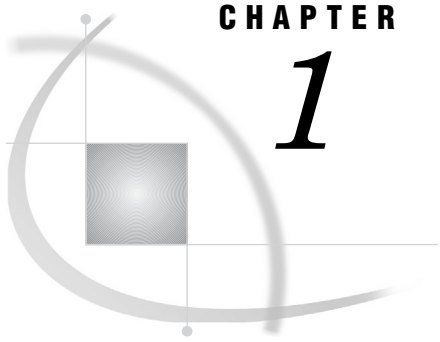


CHAPTER

1

Overview of Common Data Sources

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.

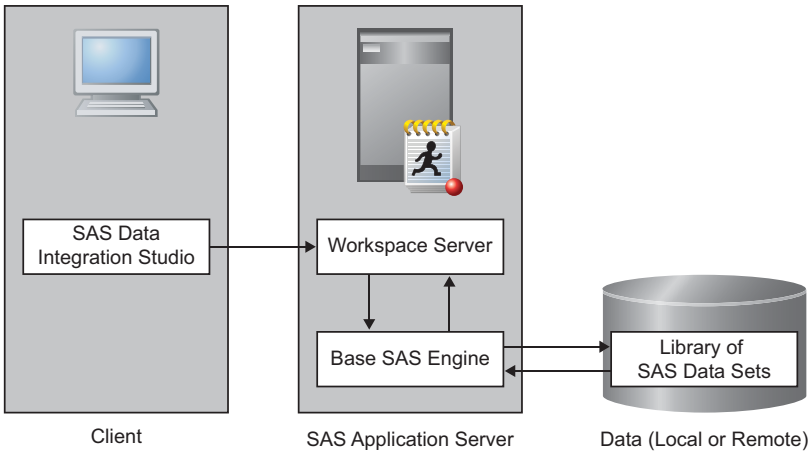
Accessibility Features in the SAS Intelligence Platform Products

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 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 figure shows how connectivity to SAS data sets is configured.

Figure 1.1 Establishing Connectivity to SAS Data Sets



For a detailed example of a SAS data set connection, see “Establishing Connectivity to a Library of SAS Data Sets” on page 19.

Shared Access to SAS Data Sets

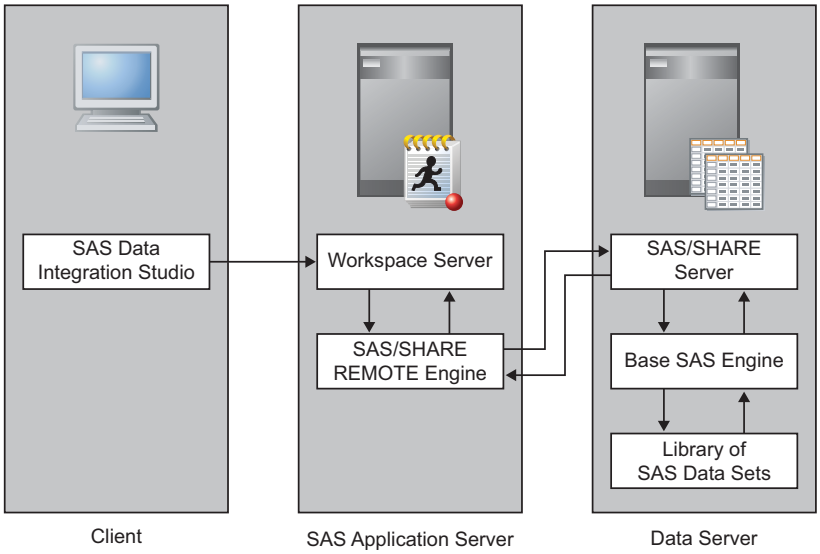
SAS/SHARE software provides concurrent update access to SAS files for multiple users. SAS/SHARE is often required for transaction-oriented applications where multiple users need to update the same SAS data sets at the same time. Data entry applications where multiple users are entering data to the same data set are a good example of this type of usage. SAS/SHARE software provides both member-level locking 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 and 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 enables you use SAS software 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 figure shows shared access to SAS data sets. Note that the data server in the figure can be a different operating system and architecture from the SAS Application Server, if the site is licensed for that configuration.

Figure 1.2 Establishing Shared Access to SAS Data Sets

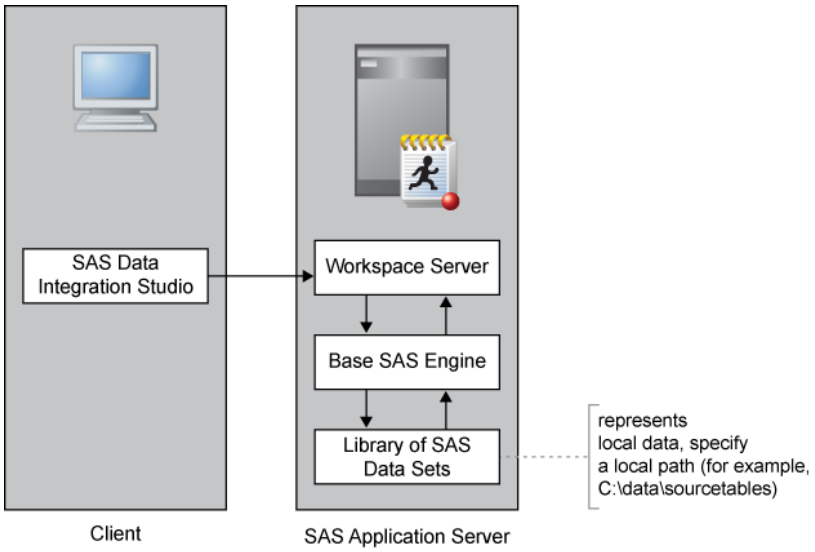


For a detailed example of a shared SAS data set connection, see “Establishing Shared Access to SAS Data Sets” on page 22.

Local and Remote Access to Data

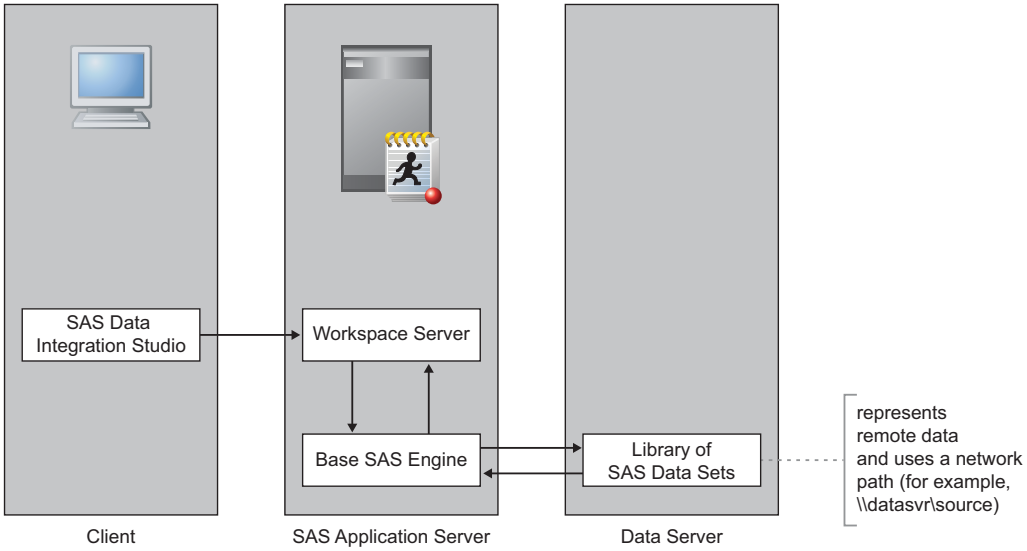
To access data you must register the data as a library in SAS Management Console. The procedures for accessing data and registering data are explained later in this document. However, one of the important details for file-based data, such as SAS data sets, is that you need to specify the file system path to the data. This path is needed so 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, then 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.

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:

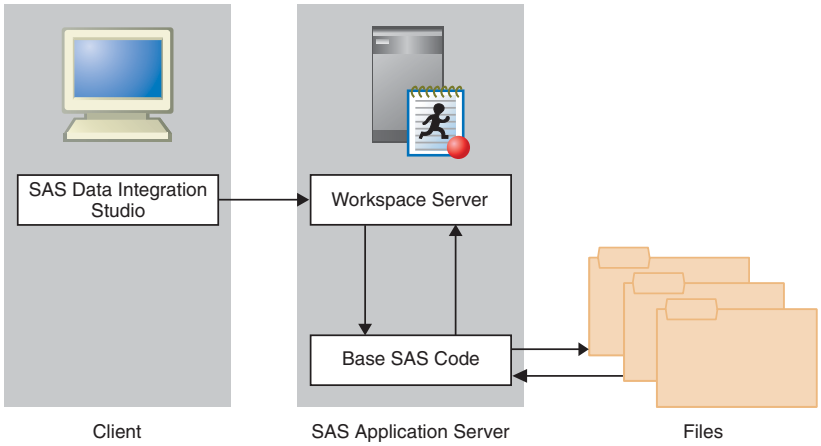
- the delimited external file wizard for external files in which data values are separated with a delimiter character. This wizard enables you to specify multiple delimiters, nonstandard delimiters, missing values, and multi-line records.
- the fixed-width external file wizard for external files in which 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 32.

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

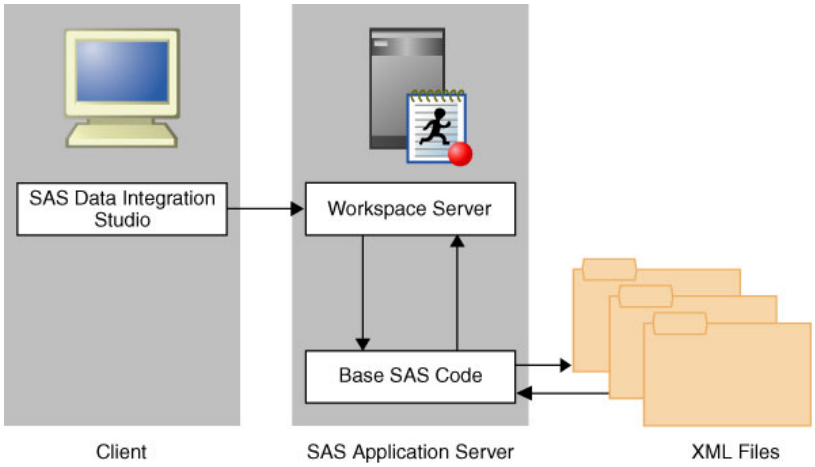
XML files are written by the XML Writer transformation provided by SAS Data Integration Studio. The XML LIBNAME engine supports Output Delivery System (ODS) tag sets; XMLMaps are not supported for writing. The XML Writer transformation in SAS Data Integration Studio ships with a sample ODS tag set, if 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 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 heterogenous 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 also can be stored in third-party hierarchical and relational databases such as DB2, Oracle, SQL Server, and Teradata. SAS/ACCESS interfaces provide fast, efficient reading and writing of data to these facilities.

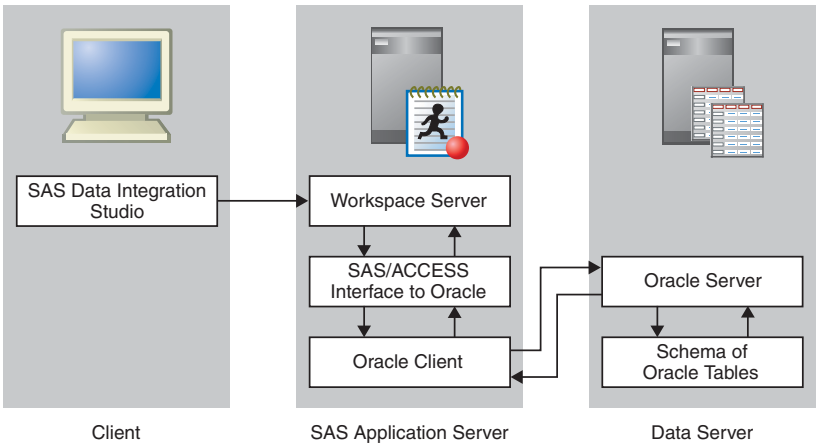
Several of the SAS/ACCESS engines support threaded reads. This enables you to read entire blocks of data on multiple threads instead of reading data just one record at a time. This feature can reduce I/O bottlenecks and enables thread-enabled procedures to read data quickly. These engines and DB2 on 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 figure shows how connectivity to Oracle databases is configured:

Figure 1.7 Establishing Connectivity to Oracle Databases



For a detailed example of an Oracle connection, see “Establishing Connectivity to an Oracle Database” on page 37.

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.

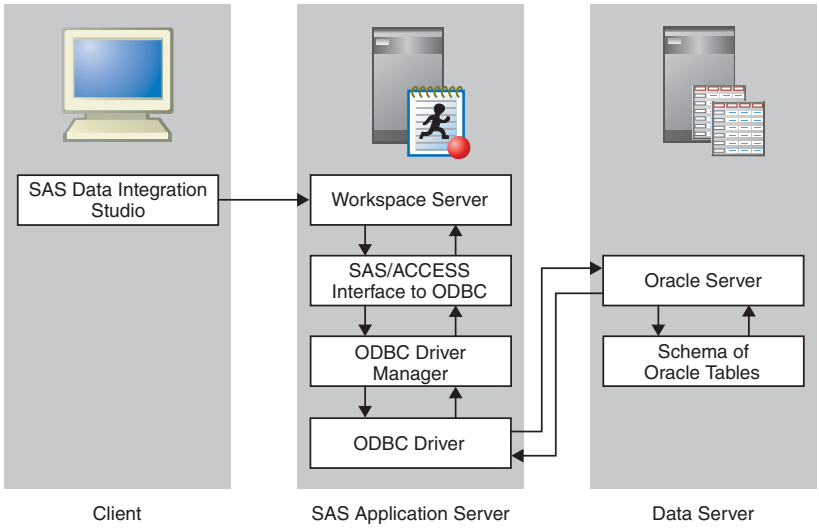
Note: If a SAS/ACCESS engine is available for a database, then performance is better with the SAS/ACCESS engine rather than with the ODBC interface. △

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.

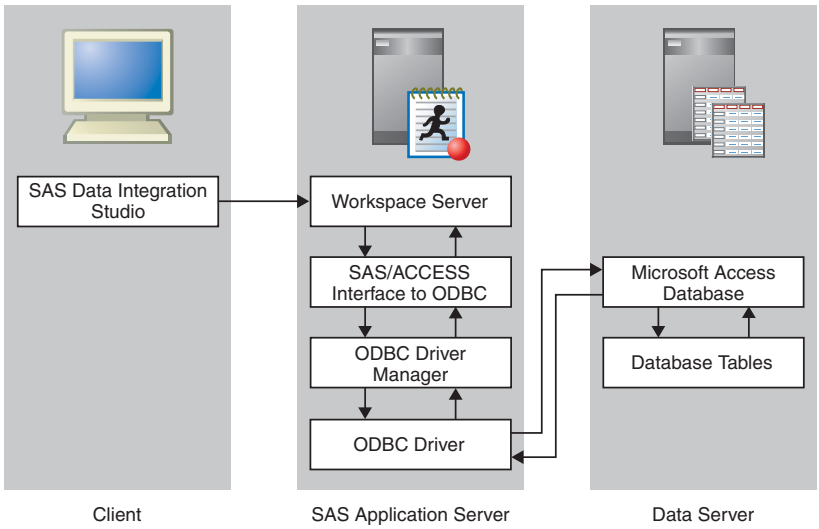
The following figure shows how ODBC is used to establish connectivity to Oracle databases:

Figure 1.8 Establishing Connectivity to Oracle Databases by Using ODBC



For a detailed example of an ODBC-based Oracle connection, see "Establishing Connectivity to an Oracle Database by Using ODBC" on page 41. 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 45.

Scalable Performance Data Server and Scalable Performance Data Engine

Overview of Scalable Performance Data Server and Scalable Performance Data Engine

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 and 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. SAS SPD Server libraries can be defined using SAS Management Console.

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

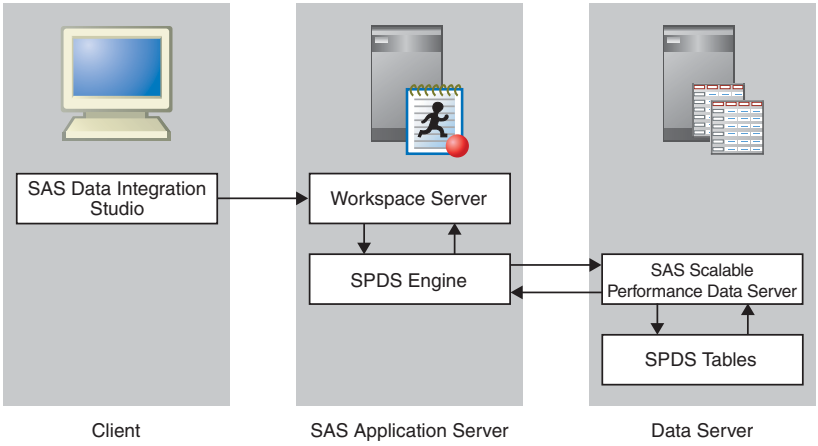
The SAS SPD Server exploits a hardware and software architecture known as symmetric multiprocessing (SMP). An SMP machine has multiple CPUs and an

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 SAS 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, a SAS 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 SAS SPD Server, see *SAS Scalable Performance Data Server: Administrator's Guide* and **`support.sas.com/rnd/scalability/spds`**.

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

Figure 1.10 Establishing Connectivity to a SAS SPD Server



For a detailed example of a SAS SPD Server connection, see "Establishing Connectivity to a Scalable Performance Data Server" on page 48.

Dynamic Clustering

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

This virtual table structure provides the SAS SPD Server with the architecture to offer flexible storage to allow a user to organize tables based on values contained in 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 SAS 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. Additionally, 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 ADD and LIST. You execute each of these commands within PROC SPDO.

The CLUSTER CREATE command requires three options:

- the name of the cluster table (cluster-table-name) that will be created
- a list of SAS Scalable Performance Data Server tables that will be included in the cluster (using the MEM= option)
- the number of slots (using the MAXSLOT= option), for member tables, that the cluster will have

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

```
PROC SPDO LIBRARY=domain-name;
SET ACLUSER user-name;
CLUSTER CREATE cluster-table-name
MEM = SPD-Server-table1
MEM = SPD-Server-table2
MEM = SPD-Server-table3
MEM = SPD-Server-table4
MEM = SPD-Server-table5
MEM = SPD-Server-table6
MEM = SPD-Server-table7
MEM = SPD-Server-table8
MEM = SPD-Server-table9
MEM = SPD-Server-table10
MEM = SPD-Server-table11
MEM = SPD-Server-table12
MAXSLOT=24;
QUIT;
```

Here is the syntax for the UNDO command:

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

This example shows the syntax for the ADD command:

```
PROC SPDO LIBRARY=domain-name;
SET ACLUSER user-name;
CLUSTER ADD sales_hist
MEM = 2005sales_table1
MEM = 2005sales_table2
MEM = 2005sales_table3
MEM = 2005sales_table4
MEM = 2005sales_table5
MEM = 2005sales_table6;
QUIT;
```

Finally, here is the syntax for the LIST command:

```
PROC SPDO LIBRARY=domain-name;
SET ACLUSER user-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.

ERP and CRM Systems

Overview of ERP and CRP Systems

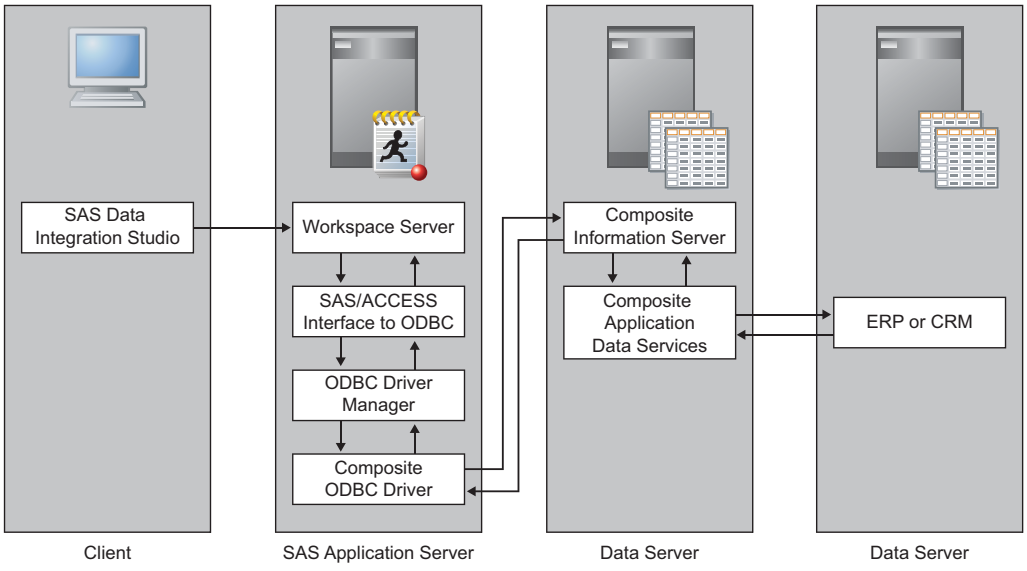
Enterprise Resource Planning (ERP) and Customer Relationship Management (CRM) systems contain a wealth of data in tables, columns, variables, and fields, but they lack several key features:

- 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)

New Data Surveyors

Previously, SAS provided data surveyors that relied on accessing the underlying database—Oracle, DB2, and SQL Server—and not the application APIs. SAS now provides, through software from Composite Software, both Service Oriented Architecture (SOA) and SQL data services that unlock the data in PeopleSoft, Oracle Applications, Siebel, as well as the recently offered Salesforce.com. The following figure shows how SAS interacts with Composite Software:

Figure 1.11 Establishing Connectivity Using Composite Software



The Composite Information Server uses a Data Service to access a data source through the data source's API. The Composite Information Server then offers the data

through an ODBC interface. You configure an ODBC data source name on the SAS Application Server with the Composite ODBC driver. Then you use SAS Management Console to register an ODBC server and an ODBC library. For a detailed example of a Composite Information Server connection to Salesforce.com, See "Establishing Connectivity to a Composite Information Server" on page 24.

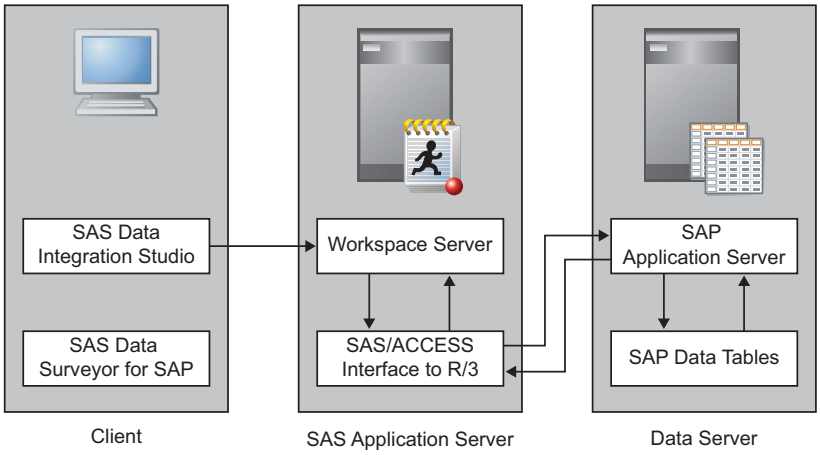
Data Surveyor for SAP

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 these business applications 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 about the ERP metadata that allows you to do the following:

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

The following figure shows how connectivity to SAP servers is established:

Figure 1.12 Establishing Connectivity to an SAP Server



For a detailed example of an SAP server connection, see "Establishing Connectivity to an SAP Server" on page 51.

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-based 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 Integration Server and SAS Data Quality Server

Certain enterprise software bundles for the SAS Intelligence Platform include data quality software from SAS and from DataFlux (a SAS company). The 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.

The data quality product from SAS is SAS Data Quality Server, which consists of SAS language elements and a Quality Knowledge Base from DataFlux. 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 Integration Server.

The data quality software from DataFlux consists of the DataFlux Integration Server, a second Quality Knowledge Base, and the dfPower Studio software. The DataFlux Integration Server runs jobs and real-time services that are created in dfPower 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 dfPower Studio software enables you to create jobs and real-time services and customize the data definitions in Quality Knowledge Bases.

SAS Data Integration Studio provides enabling software for data quality applications. Four data quality transformations enable you to analyze data, cleanse data, or trigger the execution of DataFlux jobs or real-time services on DataFlux Integration Servers.

The data quality software from SAS and DataFlux requires setup and configuration after installation. For administrative information, see “Administering SAS Data Integration Studio” in the *SAS Intelligence Platform: Desktop Application Administration Guide*.