define a database library in SAS Management Console with the SAS/ACCESS software, you need SAS/ACCESS Interface to SAP HANA software installed on your SAS server. After you verify the system requirements for the SAS/ACCESS Interface to SAP HANA, test the database connection using a LIBNAME statement:

```
libname A1 sasiohna server=server-name port=number user=user-id password=password;
```

This example does not specify a DSN style. SERVER=, DATABASE=, USER=, and PASSWORD= are connection options. This is the default method, which is recommended.

```
libname A1 sasiohna server=mysrv1 port=30015 user=myusr1 password='mypwd1';
```

This example requires that you specify a DSN style.

```
libname B1 sasiohna dsn=hnatest user=myusr1 password=mypwd1;
```

After you assign the LIBNAME statement, you can view your SAP HANA tables in the A1 library either by viewing it from the SAS Explorer window or by submitting a DATASETS procedure that is similar to the following:

```
proc datasets lib=A1;
quit;
```

To define a user or user group using SAS Management Console, see "Define the DBMS User" on page 49 and "Define the DBMS User Group" on page 50.

To define a user or user group using SAS Environment Manager, see "Managing User Access" in SAS® Environment Manager User’s Guide.

Setting up a connection from SAS to an SAP HANA server is a two-stage process:

1. Register the server.
2. Register the library.

This example shows the process for establishing a SAS connection to an SAP HANA data source. It assumes that SAS/ACCESS Interface to SAP HANA software has already been installed.
Stage 1: Register the SAP HANA Server

To register the SAP HANA server, perform the following steps:

1. Open SAS Management Console application.
2. Right-click Server Manager and select the New Server option to access the New Server wizard.
3. Select SAP HANA Server from the Database Servers list. Click Next.
4. Click Next. Select the appropriate server from the Available servers list and move it to the Selected servers list. For this example, use SASApp.
5. Enter the following server properties. An SAPHANA 1.0 installation is used as the example:

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Major Version Number</td>
<td>1</td>
</tr>
<tr>
<td>Minor Version Number</td>
<td>0</td>
</tr>
<tr>
<td>Software Version</td>
<td>1.0</td>
</tr>
<tr>
<td>Vendor</td>
<td>SAP AG</td>
</tr>
<tr>
<td>Associated Machine</td>
<td>Defaults to your machine, Click New to specify a different machine.</td>
</tr>
</tbody>
</table>

Click Next.

6. Enter the following connection properties:

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server Information</td>
<td>You are required to select at least one field. Enter the appropriate information if you choose ODBC Datasource or Connection String. If you choose Server Information, then click Options. See Server Information Options on page 133. Enter the connection information for your SAP HANA instance. There are several ways of specifying the connection information. In the Server information window, you can specify server and port, or the server and instance information. Or you can enter the ODBC data source, or a connection string. Click OK.</td>
</tr>
<tr>
<td>Authentication Type</td>
<td>The default is User/Password. Select a value from the drop-down list.</td>
</tr>
</tbody>
</table>
Field | Sample Value
--- | ---
Authentication Domain | DefaultAuth (You might need to create a new authentication domain. For more information, see “How to Store Passwords for a Third-Party Server” in SAS Intelligence Platform: Security Administration Guide.) Click New to access the New Authentication Domain dialog box. Then enter the appropriate value in the Name field and click OK to save the setting.

Enter `mysvr1` in the **Server** field. The **Driver** field is optional. Click **OK**.

7 Click **Next**.

8 Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the wizard settings.

---

Stage 2: Register the SAP HANA Library

After you have registered the server, you can register the library.

To register the library, perform the following steps:

1 In SAS Management Console, expand **Data Library Manager**. Right-click **Libraries**. Then, select the **New Library** option to access the New Library wizard.

2 Select **SAP HANA Library** from the **Database Data** list. Click **Next**.

3 Enter an appropriate library name in the **Name** field (for example, **SAP HANA Library**). The **Description** field is optional. **Location** is the metadata folder where the library object is saved. You can keep the default location or change it. Click **Next**.

4 Select an application server from the list and use the right arrow to assign the application server. This step makes the library available to the server and makes the library visible to users of the server. Click **Next**.
5 Enter the following library properties:

Table 4.56  Library Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>SHLibref</td>
</tr>
<tr>
<td>Engine</td>
<td>SASIOHNA (Accept the value that is populated automatically.)</td>
</tr>
</tbody>
</table>

Click Next.

6 Select the SAP HANA server that you entered in the Name field of the New Server wizard (for example, SAS HANA Server) by using the Database Server drop-down list. Click Next.

7 Examine the final page of the wizard to ensure that the proper values have been entered. Click Finish to save the library settings.

---

Establishing Connectivity to a SAS Scalable Performance Data Server

Overview of Establishing Connectivity to a SAS Scalable Performance Data Server

The following figure provides a logical view of using SAS Scalable Performance Data (SPD) Server tables as a data source.

Figure 4.34  Establishing Connectivity to an SPD Server
To define a user or user group using SAS Management Console, see “Define the DBMS User” on page 49 and “Define the DBMS User Group” on page 50.

To define a user or user group using SAS Environment Manager, see “Managing User Access” in SAS® Environment Manager User’s Guide.

After defining a user or user group, configuring a connection from SAS to an SPD Server is a three-stage process:

1. Configure the libnames.parm file.
2. Register the SPD Server.
3. Register the SPD Server library.

This example shows the process for establishing a SAS connection to SPD Server. It assumes that the software for the database has already been loaded by using the standard installation wizard for the database client. The SPD Server client and server software must be installed before the connection can be established.

Stage 1: Configure the libnames.parm File

When you install the SPD Server software on Windows, a libnames.parm file is created in the C:\Program Files\SAS Institute Inc\SPDS-version\Site directory. You must specify at least a LIBNAME and a pathname for the directory to use for the SPD Server tables (for example, C:\SPDSTables). For the LIBNAME, use the LIBNAME domain that you created earlier for the library (in this case, spdsrv).

A sample libnames.parm file is shown in the following figure:

![libnames.parm - Notepad](image)

Stage 2: Register the SAS Scalable Performance Data Server

To register the database server, perform the following steps:
1. Open the SAS Management Console application.

2. Right-click **Server Manager** and select the **New Server** option to access the New Server wizard.

3. Select **SAS Scalable Performance Data Server** from the **SAS Servers** list. Click **Next**.

4. Enter an appropriate server name in the **Name** field (for example, *SPDServer*). You can supply an optional description. Click **Next**.

5. Enter the following server properties:

   **Table 4.57 Server Properties**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Major Version Number</td>
<td>4</td>
</tr>
<tr>
<td>Minor Version Number</td>
<td>3</td>
</tr>
<tr>
<td>Vendor</td>
<td>SAS Institute</td>
</tr>
<tr>
<td>SAS Compatibility</td>
<td>SAS 9</td>
</tr>
</tbody>
</table>

   Click **Next**.

6. Enter the following connection properties:

   **Table 4.58 Connection Properties**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>D1234</td>
</tr>
<tr>
<td>Port Number or Name</td>
<td>5200 (Enter the port number for the SPD Server name server.)</td>
</tr>
<tr>
<td>Communication Protocol</td>
<td>TCP</td>
</tr>
<tr>
<td>Authentication Domain</td>
<td><em>SPDServer</em></td>
</tr>
</tbody>
</table>

   (You might need to create a new authentication domain. For more information, see “How to Store Passwords for a Third-Party Server” in SAS Intelligence Platform: Security Administration Guide.) Click **New** to access the New Authentication Domain dialog box. Then enter the appropriate value in the **Name** field and click **OK** to save the setting.

7. Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the wizard settings.
Stage 3: Register the SAS Scalable Performance Data Server Library

After you have registered the server, you can register the library. To register the library, perform the following steps:

1. In SAS Management Console, expand Data Library Manager. Right-click Libraries. Then, select the New Library option to access the New Library wizard.

2. Select SAS Scalable Performance Data Server Library from the SAS Data list. Click Next.

3. Enter an appropriate library name in the Name field (for example, SPDServerLibrary). The Description field is optional. Location is the metadata folder where the library object is saved. You can keep the default location or change it. Click Next.

4. Select a SAS server from the list and use the right arrow to assign the SAS server. This step makes the library available to the server and makes the library visible to users of the server. Click Next.

5. Enter the following library properties:

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>spdsrv</td>
</tr>
<tr>
<td>Engine</td>
<td>SASSPDS</td>
</tr>
</tbody>
</table>

   You can also click Advanced Options to perform tasks such as pre-assignment and optimization. Click Next to access the next page of the wizard.

6. Enter the following settings:

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAS SPD Server</td>
<td>SPDSServer   (Use the database server that you selected in the New Server wizard.)</td>
</tr>
<tr>
<td>LIBNAME Domain</td>
<td>spdsrv       (Select the domain name that you entered in the libname.parms file.)</td>
</tr>
<tr>
<td>Default Login</td>
<td>(None)       (Keep this default value.)</td>
</tr>
</tbody>
</table>

   Click Next.
Establishing Connectivity to a SASHDAT Library

Overview of Establishing Connectivity to a SASHDAT Library

The following figure provides a logical view of connecting to a SASHDAT library (sometimes referred to as SAS Data in HDFS library).

**Figure 4.35  Establishing Connectivity to an HDFS Library**

This library works only with SASHDAT files that are created with the OLIPHANT procedure or with the SASHDAT Engine. SASHDAT is the data format used for SAS tables that are added to HDFS.

The main reason for creating a new library of the type SASHDAT is to support an additional HDFS directory. For each HDFS directory, you need a designated HDFS library (and a corresponding LASR library).

Register a SASHDAT Library

To register a SASHDAT library, perform the following steps:

1. On the **Plug-ins** tab in SAS Management Console, expand **Data Library Manager**. Right-click **Libraries**, and select **New Library**.
2 In the New Library wizard, select High-Performance Analytics SASHDAT Library. Click Next.

3 Enter an appropriate library name in the Name field. The Description field is optional. Location is the metadata folder where the library object is saved. You can keep the default location or change it. Click Next.

4 Select the appropriate server from the Available servers list and move it to the Selected servers list. For this example, use SASApp. Click Next.

5 In the second quarter 2015, the Encryption Options section was added to SAS Management Console.

Enter the following library properties:

Table 4.61 Library Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>Enter an identifier of your choice (for example, SALESLIB).</td>
</tr>
<tr>
<td>Engine</td>
<td>This field is not editable. The value, SASHDAT, is the engine name for libraries of the type SASHDAT.</td>
</tr>
<tr>
<td>HDFS path</td>
<td>Enter the HDFS source path (for example, /dept/sales).</td>
</tr>
<tr>
<td></td>
<td>In the LASR library that is paired with this library, the server tag must be the HDFS source path in dot-delimited format. For example, if the path is /dept/sales, then the server tag is dept.sales.</td>
</tr>
<tr>
<td></td>
<td>Note: To take advantage of the parallel loading features in SAS Visual Analytics clients, limit the path to eight characters or less and do not use a subdirectory. For example, use /sales instead of /dept/sales. The value of the server tag, which is based on the path, is used as a libref and must meet SAS naming requirements.</td>
</tr>
<tr>
<td>Enable Encryption:</td>
<td>Specifies whether to enable on-disk AES encryption for the library. Select Inherit from data server to use the encryption setting from the associated Hadoop server definition. Enabling encryption does not immediately encrypt existing non-encrypted data. Select Yes to enable on-disk AES encryption for the library.</td>
</tr>
<tr>
<td>New Encrypt Key:</td>
<td>If you selected Yes, enter an encryption key.</td>
</tr>
<tr>
<td>Confirm Encrypt Key:</td>
<td>Confirm the encryption key.</td>
</tr>
</tbody>
</table>
Assign the library to a Hadoop server by entering the following settings:

**Table 4.62 Connection Properties**

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Database Server</strong></td>
<td>Select a server from the drop-down list. Select the server that uses the same host as the SAS LASR Analytic Server to use with this library.</td>
</tr>
<tr>
<td><strong>Connection</strong></td>
<td>Use the pre-selected value (which prepends the selected server name with the string Connection:).</td>
</tr>
<tr>
<td><strong>Default Login</strong></td>
<td>This field is not editable. The value is (None).</td>
</tr>
</tbody>
</table>
Establishing Connectivity to a Teradata Database

Overview of Establishing Connectivity to a Teradata Server

The following figure provides a logical view of using Teradata with SAS/ACCESS software.

**Figure 4.37  Establishing Connectivity to a Teradata Database**

To define a database library in SAS Management Console with the SAS/ACCESS software, you need SAS/ACCESS Interface to Teradata software installed on your SAS server.

To define a user or user group using SAS Management Console, see “Define the DBMS User” on page 49 and “Define the DBMS User Group” on page 50.

To define a user or user group using SAS Environment Manager, see “Managing User Access” in SAS® Environment Manager User's Guide.

Setting up a connection from SAS to a Teradata database is a two-stage process.

1. Register the Teradata server.
2. Register the Teradata library.

**Stage 1: Register the Teradata Server**

To register the Teradata Server, perform the following steps in SAS Management Console:
1 Right-click the **Server Manager** and select **New Server** from the menu. This selection invokes the New Server wizard.

2 Select **Teradata Server** (under **Database Servers**) in the New Server wizard. Click **Next**.

3 Specify a name for the new Teradata server. The **Description** is optional.

4 Click **Next** to continue. This page lists default server properties. Do not change these default values.

<table>
<thead>
<tr>
<th>Table 4.63</th>
<th>Server Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>Sample Value</td>
</tr>
<tr>
<td>Major Version Number</td>
<td>14</td>
</tr>
<tr>
<td>Minor Version Number</td>
<td>0</td>
</tr>
<tr>
<td>Software Version</td>
<td>14.0</td>
</tr>
<tr>
<td>Vendor</td>
<td>Teradata Corporation</td>
</tr>
<tr>
<td>Associated Machine</td>
<td>Defaults to your machine</td>
</tr>
</tbody>
</table>

5 Click **Next** to continue. On this page, enter the Teradata connection properties for your Teradata database.

<table>
<thead>
<tr>
<th>Table 4.64</th>
<th>Connection Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>Sample Value</td>
</tr>
<tr>
<td>Server</td>
<td>Specify an entry in your (client) HOSTS file that provides an IP address for a database server connection. The value for this property corresponds to the value for the SERVER= option in the SAS/ACCESS software LIBNAME statement.</td>
</tr>
<tr>
<td>Account number</td>
<td>Specify the account number that you want to charge for the Teradata session. <strong>Note</strong>: This property is optional.</td>
</tr>
<tr>
<td>Authentication type</td>
<td><strong>User/Password</strong> specify the credentials (in this example, user and password) that are retrieved from metadata and that corresponds to the selected authentication domain, TeraAuth.</td>
</tr>
<tr>
<td>Authentication Domain</td>
<td>Specify the domain (in this example, <strong>Teraauth</strong>) that is used to authenticate logon attempts to the Teradata server. (You might need to create a new authentication domain. For more information, see “How to Store Passwords for a Third-Party Server” in SAS Intelligence Platform: Security Administration Guide.) Click <strong>New</strong> to access the New Authentication Domain dialog box. Then enter the appropriate value in the <strong>Name</strong> field and click <strong>OK</strong> to save the setting.</td>
</tr>
</tbody>
</table>
6 Click **Next**.

7 Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the wizard settings.

---

**Stage 2: Register the Teradata Library**

This example shows the process for establishing a SAS connection to a Teradata database. After you verify the system requirements for the SAS/ACCESS engine, test the database connection using a LIBNAME statement:

```sas
libname teralib teradata server=server-name user=user-ID pwd=password;
```

After you assign the LIBNAME statement, you can view your Teradata tables in the TeraLib library either by viewing it from the SAS Explorer window or by submitting a DATASETS procedure that is similar to the following:

```sas
proc datasets lib=teralib;
quit;
```

Add the Teradata library using SAS Management Console Data Library Manager plug-in.

1 In SAS Management Console, expand the **Data Library Manager** node.

2 Right-click **Libraries** and select **New Library** from the menu to open the New Library wizard.

3 Under **Database Data**, select **Teradata Library**. Click **Next** to continue to the next page of the wizard.

4 Enter a name for your library (for example, **TeraLibrary**) in the **Name** field. The **Description** field is optional. **Location** is the metadata folder where the library object is saved. You can keep the default location or change it.

Using a metadata folder to organize library and table objects in metadata (or any other metadata content that is supported by folders) is an efficient way to control access to those objects. Library and table objects inherit permissions from their respective metadata folders. Controlling access to these objects through permission inheritance is much easier when the objects are stored in the same metadata folder. You control access to the libraries and tables by denying ReadMetadata permission to the folder in which the library and tables are stored.

For more details about metadata folders, see "Working with SAS Folders" in SAS Intelligence Platform: System Administration Guide.

5 Click **Next** to continue to the next page of the wizard.

6 Select the appropriate server from the **Available servers** list and move it to the **Selected servers** list on the right. Note that even though the wizard states that this step is optional, you must select the server. For this example, use **SASApp**.

7 Click **Next** to continue to the next page of the wizard. In the **Libref** field, enter your libref (for example, TeraLib).
Table 4.65 Library Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libref</td>
<td>TeraLib</td>
</tr>
<tr>
<td>Engine</td>
<td>TERADATA</td>
</tr>
</tbody>
</table>

8 Click **Next** to move to the server and connection information page. Select the Teradata server that you created.

Table 4.66 Server and Connection Information

<table>
<thead>
<tr>
<th>Field</th>
<th>Sample Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Server</td>
<td>Teradata_server (Use the database server that you created in the New Server wizard.)</td>
</tr>
<tr>
<td>Database Schema Name</td>
<td>See your database administrator for the correct value. This is optional.</td>
</tr>
<tr>
<td>Default Login</td>
<td>Use the default value of <em>(None)</em>.</td>
</tr>
</tbody>
</table>

9 Click **Next**. Examine the final page of the wizard to ensure that the proper values have been entered. Click **Finish** to save the wizard settings.

At this point, your library is defined. However, to access data, you either need to register the tables or pre-assign the library. See “Registering and Verifying Tables” on page 144.

Registering and Verifying Tables

Overview to Registering Tables

You need to make sure that the end users of your SAS applications can gain access to tables in your data libraries. The exact steps and authorization requirements vary across applications and data types. You must always log on to the application, register the needed metadata, and verify the existence of the tables. This example focuses on the process used to verify SAS tables in SAS Management Console.

This user account that is used for registering tables must have these permissions:

- ReadMetadata and WriteMetadata permission in the DefaultACT for the repository.
- ReadMetadata and WriteMetadata permission on the library.
- WriteMemberMetadata permission to the folder where the table metadata is to be stored.
Permission to the data source to read the tables. This is not a SAS metadata permission.

When you register tables in metadata, you choose the metadata folder location to use for storing the table metadata. Be aware that the table metadata inherits the metadata permissions that are granted on the folder. Select or create a folder with the metadata permissions that meet your business needs. You can also use an existing folder and change the metadata permissions.

Verifying your access to tables in SAS Management Console is a two-stage process:

1. Register the tables.

---

### Stage 1: Register the Tables

To register the tables, perform the following steps:

1. Open the SAS Management Console, if necessary.
2. Select Data Library Manager ➔ Libraries to see the list of libraries.
3. Right-click the library that contains the tables that you need to import and select Register Tables.
4. Verify that the values shown in the fields in the Library details group box are correct. Click Next.
5. Click the tables that you need to select. (Hold down the Ctrl key and click to select more than one table.)
6. Check the metadata folder path in the Location field. Navigate to a folder or create a folder that has metadata permissions for user and group access that meets your business needs. Click Next.
7. Examine the final page of the wizard to ensure that the proper values have been entered. Click Finish to save the wizard settings.

**Note:** You can also register tables by using SAS Data Integration Studio or by using the METALIB procedure. For information about using the METALIB procedure, see Chapter 2, "Managing Table Metadata," on page 17.

---

### Stage 2: Verify Access to the Data in a SAS Application

Open an application that can view SAS data in order to view the data in the imported tables and review the data. For example, you can use SAS Data Integration Studio. To use SAS Data Integration Studio to view a registered table, perform the following steps:

1. Navigate to the Inventory tree and expand the Table node.
2 Right-click a table that you need to verify and select the **Open** option. Examine the data contained in the table in the View Data dialog box.

3 Close the View Data window.

4 (Optional) You can also examine the table’s **Properties** field. Right-click the table and select the **Properties** option.

5 Click the **Columns** tab to see column data for the table. Close the Properties dialog box.

---

**Read-only Access for Reporting Libraries**

If your site uses libraries for reporting, or for access exclusively by report generating applications such as SAS Information Map Studio and SAS Web Report Studio, then consider setting Read-Only access for the library. If the library is not set for Read-Only access, then even when reporting applications raise a query against the library, the underlying SAS session opens the data in read-write mode. In this case, simultaneous queries against the same library might be prevented. Simply put, if clients access the information in Read-Only mode, then set the library to read the data source in Read-Only mode.

**Table 4.67  Setting a Library for Read-Only Access**

<table>
<thead>
<tr>
<th>Library Type</th>
<th>Where to Set Read-only Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Database Data Libraries</td>
<td>Advanced Options dialog box, Input/Output tab, option Data access level for connection</td>
</tr>
<tr>
<td>Base SAS Library</td>
<td>Advanced Options dialog box, Options for any host tab</td>
</tr>
<tr>
<td>SAS Information Map Library</td>
<td>Always read-only, no configuration needed</td>
</tr>
<tr>
<td>SAS XML Library</td>
<td>Library properties page of the wizard, Library Access option</td>
</tr>
<tr>
<td>SAS Scalable PerformanceData Engine Library</td>
<td>Advanced Options dialog box, Options for any host tab</td>
</tr>
<tr>
<td>SAS/SHARE REMOTE Engine Library</td>
<td>Advanced Options dialog box, LIBNAME Options tab</td>
</tr>
</tbody>
</table>
Setting UNIX Environment Variables for SAS/ACCESS

If you are attempting to connect to data sources located on UNIX by using SAS/ACCESS, you must set environmental variables so that SAS servers can access the database. Each database vendor and operating system vendor requires specific environment variables to be set. A typical value is LD_LIBRARY_PATH. When using ODBC, two typical values are ODBCINI and ODBCINSTINI. For more information about the correct environment variables, go to SAS 9.4 Install Center Documentation, and use the operating system and SAS version to locate the appropriate SAS Foundation Configuration Guide.

To set the appropriate environment variables in the !SASROOT/bin/sasenv_local file, perform the following steps:

1. Edit the sasenv_local file and add the variables. This example uses sample values, substitute the proper paths:

   ODBCINI=/opt/drivers/odbc/odbc.ini
   export ODBCINI

   ODBCINSTINI=/opt/drivers/odbc/odbcinst.ini
   export ODBCINSTINI

   LD_LIBRARY_PATH=/opt/drivers/odbc/lib:\
   /opt/oracle/app/oracle/product/12.1.0/lib:$LD_LIBRARY_PATH
   export LD_LIBRARY_PATH

   ORACLE_HOME=/opt/oracle/app/oracle/product/12.1.0
   EXPORT ORACLE_HOME

2. In SAS Management Console, right-click the Workspace Server connection and select Validate to verify that the workspace server starts correctly with the new environment variables.

3. Restart the SAS/SHARE and SAS/CONNECT servers, if they are present in the deployment and reference the SAS/ACCESS library.

Troubleshooting SAS/ACCESS Connections to Relational Databases

This section provides information about troubleshooting a SAS/ACCESS library configuration when registering tables fails. To troubleshoot the SAS/ACCESS library, perform the following steps:
1. From the SAS Management Console, right-click the library icon and select Display LIBNAME Statement.

2. Start SAS on the SAS server host and issue the LIBNAME statement displayed from the SAS Management Console.

3. If the SAS log indicates failure, check the following items:
   
a. If this is UNIX environment, check “Setting UNIX Environment Variables for SAS/ACCESS” on page 147.
   
b. Check and revise the LIBNAME statement. For more information about LIBNAME statements for SAS/ACCESS engines, see SAS/ACCESS for Relational Databases: Reference. If you are successful at this stage, then use the Properties tab of the library to reconfigure the library.
   
c. Confirm that SAS/ACCESS is installed correctly. For installation information, go to SAS 9.4 Install Center Documentation and use the operating system and SAS version to locate the appropriate SAS Foundation Configuration Guide.

4. If the connection succeeds, run the DATASETS procedure:

   ```
   proc datasets library=libref;
   quit;
   ```

   If no members are returned, then check the schema value by performing the next step or contacting your database administrator.

5. Log on with the user account to the host where the SAS server is running, and use the native database client to connect to the database. If this fails, confirm the user account has file system privileges to the database client binaries and libraries.
Optimizing Data Storage

Overview of Optimizing Data Storage

For the purposes of querying, cube loading, and creating data marts and data warehouses, all four data storage structures (explained in “Overview of Connecting to Common Data Sources” on page 48 ) can be optimized to improve performance. Some optimization can be achieved by specifying transformation options in SAS Data Integration Studio. Some optimization requires hardware configuration, as in the case of SAS Scalable Performance Data (SPD) Engine tables. Cubes can be optimized for querying and loading during the cube loading process. For SAS tables, database tables, and SPD Engine tables, libraries can be defined in the metadata with options that enhance performance.

For more information, see these sections:

- “Compressing Data” on page 150
Compressing Data

Compression is a process that reduces the number of bytes that are required to represent each table row. In a compressed file, each row is a variable-length record. In an uncompressed file, each row is a fixed-length record. Compressed tables contain an internal index that maps each row number to a disk address so that the application can access data by row number. This internal index is transparent to the user. Compressed tables have the same access capabilities as uncompressed tables. Here are some advantages of compressing a file:

- reduced storage requirements for the file
- fewer I/O operations necessary to read from or write to the data during processing

Here are some disadvantages of compressing a file:

- More CPU resources are required to read a compressed file because of the overhead of uncompressing each observation.
- There are situations when the resulting file size might increase rather than decrease.

These are the types of compression that you can specify:

- CHAR to use the RLE (Run Length Encoding) compression algorithm, which works best for character data.
- BINARY to use the RDC (Ross Data Compression) algorithm, which is highly effective for compressing medium to large (several hundred bytes or larger) blocks of binary data.

You can compress these types of tables:

- all tables that are created during a SAS session. Besides specifying SAS system options on the command line or inside a SAS program with the OPTIONS statement, you can use SAS Data Integration Studio to set system options. For example, you can use the System Options field to set the COMPRESS= system option on a table loader transformation. (A table loader transformation generates or retrieves code that puts data into a specified target table.)
all tables for a particular library. For example, when you register a Base SAS engine library in the metadata, you can specify the COMPRESS= option in the Other options to be appended field on the Options for any host tab. For more information, see “Setting SAS LIBNAME Options That Affect Performance of SAS Tables” on page 159. For third-party relational database tables, you can use the Options to be appended field on the Other Options tab. For more information, see “Setting SAS LIBNAME Options That Affect Performance of SAS/ACCESS Databases” on page 160.

Note: You cannot specify compression for an SPD Engine data library.

For SPD Engine tables and third-party relational database tables, you can use the Table Options field in the table properties dialog box to specify the COMPRESS= option.

Note: The SPD Engine compresses the data component (DPF) file by blocks as the engine is creating the file. (The data component file stores partitions for an SPD Engine table.) To specify the number of observations that you want to store in a
compressed block, you use the IOBLOCKSIZE= table option in addition to the COMPRESS= table option. For example, in the Table Options field in the table properties dialog box, you might enter \texttt{COMPRESS=YES IOBLOCKSIZE=10000}. The default blocksize is 4096 (4k).

When you create a compressed table, SAS records in the log the percentage of reduction that is obtained by compressing the file. SAS obtains the compression percentage by comparing the size of the compressed file with the size of an uncompressed file of the same page size and record count. After a file is compressed, the setting is a permanent attribute of the file, which means that to change the setting, you must re-create the file. For example, to uncompress a file, in SAS Data Integration Studio, select \texttt{Default (NO)} for the Compressed option in the table properties dialog box for a SAS table.

For more information about compression, see \textit{SAS Data Set Options: Reference}.

---

**Indexing Data**

An index is an optional file that you can create to provide direct access to specific rows. The index stores values in ascending value order for a specific column or columns and includes information about the location of those values within rows in the table. In other words, an index enables you to locate a row by value. For example, if you use SAS to find a specific Social Security number (123-45-6789), SAS performs the search differently depending on whether there is an index on the column that contains the Social Security numbers:

- Without an index, SAS accesses rows sequentially in the order in which they are stored in the table. SAS reads each row, looking for \texttt{SSN=123-45-6789} until the value is found, or all observations are read.
- With an index on column \texttt{SSN}, SAS accesses the row directly. SAS satisfies the condition by using the index and going straight to the row that contains the value. SAS does not have to read each row.

When you create an index, you designate which columns to index. You can create two types of indexes:

- a simple index, which consists of the values of one column.
- a composite index, which consists of the values of more than one column. The values are concatenated to form a single value

For each indexed column, you can also perform these tasks:

- declare unique values. A unique index guarantees that values for one column or the combination of a composite group of columns remain unique for every row in the table. If an update tries to add a duplicate value to that column, then the update is rejected.
- keep missing values from using space in the index by specifying that missing values are not maintained by the index.

In addition to writing SAS code to create indexes, you can create indexes on target tables by using SAS Data Integration Studio. In SAS Data Integration Studio, you use the properties window for the table to index individual columns. When you create the index, you can also specify \texttt{Unique values} and \texttt{No missing values}. Note that any indexes registered in metadata for a target table are physically created when the job is run. Simply editing the properties for an existing table and
adding indexes does not update the physical table. The SAS Data Integration Studio properties dialog box for a table is shown below:

Figure 5.3  The Indexes Tab in the Properties Dialog Box for a Table Named STORE_ID

In general, SAS can use an index to improve performance in these situations:

- For cube loading, a composite index on the columns that make up the cube's hierarchies might provide best results.

- For WHERE processing, an index can provide faster and more efficient access to a subset of data. Note that to process a WHERE expression, SAS decides whether to use an index, or to read the table sequentially.

  Note: For WHERE processing, the Base SAS engine uses a maximum of one index. The SPD Engine can use multiple indexes.

Even though an index can reduce the time that is required to locate a set of rows, especially for a large table, there are costs that are associated with creating, storing, and maintaining the index. When deciding whether to create an index, you must consider increased resource usage, along with the performance improvement.

Once an index exists, SAS treats it as part of the table. That is, if you add or delete columns or modify values, the index is automatically updated.

For more information about indexes, see SAS Language Reference: Concepts.
Sorting Data

Overview to Sorting Data

You can sort table rows by the values of one or more character or numeric columns. For Base SAS tables and third-party relational database tables, the process either replaces the original table or creates a new table. You can perform sorting in two ways:

- using the SAS SORT procedure
- setting properties for a SAS sort template in SAS Data Integration Studio, as shown below:

**Figure 5.4  The Sort by Columns Tab in the Sort Properties Dialog Box**

To manage the memory that is used for the sorting process, you can specify the maximum amount of memory that is available to the sort. Generally, the sort size should be less than the physical memory available to the process. If the sorting requires more memory than you specify, then SAS creates a temporary utility file on disk. To specify a sort size in SAS Data Integration Studio, access the **Options** tab in the properties window for the sort template and enter a value in the **Sortsize** field, as shown below:
The SPD Engine has implicit sorting capabilities, which saves time and resources for SAS applications that process large tables. When the SPD Engine encounters a BY clause, if the data is not already sorted or indexed on the BY column, then the SPD Engine automatically sorts the data without affecting the permanent table or producing a new table. You can change the implicit sorting options when you define an SPD Engine library in the metadata. See “Setting SAS LIBNAME Options That Affect Performance of SAS SPD Engine Tables” on page 163.

For more information about the SORT procedure, see “SORT Procedure” in Base SAS Procedures Guide.

Multi-Threaded Sorting

The SAS system option THREADS activates multi-threaded sorting, which achieves a degree of parallelism in the sorting operations. This parallelism is intended to reduce the real time to completion for a given operation. However, the parallelism comes at the possible cost of additional CPU resources. For more information, see “Support for Parallel Processing” in SAS Language Reference: Concepts.

The performance of the multi-threaded sort is affected by the value of the SAS system option CPUCOUNT=. CPUCOUNT= indicates how many system CPUs are available for use by the multi-threaded sort. The multi-threaded sort supports concurrent input from the partitions of a partitioned table.

Note: For information about the support of partitioned tables in your operating environment, see the SAS documentation for your operating environment.

For more information about THREADS and CPUCOUNT=, see the chapter about SAS system options in SAS System Options: Reference.
Sorting a Database Table

When you use a third-party database table, the column ordering that is produced by the SORT procedure depends on whether the DBMS or SAS performs the sorting. If you use the BEST value of the SAS system option SORTPGM=, then either the DBMS or SAS performs the sort. If the DBMS performs the sort, then the configuration and characteristics of the DBMS sorting program affect the resulting data order. Most database management systems do not guarantee sort stability, and the sort might be performed by the database table regardless of the state of the SORTEQUALS or NOSORTEQUALS system options and the EQUALS or NOEQUALS procedure options.

If you set the SAS system option SORTPGM= to SAS, then unordered data is delivered from the DBMS to SAS and SAS performs the sorting. However, consistency in the delivery order of columns from a database table is not guaranteed. Therefore, even though SAS can perform a stable sort on the DBMS data, SAS cannot guarantee that the ordering of columns within output BY groups will be the same, run after run. To achieve consistency in the ordering of columns within BY groups, first populate a SAS table with the database table and then use the EQUALS or SORTEQUALS option to perform a stable sort.

Buffering Data for Base SAS Tables

For Base SAS tables, you might be able to make performance improvements by performing these tasks:

- tuning the size of table pages on disk by using the BUFSIZE= system option. SAS uses the BUFSIZE= option to set the permanent page size for the SAS table. The page size is the amount of data that can be transferred for an I/O operation to one buffer. If you know that the total amount of data is going to be small, you can set a small page size, so that the total table size remains small and you minimize the amount of wasted space on a page. Large tables that are accessed sequentially benefit from larger page sizes because sequential access reduces the number of system calls that are required to read the table.

- adjusting the number of open page buffers when the SAS table is processed. Increasing the value of the BUFNO= option can improve performance by enabling applications to read more data with fewer passes; however, your memory usage increases. You must determine the optimal value for your needs.

Besides specifying SAS system options on the command line or inside a SAS program with the OPTIONS statement, you can set the BUFSIZE= and BUFNO= system options in SAS Data Integration Studio. For example, you can set these System Options in the properties window for a table loader transformation.
Buffering Data for SAS/ACCESS Databases

For DB2 (UNIX and PC), ODBC, OLE DB, Oracle, SQL Server, and Sybase, you can adjust page buffers by setting the INSERTBUFF= and READBUFF= options on the library (see “Setting SAS LIBNAME Options That Affect Performance of SAS/ACCESS Databases” on page 160) or on the individual table. The options are described as follows:

- The INSERTBUFF= option specifies the number of rows to insert. SAS allows the maximum that is supported by the DBMS. The optimal value for this option varies with factors such as network type and available memory. You might need to experiment with different values in order to determine the best value for your site.

- The READBUFF= option specifies the number of rows to hold in memory. SAS allows the maximum number that is supported by the DBMS. Buffering data reads can decrease network activities and increase performance. However, because SAS stores the rows in memory, higher values for READBUFF= use more memory. In addition, if too many rows are selected at once, then the rows that are returned to the SAS application might be out of date. For example, if someone else modifies the rows, you might not see the changes.

For more information about the INSERTBUFF= and READBUFF= options, see SAS/ACCESS for Relational Databases: Reference.
Using Threaded Reads

Most SAS/ACCESS interfaces support threaded Reads. With a threaded Read, the table read time can be reduced by retrieving the result set on multiple connections between SAS and a DBMS. To perform a threaded Read, SAS performs these tasks:

1. It creates threads, which are standard operating system tasks that are controlled by SAS, within the SAS session.
2. It establishes a DBMS connection on each thread.
3. It causes the DBMS to partition the result set and reads one partition per thread. To cause the partitioning, SAS appends a WHERE clause to the SQL so that a single SQL statement becomes multiple SQL statements, one for each thread.

Threaded Reads only increase performance when the DBMS result set is large. Performance is optimal when the partitions are similar in size. In most cases, threaded Reads should reduce the elapsed time of the SAS job. However, threaded Reads generally increase the workload on the DBMS. For example, threaded Reads for DB2 under z/OS involve a trade-off, generally reducing job elapsed time but increasing DB2 workload and CPU utilization.

Threaded Reads are most effective on new, faster computer hardware running SAS, and with a powerful parallel edition of the DBMS. For example, if SAS runs on a fast uniprocessor or on a multiprocessor machine and your DBMS runs on a high-end SMP server, you receive substantial performance gains.

For information about how to turn the threaded Read function on or off for a DBMS library, see “Setting SAS LIBNAME Options That Affect Performance of SAS/ACCESS Databases” on page 160.

For information about threaded Reads, see SAS/ACCESS for Relational Databases: Reference.

Validating SAS Scalable Performance Data Engine Hardware Configuration

The SAS Scalable Performance Data Engine (SPD Engine) automatically determines the optimal process to use to evaluate observations for qualifying criteria specified in a WHERE statement. WHERE statement efficiency depends on such factors as whether the columns in the expression are indexed. A SAS configuration validation program that measures I/O scalability with respect to WHERE processing can help you determine whether your system is properly configured for performing WHERE processing with the SPD Engine. The program performs these tasks:

1. It creates a table with two numeric columns.
Setting SAS LIBNAME Options That Affect Performance of SAS Tables

You can set SAS LIBNAME options that might affect performance of the Base SAS engine. You set these options when you use the New Library wizard to register a Base SAS engine library in the metadata repository. The LIBNAME options are available on the Options for any host tab and the Host-specific options tab in the Advanced Options dialog box. To access the Advanced Options dialog box, click the Advanced Options button on the Library Options page of the New Library wizard.

Figure 5.7 The Options for Any Host Tab in the Advanced Options Dialog Box for a Base SAS Library

Here are some examples of options that might affect performance:

Data representation for the output file (OUTREP=)  
For all operating environments, you can specify the data representation for the output file. Specifying this option enables you to create files within the native environment by using a foreign environment data representation. For example, an administrator who works in a z/OS operating environment might want to create a file on an HFS system so that the file can be processed in an HP UNIX environment. Specifying HP_UX_64 as the value for this option forces the data representation to match the data representation of the UNIX operating environment that will process the file. This method of creating the file can
enhance system performance because the file does not require data conversion when being read by an HP UNIX machine.

Input/output block size (BLKSIZE=)
For Windows, UNIX, and z/OS environments, you can specify the number of bytes that are physically read during an I/O operation. The default is 8 kilobytes, and the maximum value is 1 megabyte.

Number of page caches to use for each open member (CACHENUM=)
For VMS, you can specify the number of page caches to use during I/O operations. The number of caches can potentially reduce the number of I/Os that are required to access the data. You can also set the size of each cache (CACHESIZE= option).

The **Other option(s) to be appended** field can be used to specify LIBNAME options such as COMPRESS=. (See “Compressing Data” on page 150.)

For information about each of the LIBNAME options in the Advanced Options dialog box, click the **Help** button.

---

Setting SAS LIBNAME Options That Affect Performance of SAS/ACCESS Databases

The following SAS LIBNAME options can be used to tune performance of the SAS/ACCESS engines. You can set these options when you use the New Library wizard to register the database libraries in the metadata repository. To access the Advanced Options dialog box, click the **Advanced Options** button on the Library Options page of the New Library wizard.
The tabs that are available in the Advanced Options dialog box, as well as the options on each of the tabs, vary between database management systems. The following list provides a description of the options on the **Optimization** tab for DB2 libraries for UNIX and PC:

**Block insert buffer size (INSERTBUFF=)**
- Specifies the number of rows in a single Insert operation. See “Buffering Data for SAS/ACCESS Databases” on page 157.

**Block read buffer size (READBUFF=)**
- Specifies the number of rows of DBMS data to read into the buffer. See “Buffering Data for SAS/ACCESS Databases” on page 157.

**Pass functions to the DBMS that match those supported by SAS (SQL_FUNCTIONS=)**
- When set to **ALL**, specifies that functions that match functions supported by SAS should be passed to the DBMS. The functions that are passed are: DATE, DATEPART, DATETIME, TIME, TIMEPART, TODAY, QRT, COMPRESS, SUBSTR, DAY, SECOND, INDEX, TRANWRD, HOUR, WEEKDAY, LENGTH, TRIMN, MINUTE, YEAR, REPEAT, MOD, MONTH, BYTE, and SOUNDEX. Use of this option can cause unexpected results, especially if used for NULL processing and date, time, and timestamp handling. Exercise care when using this option.
Pass DELETE to the DBMS (DIRECT_EXE=)
specifies that an SQL delete statement is passed directly to the DBMS for
processing. Selecting this option improves performance because SAS does not
have to read the entire result set and delete one row at a time.

Whether to use indexes (DBINDEX=)
specifies whether SAS uses indexes that are defined on DBMS columns to
process a join. Valid values are YES or NO. For more information about indexes,
see “Indexing Data” on page 152.

Whether to check for null keys when generating WHERE clauses (DBNULLKEYS=)
specifies whether the WHERE clause should detect NULL values in columns.
Valid values are YES or NO. YES is the default for most interfaces and enables
SAS to prepare the statement once and use it for any value (NULL or NOT
NULL) in the column.

Multiple data source optimization (MULTI_DATASRC_OPT=)
when processing a join between two tables, specifies whether an IN clause
should be created to optimize the join. Valid values are NONE and IN_CLAUSE.
IN_CLAUSE specifies that an IN clause containing the values read from a
smaller table are used to retrieve the matching values in a larger table based on
a key column designated in an equijoin.

When processing a join between a SAS table and a DBMS table, the SAS table
should be smaller than the DBMS table for optimal performance.

Whether to create a spool file for two-pass processing (SPOOL=)
specifies whether to create a utility spool file during transactions that read data
more than once. In some cases, SAS processes data in more than one pass
through the same set of rows. Spooling is the process of writing rows that have
been retrieved during the first pass of a data Read to a spool file. In the second
pass, rows can be re-read without performing I/O to the DBMS a second time. In
cases where the data needs to be read more than once, spooling improves
performance. Spooling also guarantees that the data remains the same between
passes. Valid values are YES or NO.

Threaded DBMS access (DBSLICEPARM=)
specifies the scope of DBMS threaded Reads and the number of threads. If this
option is set to the default, then PROC SQL does not use threading to read data
for a web report. To force a specified number of threads for a threaded Read
from the DBMS server, change the default to (ALL, number-of-threads).

Note: If PROC SQL attempts implicit pass-through, then threading is disabled,
regardless of the Threaded DBMS access setting. To disable implicit pass-
through, set the Pass generated SELECT SQL to the DBMS - DBMS
processing option to NO.

For more information about threaded Reads, see “Using Threaded Reads” on
page 158.

Pass generated SELECT SQL to the DBMS - DBMS processing (DIRECT_SQL=)
specifies whether generated SQL is passed to the DBMS for processing. Valid
values are YES or NO.

Pass generated SELECT SQL to the DBMS - exceptions to DBMS processing
(DIRECT_SQL=)
if the value for the previous option is YES, then this option specifies how
generated SQL is passed to the DBMS for processing. For example, NOWHERE
prevents WHERE clauses from being passed to the DBMS for processing.
Setting SAS LIBNAME Options That Affect Performance of SAS SPD Engine Tables

The **Other Options** tab, which is available for all database management systems, can be used to specify LIBNAME options such as the COMPRESS= option. For more information, see “Compressing Data” on page 150.

For information about each of the LIBNAME options in the Advanced Options dialog box, click the **Help** button. For information about all SAS/ACCESS LIBNAME options, see *SAS/ACCESS for Relational Databases: Reference*.

---

### Setting SAS LIBNAME Options That Affect Performance of SAS SPD Engine Tables

The following SAS LIBNAME options can be used to tune performance of the SAS Scalable Performance Data Engine (SPD Engine). You can set these options when you use the New Library wizard to register an SPD Engine library in the metadata repository. The LIBNAME options are available on the **Options for any host** tab in the Advanced Options dialog box. To access the Advanced Options dialog box, click the **Advanced Options** button on the Library Options page of the New Library wizard. The **Advanced Options** dialog box is shown below:

**Figure 5.9** The Options for Any Host Tab in the Advanced Options Dialog Box for an SPD Engine Library

![Advanced Options Dialog Box](image)

- **Data path:** `/disk2/data'`='/disk3/data'
- **Index path:** `/disk2/idx'`='/disk3/idx'
- **Meta path:**
- **Partition size:** 1G24
- **Temp:** No
- **By sort:** Yes
- **Idx By:** Yes
- **Starting observation number:**
- **Ending observation number:**
- **Read Only:** No
Data path (DATAPATH=) specifies a list of paths in which to store partitions (DPF) files for an SPD Engine table. The engine creates as many partitions as are needed to store all the data. The size of the partitions is set using the PARTSIZE= option. Partitions are created in the specified paths in a cyclic fashion. The data path area is best configured as multiple paths. Allot one I/O controller per data path to provide high I/O throughput, which is the rate at which requests for work are serviced by a computer system. The data path area is best configured for redundancy (RAID 1).

Index path (INDEXPATH=) specifies a path or a list of paths in which to store the two index component files (HBX and IDX) that are associated with an SPD Engine table. Additional specified paths accept the overflow from the immediately preceding path. The index path area is best configured as multiple paths. Use a volume manager file system that is striped across multiple disks (RAID 0) to enable adequate index performance, both when evaluating WHERE clauses and creating indexes in parallel. Redundancy (RAID 5 or RAID 10) is also recommended.

Meta path (METAPATH=) specifies a list of overflow paths in which to store metadata component (MDF) files for an SPD Engine table. The metadata component file for each table must begin in the primary path. When that primary path is full, the overflow is sent to the specified METAPATH= location. The metadata path area is best configured for redundancy (RAID 1) so that metadata about the data and its indexes is not lost.

Partition size (PARTSIZE=) specifies the size (in megabytes) of the data component partitions when an SPD Engine table is created. By splitting the data portion of an SPD Engine table at fixed-size intervals, you can gain a high degree of scalability for some operations. For example, the SPD Engine can spawn threads in parallel, up to one thread per partition for WHERE evaluations.

Temp (TEMP=) specifies whether to create a temporary subdirectory of the directory specified in the Path field on the Library Properties window. The directory is used to temporarily store the metadata component files associated with table creation. It is deleted at the end of the SAS session.

By sort (BYSORT=) specifies that the SPD Engine should perform an automatic implicit sort when it finds a BY statement for processing data in the library (unless the data is indexed on the BY column). Valid values are YES (perform the sort) and NO (do not perform the sort). The default is YES.

Starting observation number (STARTOBS=) specifies the number of the starting observation in a user-defined range of observations that are qualified with a WHERE expression. By default the SPD Engine processes all observations in the table.

Ending observation number (ENDOBS=) specifies the number of the ending observation in a user-defined range of observations that are qualified with a WHERE expression. By default the SPD Engine processes all observations in the table.

In addition to the LIBNAME options, there are also table and system options that can be used to tune SPD Engine performance. For example, the SPDEUTILLOC= system option allots space for temporary files that are generated during SPD Engine operations. This area is best configured as multiple paths. Use a volume manager
file system that is striped across multiple disks (RAID 0) to reduce out-of-space conditions and improve performance. Redundancy (RAID 5 or RAID 10) is also recommended because losing the work area could stop the SPD Engine from functioning.

The “Quick Guide to the SPD Engine Disk-I/O Set-Up” available at http://support.sas.com/rnd/scalability/spde/spde_setup.pdf helps you do the following:

- determine the amount of space that needs to be allocated to the data, metadata, index, and work areas
- evaluate the advantages and disadvantages of different RAID groups for each of the different types of areas

For more information about table and other system options for the SPD Engine, see http://support.sas.com/rnd/scalability/spde/syntax.html. For more information about each of the LIBNAME options in the Advanced Options dialog box, click the Help button.

---

**SAS Grid Computing Data Considerations**

SAS Grid computing has become an important technology for organizations that do the following:

- have long-running applications that can benefit from parallel execution
- want to leverage existing IT infrastructure to optimize computing resources and manage data and computing workloads

The function of a grid is to distribute tasks. Each of the tasks that are distributed across the grid must have access to all the required input data. Computing tasks that require substantial data movement generally do not perform well in a grid. To achieve the highest efficiency, the nodes should spend the majority of the time computing rather than communicating. With grid computing using SAS Grid Manager, the speed at which the grid operates is related more to the storage of the input data than to the size of the data.

Data must either be distributed to the nodes before running the application or—much more commonly—made available through shared network libraries. Storage on local nodes is discouraged. The data storage must scale to maintain high performance while serving concurrent data requests.

The parallel data load is monitored throughout.

---

**Application Response Monitoring**

SAS implements the Application Response Monitoring 4.0 (ARM) specification. SAS offers macros, system options, and logging facility as an ARM agent for collecting application availability, performance, usage, and transaction response time. For more information about the ARM implementation, see the SAS Interface to Application Response Measurement (ARM): Reference.

SAS Data Integration Studio can report the following measures for jobs:
- number of records processed
- duration of step in the job
- I/O statistics

To view the metrics within SAS Data Integration Studio, right-click the diagram background for the job and select **Collect Runtime Statistics**. When the job is run, view the **Statistics** tab in the Details area of the window.
Introduction to Managing OLAP Cube Data

Online Analytical Processing (OLAP) is a technology that is used to create decision support software. OLAP enables application users to quickly analyze information that has been summarized into multidimensional views and hierarchies. By summarizing predicted queries into multidimensional views before run time, OLAP tools provide the benefit of increased performance over traditional database access tools. Most of the resource-intensive calculation that is required to summarize the data is done before a query is submitted. One of the advantages of OLAP is how data and its relationships are stored and accessed. OLAP systems house data in structures that are readily available for detailed queries and analytics.

Data Storage and Access

Organizations usually have databases and data stores that maintain repeated and frequent business transaction data. This provides simple yet detailed storage and
retrieval of specific data events. However, these data storage systems are not well suited for analytical summaries and queries that are typically generated by decision makers. For decision makers to reveal hidden trends, inconsistencies, and risks in a business, they must be able to maintain a certain degree of momentum when querying the data. An answer to one question usually leads to additional questions and review of the data. Simple data stores do not generally suffice.

The data warehouse is a structure better suited for this type of querying. In a data warehouse, data is maintained and organized so that complicated queries and summaries can be run. OLAP further organizes and summarizes specific categories and subsets of data from the data warehouse. One particular type of data structure derived from a data warehouse is the cube. A cube is a set of data that is organized and structured in a hierarchical, multidimensional arrangement. Such an arrangement results in a robust and detailed level of data storage with efficient and fast query returns. Stored, precalculated summarizations called aggregations can be added to the cube to improve cube access performance.

Exporting and Importing Cubes

Cubes are exported and imported as part of a SAS package. SAS Management Console is one of the user interfaces that can perform the import and export of packages. For more information about creating SAS packages, see "Promotion Tools Overview" in SAS Intelligence Platform: System Administration Guide.

The data administrator impact of exporting and importing cubes is that when cubes are imported, the tables used in the cube must be available. Building the aggregations for the cube is also computationally intensive. The following list highlights some best practices:

- It is impractical to package the detail tables and summary data for large cubes. Do not export them in the package.
- If the cube is being imported to a new metadata server (as opposed to being moved to a new folder on the same metadata server), then make sure the same data sources for the detail tables used by the original cube are available and registered. Accomplish this by registering the same data servers and data libraries on the destination metadata server. If the cube uses a drill-through table, ensure that the library is pre-assigned.
- When importing the cubes, be prepared to associate the cube with an OLAP schema. Consider that the OLAP schema determines the group of cubes that an OLAP server can access.
- Once the cube is imported, the cube and its job are registered in metadata with relationships to an OLAP schema, tables, and folders. By default, the aggregations for the cube must be built after the cube is imported. Due to the computational intensity, consider rebuilding the cube during a period of low activity.
About OLAP Schemas

OLAP schemas provide an organizational function. An OLAP schema is a list of cubes that are grouped together so that they can be accessed by one or more SAS OLAP Servers. Each cube is listed in one and only one OLAP schema. Each SAS OLAP Server is required to use one OLAP schema. Multiple servers can use the same schema. To assign cubes to specific servers, you create new OLAP schemas. This might be necessary if you have multiple large cubes. In that case, you might want to assign one cube to one host, to one SAS OLAP Server, and to one OLAP schema.

New OLAP schemas are created with the Create OLAP Schema wizard in SAS OLAP Cube Studio or SAS Management Console. SAS OLAP Servers are assigned to new OLAP schemas by changing server properties in SAS Management Console. To create a new OLAP schema or assign an OLAP schema to a SAS OLAP Server using SAS Management Console, see “Create or Assign an OLAP Schema” on page 169. A SAS OLAP Server reads its assigned OLAP schema from metadata only as the server starts. Assigning a new OLAP schema to a server requires that you restart the SAS OLAP Server.

When building, updating, or deleting cubes, you can specify OLAP schemas in the Cube Designer wizard of SAS OLAP Cube Studio. Alternatively, if you choose to write SAS code for PROC OLAP, the schema is specified in the OLAP_SCHEMA= option of the METASVR statement.

Create or Assign an OLAP Schema

To create a new OLAP schema or assign an OLAP schema to a SAS OLAP Server, perform the following steps:

1. Open SAS Management Console.
2. In the left pane, expand Server Manager.
3. Under Server Manager, locate the SAS Application Server that contains the SAS OLAP Server. The name of one such SAS Application Server might be SASApp, for example.
4. Right-click the top-level SAS Application Server and select Properties.
5. In the Properties window, click the OLAP Schema tab.
6. Click New to create a new OLAP schema, or select the down arrow to choose an existing OLAP schema.
7. Click OK to save changes and close the Properties window.
8. Restart the SAS OLAP Server using the SAS OLAP Server Monitor.
Building a Cube

Overview of Building a Cube

The following is a summary of the cube-building process. For additional information about building and modifying SAS OLAP cubes, see the *SAS OLAP Server: User’s Guide*.

Before building a cube, you should collect and scrub your data in addition to planning a dimensional design. When you define the cube, you define the dimensions and measures for the cube along with information about how aggregations should be created and stored. There are two methods of creating a cube:

- You can submit PROC OLAP code by using either the SAS Program Editor or a batch job. If you use PROC OLAP, the cube is created, and then the cube definition is stored in a metadata repository. This is referred to as the long form of PROC OLAP.

- You can use the Cube Designer interface in SAS OLAP Cube Studio to define and create the cube. The Cube Designer first stores the cube definition in a metadata repository, and then submits a shorter form of PROC OLAP code to create the cube. This is referred to as the short form of PROC OLAP.

**Note:** The Cube Designer can also be launched from SAS Data Integration Studio.

Preparations for Building a Cube

To build a cube by using either PROC OLAP or SAS OLAP Cube Studio, you must perform several preliminary tasks:

- Configure a metadata server.

- Define an OLAP server in the metadata. The server does not need to be running to create cubes, but it must be defined in the metadata.

- Analyze the data to determine the location of the table or tables to use for building your cubes and what dimensions and measures will be created.

- Register the table or tables to use for creating the cube in the metadata. You do this by using SAS Data Integration Studio or by using SAS OLAP Cube Studio and SAS Management Console as follows:

  - Use SAS Management Console to register the metadata for the server to use for accessing the tables. This is a SAS Application Server with a workspace server component.

  - Use SAS Management Console to register metadata for the library that contains the table.
In SAS OLAP Cube Studio, specify the server to use for accessing the tables. To set the server, select **Tools ⇒ Options**. Or, if the shortcut bar is displayed, select **Options** to set the server.

In SAS OLAP Cube Studio, select **Source Designer** to load the table definitions (or other information source) as follows:

- From the shortcut bar, select **Tools ⇒ Source Designer** or select **Source Designer**.
- Select a Source Type (SAS, ODBC, and so on), and then select **Next**.
- If you have not specified a server, or if the server that is specified is not valid, then you are prompted again for a server.
- Select the library that contains the tables that you want to use, and then select **Next**.
- Select the tables to register and then select **Next**.
- Select **Finish**. The table definitions are register in metadata.

If you start to create a cube and do not see the table that you need to continue, then you can click the **Define Table** button in any of the windows that prompt for tables.

In the Finish window of the cube designer, you are given the option to create the physical cube. The metadata definition is always stored as you leave the Finish window. However, you can defer creation of the physical cube because it might be a resource and time intensive process. If you choose to create the cube as you leave the Finish window, then a SAS Workspace Server must be defined so that you can submit PROC OLAP code to it. This server is defined in SAS Management Console.

For more information about the different data types that you can use to load cubes from, see “Planning for SAS OLAP Cubes” in **SAS OLAP Server: User’s Guide**.

**Note:** The SAS Metadata Server enables duplicate librefs to be defined in the metadata. To ensure that the correct library definition is found on the metadata server, you should assign the libref by using the LIBNAME statement for the metadata engine before submitting the PROC OLAP code. Otherwise, PROC OLAP selects the first library definition that it finds with your specified libref, and it associates your cube metadata with that definition. The selected library definition might or might not contain a description of the data that was actually used to build your cube. For more information about using the LIBNAME statement for the metadata engine, see “Metadata LIBNAME Engine” in **SAS Language Interfaces to Metadata**.

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, if you save a cube in `c:\olapcubes` and name the cube **Campaigns**, the cube is saved in the directory `c:\olapcubes\CAMPAIGNS`.

---

### Storage Location Requirements for Cube Metadata and Related Objects

When storing metadata that describes a cube, the metadata objects that describe the cube and associated libraries and source tables must be stored in the same repository, or the metadata that describes the cube must be in a custom repository.
that is dependent on the repository that contains the library and table objects. Otherwise, you will not be able to create the cube. In addition, the library and table objects that are referenced by a cube must always be in the same repository. The following options illustrate these conditions:

- The library, table, and cube objects can be in a Foundation repository.
- The library, table, and cube objects can be in Project A, which is dependent on the Foundation repository.
- The library and table objects can be in the Foundation repository, and the cube object can be in Project A.
- The cube object cannot be in the Foundation repository, and the library and table objects cannot be in Project A.
- The table object cannot be in the Foundation repository, and the library and cube objects cannot be in Project A.
- The library object cannot be in the Foundation repository, and the table and cube objects cannot be in Project A.

---

Making Detail Data Available to a Cube for Drill-Through

You can drill through an OLAP report to the underlying detail data only after you make the detail data available to the cube. You can use either SAS OLAP Cube Studio or the OLAP procedure to make detail data available to the cube:

- In SAS OLAP Cube Studio, you can specify a table for drill-through when you create or edit the cube using the Cube Designer wizard. On the Drill-Through page of the wizard, either select a table and click the right-arrow and then Next to specify the drill-through table, or just click Next if drill-through is not needed. The following figure shows the Cube Designer - Drill Through page of the Cube Designer wizard:
For more information about the Cube Designer wizard, see the SAS OLAP Cube Studio Help. Note that for star schema tables, a view that fully joins the fact and dimension tables is the drill-through table.

In the PROC OLAP statement, use the DRILLTHROUGH_TABLE option to specify the name of the drill-through table to use. For more information about the DRILLTHROUGH_TABLE option, see “OLAP Procedure” in *SAS OLAP Server: User’s Guide*.

### Making Detail Data Available to an OLAP Server for Drill-Through

You can drill through an OLAP report to the underlying detail data only after you make the detail data available to the OLAP Server. In order for the OLAP server to make detail data available for a cube, the library for the table that contains the detail data must be registered so that the OLAP server can identify the library to use. Also, the library permissions must allow ReadMetadata permission. The simplest way to register the library to the server is to pre-assign it and store the library in a folder that grants ReadMetadata permission to PUBLIC.

To specify a library as pre-assigned for an OLAP server, perform the following steps:

1. In Data Library Manager (in SAS Management Console), find the **Libraries** folder and perform one of the following tasks to get to the dialog box that lets you select advanced options:
   - For a new library, right-click the **Libraries** folder and select **New Library** to start the New Library wizard. Then navigate to the page that enables you to specify the libref.
For an existing library, open the **Libraries** folder and right-click the desired library. Select **Properties** from the drop-down menu, and then select the **Options** tab in the properties dialog box.

**Figure 6.2**  **Library Properties**

2. Click **Advanced Options**.

3. Select the **Library is pre-assigned** check box on the **Pre-Assign** tab in the Advanced Options dialog box.

4. On the **Assign** tab of the properties dialog box or the server selection page of the New Library wizard, ensure that the selected application server is the server container that contains your OLAP server.

**Figure 6.3**  **Specifying an Assigned Server**

5. Click **OK** in the properties dialog box, or finish entering information in the wizard.

6. Restart the OLAP server.

The selected library is assigned after the selected OLAP server starts. After the OLAP server starts, ensure that the library is pre-assigned to the correct SAS OLAP server. The OLAP server also generates a record in the log file stored at `SAS-configuration-directory\ Lev1\SASApp\OLAPServer\Logs\`.

The following example shows how pre-assigned libraries are identified in the log file:
Making Detail Data Available to an Information Map for Drill-Through

You can drill through an OLAP report to the underlying detail data only after you make the detail data available to the information map. In order for an information map to produce a report that has drill-through capabilities, an option must first be set in the information map.

For an existing information map, open the information map, right-click it, and then select Properties from its drop-down menu. Select the Allow drill-through to detail data check box on the Definition tab in the Information Map Properties dialog box. This check box is displayed only when a drill-through table is specified for the cube that the OLAP information map is using as its data source.

Figure 6.4 Information Map Properties
If your cube contains an extremely large amount of detail data, then in order to view that data from within SAS Information Map Studio, you might need to increase the Java heap size for SAS Information Map Studio or increase the maximum number of drill-through rows that your SAS OLAP Server can handle. The default number of drill-through rows that can be displayed by a query is 300,000 rows.

You can increase the number of drill-through rows that your OLAP server can handle by changing the OLAP server definition with the Server Manager plug-in to SAS Management Console. To increase the number of drill-through rows, perform the following steps:

1. In the navigation tree for Server Manager, find the node that represents your physical OLAP server.
2. Right-click the icon and select Properties.
3. In the properties dialog box, select the Options tab, and then click Advanced Options.
4. In the Advanced Options dialog box, select the Server tab, and then enter the desired value for the Maximum number of flattened rows field.
5. Click OK to save the setting.
Library Options

Library Options: Base SAS

Options

Where applicable, the corresponding LIBNAME option is provided in parentheses following the field name.

Library is pre-assigned
if the library is pre-assigned, select this check box and choose a pre-assignment type.

Pre-assignment Type
specifies the type of pre-assignment used for the library, if applicable. Choices are as follows:

By native library engine
The library is assigned through METAAUTORESOURCES options, using the library engine defined for that library.

By metadata library engine
The library is assigned through METAAUTORESOURCES options, using the Metadata Library Engine (MLE). Using MLE ensures that access controls that are placed on the library and its tables and columns are enforced.

By external configuration
The library is assigned through an external definition or by an autoexec file.

Advanced Options: Options for Any Host

For details, see LIBNAME Statement in SAS Global Statements: Reference.
Library access (ACCESS)
specifies the type of access that users should have to this library. Choices are as follows:

- Blank (no value)
- READONLY (users have Read access)
- TEMP (the library should be treated as a scratch library)

Data representation for output file (OUTREP)
specifies the default format for data sets in the library. New data sets are created in this format, and any existing data sets that are written to the library are given the selected format.

Encoding to use when reading a file (INENCODING)
specifies the file type encoding to use when accessing a data set in the library.

Encoding to use when creating a file (OUTENCODING)
specifies the file type encoding to use when creating a data set in the library.

Whether to replace empty SAS data sets (REEMPTY)
specifies whether new empty data sets replace existing data sets with the same name. The choices are YES or NO.

Translation table for encoding conversion (TRANTAB)
specifies the translation table to use for character conversions. SAS applies the translation table whenever a data set that uses a format different than the host is written to the host.

Other option(s) to be appended
specifies any other valid LIBNAME options in this field. Any options and values that you specify must use correct syntax. Options are added to the generated LIBNAME statement exactly as they were specified in this field and without any error checking.

Advanced Options: Host-Specific Options

Note: Host-specific fields are displayed after you make a selection in the Select Host drop-down list.

Windows Host (PC)
For details, see LIBNAME Statement: Windows in SAS Companion for Windows.

Input/output block size (BLKSIZE= )
specifies the number of bytes that are physically read during an input or output operation. The default value is 8KB, and the maximum value is 1MB.

UNIX Host (UNIX)
For details, see LIBNAME Statement: UNIX in SAS Companion for UNIX Environments.

Input/output block size (BLKSIZE= )
specifies the number of bytes that are physically read during an input or output operation. The default value is 8KB, and the maximum value is 1MB.
Control file locking (FILELOCKS=)
  specifies the file locking for the library. Valid values are as follows:
  - **NONE**: Locking is off. The library is not protected from shared access.
  - **FAIL**: Locking is on. If the file cannot be locked, it cannot be opened.
  - **CONTINUE**: Locking is on. The library cannot be opened if locked by someone else but can be opened with a warning message if it is locked for some other reason.

Enable direct file I/O (ENABLEDIRECTIO=)
  specifies that direct file I/O is available for all files that are associated with the library.

Turn on direct file I/O for all the files (USEDIRECTIO=)
  if the value of the Enable direct file I/O field is set to **Yes**, specifies whether to turn on or off direct file I/O for all files associated with the library.

Size of blocks of data (TRANSFERSIZE=)
  specifies the size of blocks of data in kilobytes or megabytes. Specify the value as nK for kilobytes and nM for megabytes.

z/OS Host (MVS)

For details, see LIBNAME Statement: z/OS in SAS Companion for z/OS.

Input/output block size (BLKSIZE=)
  specifies the block size that SAS uses when allocating the library data set. The maximum value is 32760.

Data class for an SMS-managed data set (DATACLAS=)
  specifies the data class for an SMS-managed data set. The value for this field can have up to eight characters. Ask your system administrator which data-class names are used for SAS libraries at your site.

Disposition of data set (DISP=)
  specifies the status of the library data set at the beginning and ending of a job, as well as what to do if the library creation process terminates abnormally. Valid values for this field are **status** or (**status**, **normal-termination-disp**, **abnormal-termination-disp**) where

  **status**
  specifies the status of the file at the beginning of a job. Valid values are as follows:
  - **NEW**: A new library data set is created.
  - **OLD**: The data set exists and is not to be shared.
  - **SHR**: The data set exists and can be shared.

  **normal-termination-disp**
  specifies what happens to the library data set if the job using the data set ends normally. Valid values are as follows:
  - **DELETE**: The library is deleted at the end of the step.
  - **KEEP**: The data set is kept. This is the default value for existing data sets.
  - **CATLG**: The system places an entry in the system or user catalog. This is the default value for new library data sets.
  - **UNCATLG**: The system deletes the entry in the system or user catalog.
abnormal-termination-disp
specifies what happens to the library data set if the job using the data set ends abnormally. Valid values are as follows:

- **DELETE**: The library is deleted at the end of the step.
- **KEEP**: The data set is kept. This is the default value for existing data sets.
- **CATLG**: The system places an entry in the system or user catalog. This is the default value for new library data sets.
- **UNCATLG**: The system deletes the entry in the system or user catalog.

**Data set label type and contents**
specifies the type and contents of the library data set label. This option corresponds to the LABEL= option in the FILENAME statement and is identical to the JCL LABEL= parameter.

**Data set name from which to copy DCB attributes for new lib (LIKE=)**
specifies the data set to copy DCB attributes from for the new data set.

**Management class for SMS data set (MGMTCLAS=)**
specifies the management class for the library data set. The class is predefined and controls how the data set is managed (for example, how often it is backed up and how it is migrated). The value can be up to eight characters. This option is compatible with the SMS MGMTCLAS= JCL parameter.

**How much disk space to allocate (SPACE=)**
specifies how much disk space to provide for the library data set. The format for this option is `(unit, (primary,secondary),RLSE,type, ROUND)`, where
- **unit** specifies the units used for allocating space. Valid values are TRK (tracks), CYL (cylinders), and blklen (blocks whose length is blklen bytes).
- **primary** specifies the number of tracks, cylinders, or blocks to allocate.
- **secondary** specifies how many additional tracks, cylinders, or blocks to allocate if more space is needed.
- **RLSE** specifies that unused allocated space be released when the data set is closed.
- **type** specifies the type of storage space required. Valid values are CONTIG (allocated space must be contiguous), MXIG (maximum contiguous space is required), and ALX (allocated space must be in different area of contiguous space).
- **ROUND** if unit is blklen, specifies that allocated space must not include a fractional part of a cylinder.

**Storage class for an SMS-managed data set (STORCLAS=)**
specifies a storage class for the library data set. The library data set is predefined. It controls the device that the SMS data set is stored on. The value can be up to eight characters. This option is compatible with the SMS STORCLAS= JCL parameter.

**Device where the data library is to reside (UNIT=)**
specifies a device for multivolume data libraries.

**Volume(s) where the data library is to reside (VOLSER=)**
specifies up to five disk volume serial numbers where the library resides.
Number of minutes to wait for a data set already in use (WAIT=)
specifies the number of minutes the SAS software waits for a library data set
before the operation fails.

OpenVMS Host (VMS)
For details, see LIBNAME Statement: OpenVMS in SAS Companion for OpenVMS.

Initial number of blocks allocated to a file (ALQ=)
specifies the number of disk blocks to allocate to the library data set.

Initial number of pages allocated to a file (ALQMULT=)
specifies the number of pages to allocate to the library data set.

Number of page caches to use for each open member (CACHENUM=)
specifies the number of input/output data caches used for each SAS file.

Number of blocks to extend a file's allocation (DEQ=)
specifies how many disk blocks to add when OpenVMS automatically extends a
SAS data set during a Write operation.

(DEQMULT=)
specifies how many pages to add when OpenVMS automatically extends a SAS
data set during a Write operation.