CHAPTER

1

Overview of Common Data Sources

Overview

This chapter provides a brief introduction to the most common data sources that you encounter as you perform administrative tasks. The features of each data source are described. 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.

Accessibility Features in the SAS Intelligence Platform Products

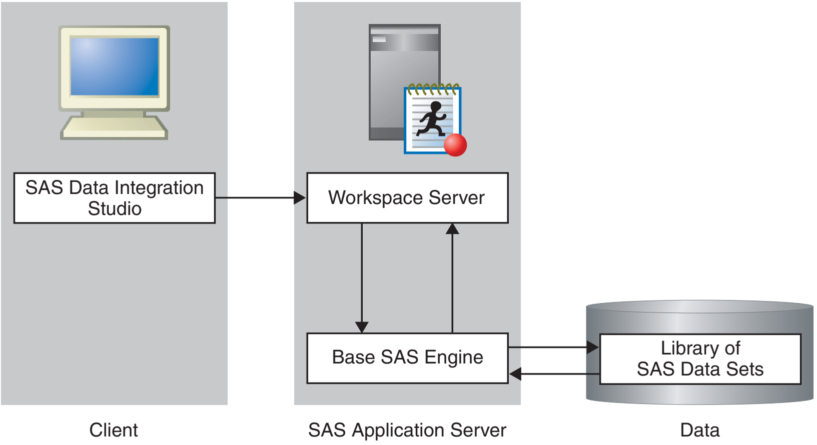
For information about accessibility for any of the products mentioned in this book, see the documentation for that product. If you have questions or concerns about the accessibility of SAS products, send e-mail to **accessibility@sas.com**.

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 data 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*

Language Reference: Dictionary. The following display shows how connectivity to SAS data sets is set up.

Figure 1.1 Establishing Connectivity to SAS Data Sets



See “Establishing Connectivity to a Library of SAS Data Sets” on page 12 for a detailed example of a SAS data set connection.

Shared Access to SAS Data Sets

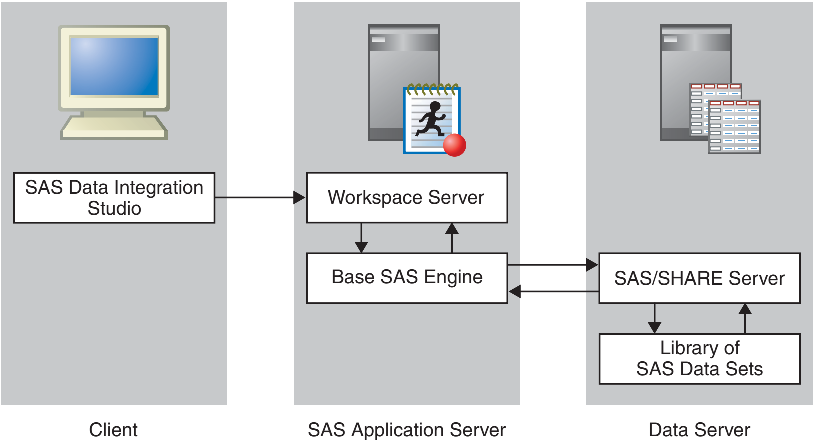
SAS/SHARE software provides concurrent update access to SAS files for multiple users. SAS/SHARE is often required for transaction-oriented applications where multiple users need to update the same SAS data sets at the same time. Data entry applications where multiple users are entering data to the same dataset are a good example of this type of usage. SAS/SHARE software provides both member- and record-level locking. Therefore, two or more users can update different observations within the same data set, and other users can print reports from the same data set.

SAS/SHARE supports multi-user read/write access to both SAS data files and SAS catalogs. Multi-user access to SAS catalogs simplifies the maintenance of applications by allowing users and developers to share the same program libraries. Users can execute applications at the same time that developers update the source programs.

SAS/SHARE software also acts as a data server that delivers data to users for their processing needs. This capability provides data administrators both a centralized point of control for their data and a secure environment to control who accesses the data. SAS/SHARE is also designed to be a reliable data server that functions as long as the system that the server is running on is operational.

Finally, SAS/SHARE allows you to make use of SAS software’s ability to define views of your data. This allows administrators to restrict certain users to subsets of data for security or efficiency purposes. Access to rows and columns in SAS tables can be defined using this technique. The following display illustrates shared access to SAS data sets.

Figure 1.2 Establishing Shared Access to SAS Data Sets



See "Establishing Shared Access to SAS Data Sets" on page 15 for a detailed example of a share SAS data set connection.

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:

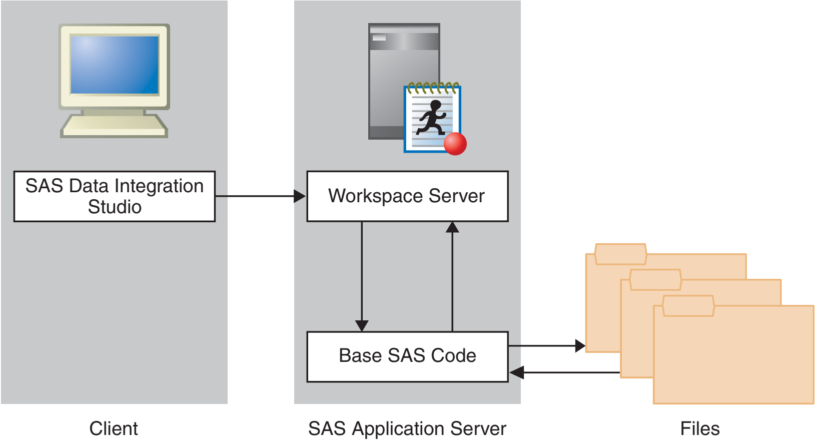
- the delimited external file wizard for external files in which data values are separated with a delimiter character. This wizard enables you to specify multiple delimiters, nonstandard delimiters, missing values, and multi-line records.
- the fixed-width external file wizard for external files in which 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 display illustrates establishing connectivity to external files.

Figure 1.3 Establishing Connectivity to External Files



See "Establishing Connectivity to a Flat File" on page 17 for a detailed example of an external file connection.

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 data 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 product. For additional information, see the *SAS XML LIBNAME Engine User's Guide*.

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 ships with a sample ODS tag set, if needed. An output XML document can either be:

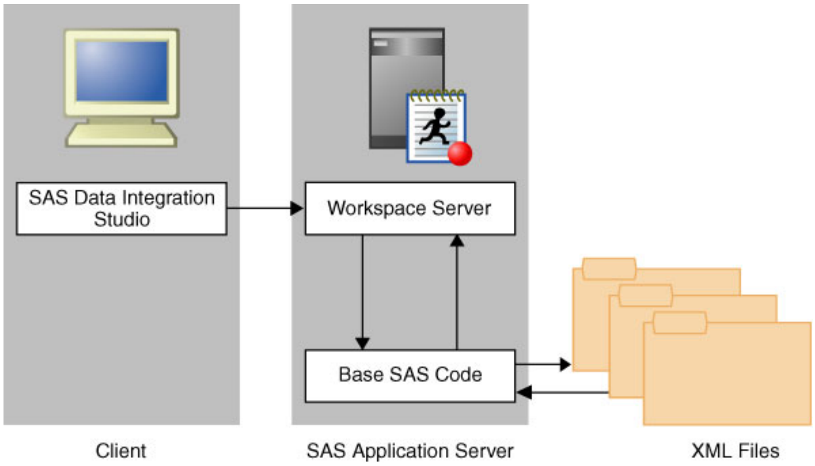
- □ used by a product that processes XML documents
- □ moved to another host for the XML LIBNAME engine to then process by translating the XML markup back to a SAS data set

Since 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.

When you are writing an XML file, we recommend that you define the library specifically for your write operation.

The following display illustrates connectivity to XML files.

Figure 1.4 Establishing Connectivity to XML Files



Relational Database Sources

SAS/ACCESS

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

Several of the SAS/ACCESS engines support threaded reads. This enables you to read entire blocks of data on multiple threads instead of reading data just one record at a time. This feature can reduce I/O bottlenecks and enables thread-enabled procedures to read data quickly. These engines and DB2 on z/OS also have the ability to access database management system (DBMS) data in parallel by using multiple threads to the parallel DBMS server.

The following SAS/ACCESS engines support this functionality:

□ Oracle

□ Sybase

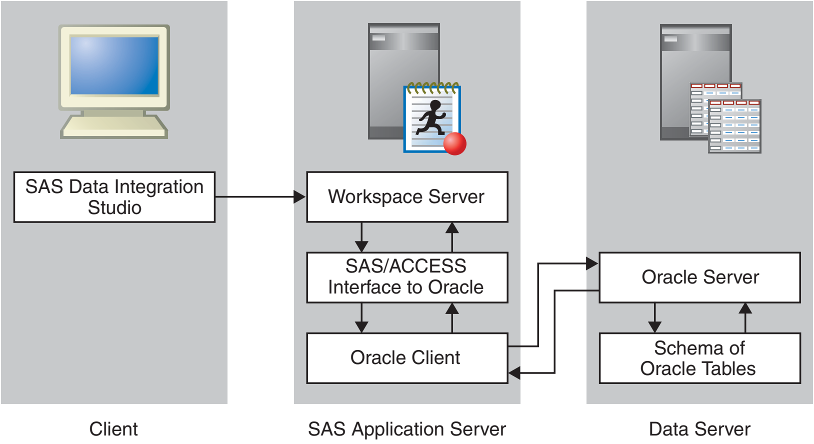
□ DB2 (UNIX and PC)

□ SQL Server

□ Teradata

For more information about using the SAS/ACCESS interfaces, see *SAS/ACCESS for Relational Databases: Reference*. The following display illustrates how connectivity to Oracle databases is set up.

Figure 1.5 Establishing Connectivity to Oracle Databases



See “Establishing Connectivity to an Oracle Database” on page 21 for a detailed example of an Oracle connection.

ODBC Sources

Open database connectivity (ODBC) standards provide a common interface to a variety of databases, including AS/400, dBASE, Microsoft Access, Oracle, Paradox, 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 adheres to the specification.

The basic components and features of ODBC include the following:

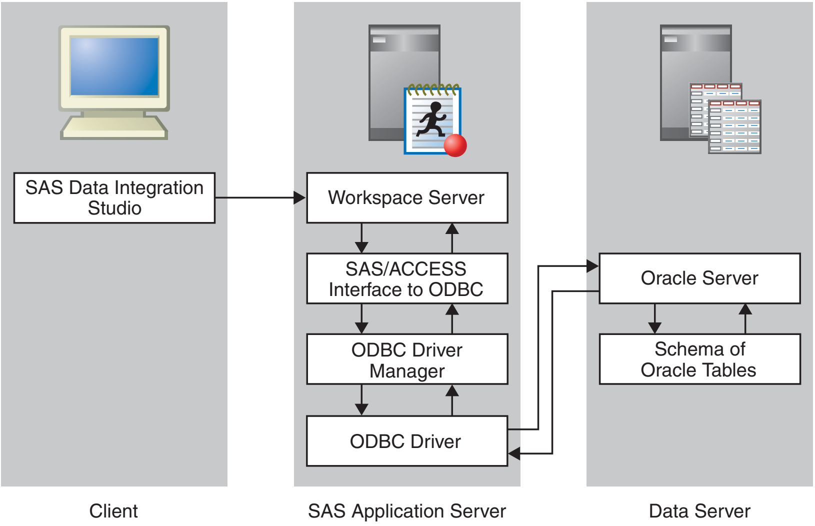
□ ODBC functionality is provided by three components: the client interface, the ODBC driver manager, and the ODBC driver. SAS provides the SAS/ACCESS interface to ODBC, which is the client interface. For PC platforms, Microsoft developed the ODBC Administrator, which is used from the Windows Control 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, 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.

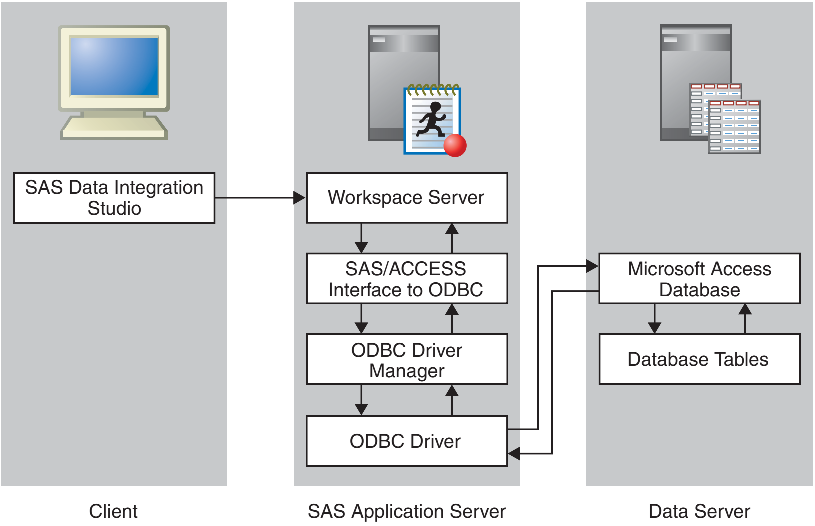
The following display illustrates how ODBC is used to establish connectivity to Oracle databases.

Figure 1.6 Establishing Connectivity to Oracle Databases By Using ODBC



See “Establishing Connectivity to an Oracle Database by Using ODBC” on page 26 for a detailed example of an ODBC-based Oracle connection. The following display illustrates how ODBC is used to establish connectivity to Access databases.

Figure 1.7 Establishing Connectivity to Access Databases By Using ODBC



See “Establishing Connectivity to a Microsoft Access Database by Using ODBC” on page 31 for a detailed example of an ODBC-based Access connection.

Scalable Performance Data Servers

Both the SAS Scalable Performance Data Engine (SPD Engine) and the SAS Scalable Performance Data Server (SPD Server) are designed for high-performance data delivery.

They enable rapid access to SAS data for intensive processing by the application. The SAS SPD Engine and SAS 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/output functions.

The SAS SPD Engine 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 SAS SPD Server, but it lacks the additional complexity of a full-blown server. The SAS SPD Server is available as a separate product or as part of the SAS Intelligence Storage bundle. It is a multi-user parallel-processing data server with a comprehensive security infrastructure, backup and restore utilities, and sophisticated administrative and tuning options. The SAS SPD Server libraries can now be defined using SAS Management Console.

The SAS SPD Engine and SAS 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 machine's available CPUs.

Although threaded I/O is an important part of both product offerings' functionality, their real power comes from the way that 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, SAS SPD Engine and SAS SPD Server are high-speed alternatives for processing very large tables. They read and write tables that contain billions of observations.

The SAS SPD Engine and SAS 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

The SAS SPD Engine runs on UNIX, Windows, z/OS (on HFS and zFS file systems only), and OpenVMS for Integrity Servers (on ODS-5 file systems only) platforms. The SAS SPD Server runs on Tru64 UNIX, Windows Server, HP-UX, and Sun Solaris platforms.

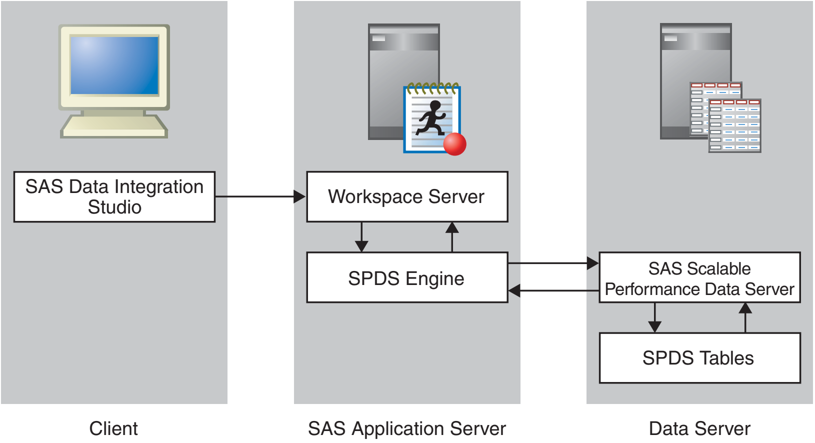
Symmetric Multiprocessing

The SAS SPD Engine 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 SAS SPD Engine reads a data file, it launches one or more threads for each CPU; these threads then read data in parallel. By using these threads, a SAS SPD Engine 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 SAS SPD Engine, see *SAS Scalable Performance Data Engine: Reference* and **support.sas.com/rnd/scalability/spde**.

The following display illustrates how connectivity to SPDS servers is established.

Figure 1.8 Establishing Connectivity to an SPD Server



See “Establishing Connectivity to a Scalable Performance Data Server” on page 34 for a detailed example of an SPDS server connection.

ERP Systems

Enterprise Resource Planning (ERP) systems are composed of thousands of tables, columns, variables and fields. Although they contain a wealth of data, they lack several key features:

- □ the ability to provide integration with other data sources
- □ the ability to do backward-looking drill-down analysis into what caused the effect (Business Intelligence)
- □ the ability to do forward-looking cause and effect analysis (Business Analytics)

To make it possible to get to the data the ERP systems contain, SAS provides data surveyors for each of the following ERPs:

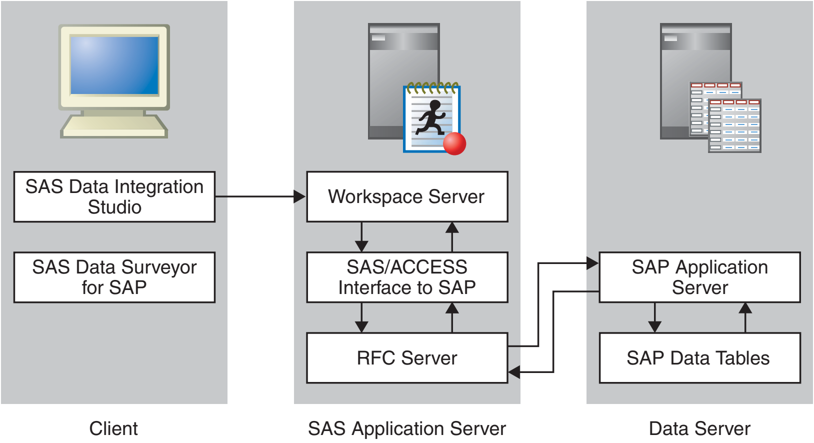
- □ SAP
- □ Peoplesoft
- □ Oracle
- □ Siebel

Data surveyors contain 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 these business applications is at the heart of the data surveyor. Each data surveyor has knowledge about the specific application it is designed for. This knowledge contains information about the ERP metadata that allows you to do the following:

- □ understand complex data structures
- □ navigate the large amounts of tables (SAP is over 20,000)

The following display illustrates how connectivity to SAP servers is established.

Figure 1.9 Establishing Connectivity to an SAP Server



See “Establishing Connectivity to a SAP Server” on page 38 for a detailed example of an SAP server connection.