Overview: The SAS ODBC Driver

This book documents SAS 9.1 of the SAS ODBC driver. If you are using the Version 8 driver, you should see *SAS ODBC Driver User's Guide and Programmer's Reference, Version 8*.

This book is intended for three audiences:

- users who use the SAS ODBC driver to access data that is stored on their own computers*
- system administrators who use the SAS ODBC driver to enable multiple users to access shared data on a remote machine
- applications programmers and others who need detailed information about how the SAS ODBC driver is implemented.

This book assumes that you are familiar with the Microsoft Windows NT, Windows 2000, or Windows XP operating environments and that you know how to use a mouse and keyboard to perform common Windows tasks.

* See “Types of Data Accessed with the SAS ODBC Driver” on page 3 for information about what kinds of data you can access.
What Is ODBC?

ODBC stands for Open Database Connectivity. It is an interface standard that provides a common application programming interface (API) to access databases. Many software products that run in the Windows operating environment adhere to this standard so that you can access data that other software has created.

ODBC functionality is provided by three main components:

- the client application
- the ODBC driver manager
- the ODBC driver.

Figure 1.1 on page 3 displays components of ODBC functionality. The ODBC driver manager, which was developed by Microsoft, manages the interaction between a client application and one or more ODBC drivers.

What Is the SAS ODBC Driver?

The SAS ODBC driver is an implementation of the ODBC standard that enables you to access, manipulate, and update SAS data sources from applications that are ODBC compliant. As Figure 1.1 on page 3 shows, the SAS ODBC driver uses a SAS server to access data from your SAS data sources. The SAS server can be either a local, iconified SAS session or a remote SAS/SHARE server. (See “SAS Servers” on page 6 for more information.) If you use other ODBC drivers (such as those for ORACLE or SQL Server) to access other data sources, those drivers might require additional software components.

Note: To access ODBC data sources from within SAS (the opposite of what the SAS ODBC driver enables you to accomplish), you must license the SAS/ACCESS interface to ODBC. For additional information see SAS/ACCESS for Relational Databases: Reference.
Types of Data Accessed with the SAS ODBC Driver

In Figure 1.1 on page 3 and elsewhere throughout this document, the term SAS data sources is used to describe data sources that you have defined in the SAS ODBC driver. These can include SAS data sets, flat files, and VSAM files, as well as data from many database management systems (DBMSs) through the use of SAS/ACCESS software. (See “SAS Data Sets” on page 5 for details.)

If your personal computer is connected to a TCP/IP network, you can access both local data sources and remote data sources. (See “What Software Do I Need?” on page 8 for information about software requirements.) Local data is data that you access through a SAS server on your local machine. The data can be stored either on your computer’s hard drive or on a network file system such as a Netware or Windows NT
file server, that makes the physical location of the data transparent to applications. Remote data is data that you access through a SAS/SHARE server that runs on another (remote) machine.

The ability to use the SAS ODBC driver in conjunction with SAS/ACCESS software as a gateway to DBMS data is particularly useful under any of the following circumstances:

- There is no ODBC driver available for the DBMS. In this situation, you can use the SAS ODBC driver in conjunction with SAS/ACCESS on the SAS server to connect to the DBMS.
- You do not license the necessary software (either an ODBC driver for the DBMS or DBMS network access software) on your client machine.
- You want to join or merge DBMS data with other data.

Currently, SAS/ACCESS software is available for the following systems, including:

ADABAS
CA-Datacom/DB
CA-IDMS
CA-OpenIngres
DB2 under OS/390, UNIX, and Windows
IMS-DL/I
Informix
Microsoft SQL Server
ODBC
OLE DB Providers
ORACLE
Oracle Rdb
PC File Formats*
SYBASE
SYSTEM 2000 Data
Teradata

*This includes Lotus 1-2-3, Microsoft Excel (5, 7/95, 97, 2000, 2002), and Microsoft Access (97, 2000, 2002) file formats.

Note: You can use the SAS/ACCESS Interface to ODBC to access many sources that provide an ODBC driver, including AS/400, SQL Server, and Microsoft Access data.

For information about the individual SAS/ACCESS interfaces, see “Recommended Reading” on page 49.
Introducing the SAS ODBC Driver

Understanding SAS

To use the SAS ODBC driver, you should understand three components of SAS:

- SAS data sets
- SAS data libraries
- SAS servers.

You should also understand how ODBC terminology corresponds to SAS terminology. See “SAS Terminology” on page 8 for more details.

SAS Data Sets

A SAS data set is a file structured in a format that SAS can access. The physical object contains:

- data values that are stored in tabular form
- a descriptor portion that defines the types of data to SAS.

The physical locations of the data values and the descriptor are not necessarily contiguous.

SAS data sets have two forms: data files and data views. SAS data files are essentially relational tables with columns (or variables) and rows (or observations). The SAS data file structure can have many of the characteristics of a database management system, including indexing, compression, and password protection.

SAS data views are definitions or descriptions of data that resides elsewhere. They enable you to use SAS to access many different data sources, including flat files, VSAM files, and DBMS structures, as well as native SAS data files. A data view eliminates the need to know anything about the structure of the data or the software that created it. Data views take up very little storage because they contain no data. They also access the most current data from their defined sources because they collect the actual data values only when called.

You can use data views to define subsets of larger structures, or supersets of data that have been enhanced with calculated values. You can also create SAS data views that combine views of dissimilar data sources. For example, you can combine a view of a relational DB2 table with a view of a SAS data file, a view of hierarchical IMS-DL/I data, or even a view from a PC-based dBASE file.

You create SAS data views in three ways:

- with the DATA step (DATA step views)
- with the SQL procedure (PROC SQL views)
- with the ACCESS procedure (SAS/ACCESS views).

You can also use SAS/ACCESS software to work directly with DBMS tables such as DB2 and ORACLE, as if they were SAS data sets and views by using the SAS/ACCESS LIBNAME statement. See SAS/ACCESS for Relational Databases: Reference for more information.

Some variation occurs among these types of views:

DATA step views describe data from one or more sources, including flat files, VSAM files, and SAS data sets (either files or other views). You cannot use a DATA step view to update the view’s underlying data because DATA step views only read other files.

For more information about how to create and use DATA step views, see the chapter “SAS Data Views” in SAS Language Reference: Concepts.

PROC SQL views
define either a subset or a superset of data from one or more SAS data sets. These
data sets can be files or views, and can include data sets composed of DBMS data
that are created with the SAS/ACCESS LIBNAME statement, and views that are
created with the PROC SQL Pass-Through Facility to access DBMS data. You can
also construct PROC SQL views of DBMS data by using an embedded LIBNAME
statement.

For example, the SQL procedure can combine data from PROC SQL views,
DATA step views, and SAS/ACCESS views with data in a SAS data file. You
cannot use PROC SQL views to update the data in the view’s underlying files or
tables. However, with some restrictions, you can use the UPDATE, DELETE, and
INSERT statements in the SQL procedure to update data that is described by
SAS/ACCESS views.

For information about the PROC SQL Pass-Through Facility, see the chapter
“The SQL Procedure” in Base SAS Procedures Guide, or SAS/ACCESS for
Relational Databases: Reference.

SAS/ACCESS views
describe views that are created with the ACCESS procedure in SAS/ACCESS
software. They are usually called view descriptors to distinguish them from PROC
SQL or DATA step views. You can use them to describe data from any DBMS for
which you license a SAS/ACCESS interface (see the list in “Types of Data Accessed
with the SAS ODBC Driver” on page 3). Each view descriptor describes all or
some of the data in one DBMS table or in one DBMS view. Some SAS/ACCESS
interfaces do not permit you to use a view descriptor to update the underlying
DBMS data.

For more information about view descriptors, see SAS/ACCESS for Relational
Databases: Reference.

SAS Data Libraries
SAS data sets are contained in data libraries. Each SAS data library has two
names: a physical name and a logical name (libref). The physical name of the library
fully identifies the directory or operating system data structure that contains the data
sets. Therefore, the physical name must conform to the rules for naming files within
your operating system.

You use the libref to identify a group of data sets (files or views) within SAS. The
libref is a temporary name that you associate with the physical name of the SAS data
library during each SAS job or session. After the libref is assigned, you can read,
create, or update files in the data library. A libref is valid only for the current SAS job
or session and you can reference it repeatedly within that job or session.

You can also use SAS/ACCESS software to associate a SAS libref with a DBMS
database, schema, server, or group of tables and views, such as a DB2 database or
group of ORACLE tables and views. See SAS/ACCESS for Relational Databases:
Reference for more information about using SAS/ACCESS software.

For more information about SAS data libraries, see the chapter “SAS Data Libraries”

SAS Servers
To access your SAS data sources, the SAS ODBC driver uses a SAS server. A SAS
server is a SAS procedure (either PROC SERVER or PROC ODBCSERV) that runs in
its own SAS session. It accepts input and output requests from other SAS sessions and
Introducing the SAS ODBC Driver

Converting DDE Servers to TCP Servers

From the SAS ODBC driver on behalf of the applications that are ODBC compliant. While the server is running, the SAS session does not accept input from the keyboard.

Note: Beginning with Version 7 of SAS, the SAS ODBC driver uses the TCP/IP protocol to communicate with both local servers and remote SAS/SHARE servers. Support for the DDE protocol has been discontinued. Therefore, you should reconfigure your driver settings to change any DDE (local or network) servers to TCP/IP servers, as described in “Converting DDE Servers to TCP Servers” on page 7.

The type of server that the driver uses depends on whether you are accessing local data or remote data:

local data
The driver uses a SAS ODBC server to access your data. If you do not already have a SAS session running on your computer, the driver starts a SAS session and executes PROC ODBCSERV, thereby automatically starting the server when you connect to your local data source. See “Accessing Local SAS Data Sources” on page 26 for more information. If you have a SAS session (but not a SAS ODBC server) running on your computer, then you must either start the SAS ODBC server manually or end the SAS session before you connect to your SAS data sources (except under Windows NT, which supports multiple concurrent SAS sessions). See “Starting a SAS ODBC Server” on page 26 for details.

remote data
The driver uses a SAS/SHARE server. This requires that you license the SAS/SHARE software on the remote host. The driver also requires TCP/IP software that is included with your operating environment to communicate with the server. Your server administrator uses PROC SERVER to start the server on your remote host. See “Accessing Remote (SAS/SHARE) Data Sources” on page 27 for more information.

The SAS ODBC driver can communicate with a SAS/SHARE server or a Scalable Performance Data (SPD) server. You can interchange SAS data and SPD server data by using the LIBNAME statement engine option in either SAS or SPD server.

Scalable Performance Data Server

The Scalable Performance Data (SPD) server utilizes the latest parallel processing methods and data server capabilities to efficiently access large volumes of data and serve large numbers of concurrent users. It provides efficient data access for hundreds of users across multiple processors.

The SPD server allows access to SAS data for intensive processing (queries and sorts) on the host server machine. It organizes and processes SAS data to take advantage of parallel processors on specific host servers.

The SPD SNET server handles communication between SAS clients and an SPD server. You must have the SPD server licensed on your client machine. Then you can interchange SAS data and SPD server data by using the LIBNAME statement engine option either in SAS or on the SPD server. For more information, see Scalable Performance Data Server User’s Guide.

Converting DDE Servers to TCP Servers

If you still use SAS servers that are configured to use DDE, you must convert them to TCP servers to use them with this release of the SAS ODBC driver.

To convert a DDE (or Network DDE) server to a TCP server, perform these steps:
1. Install the new SAS ODBC driver.
2. Select the Servers tab from the SAS ODBC Driver Configuration dialog box.
3. Select the existing DDE server that you want to convert to TCP in the Servers list to the left of the Servers page.
4. Select **Configure** to verify that the options are correct, and then select **OK**.

   *Note:* If you are converting from a local DDE server, you need to remove the `-comamid dde` option from the SAS Startup Parameters text box.

5. Select **Update** and then **OK** to store your changes.

   *Note:* For local TCP servers you will also have to verify that your TCP/IP Services file has an appropriate entry for the server that you are configuring. For more information, see “The TCP/IP Services File” on page 29.

---

**SAS Terminology**

Software products often include similar components or constructs that are known by different names. For the ODBC standard and SAS, the following correspondences exist:

<table>
<thead>
<tr>
<th>ODBC term</th>
<th>SAS term</th>
</tr>
</thead>
<tbody>
<tr>
<td>owner</td>
<td>library name (libref)</td>
</tr>
<tr>
<td>table</td>
<td>data set</td>
</tr>
<tr>
<td>qualifier</td>
<td>not used</td>
</tr>
</tbody>
</table>

Therefore, if your application that is ODBC compliant asks you to specify the owner for a SAS data library, you should specify the libref. If the application asks for a table name, you should specify the name of the SAS data set. If a qualifier is requested, you can usually leave the field blank.

---

**What Software Do I Need?**

The SAS ODBC driver runs under 32-bit Windows (Windows NT, Windows XP, and Windows 2000). Other software requirements depend on your hardware configuration and on the data source(s) that you want to access, as shown in Table 1.1 on page 9.

*Note:* The 16-bit version (shipped with Releases 6.10, 6.11 and 6.12 of SAS) is downloadable from the SAS Web site and runs under Windows 3.1, 32s, and Windows for Workgroups.
Introducing the SAS ODBC Driver

### Table 1.1 Software Requirements

<table>
<thead>
<tr>
<th>Data Source(s)</th>
<th>Configuration</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>local SAS data only</td>
<td>Stand-alone PC</td>
<td>Base SAS*</td>
</tr>
<tr>
<td>local SAS data and</td>
<td>Stand-alone PC</td>
<td>Base SAS*</td>
</tr>
<tr>
<td>local DBMS data</td>
<td>PC on a network</td>
<td>SAS/ACCESS interface for each DBMS TCP/IP**</td>
</tr>
<tr>
<td>remote SAS data</td>
<td>PC on a network</td>
<td>On PC: SAS ODBC driver*** TCP/IP**</td>
</tr>
<tr>
<td></td>
<td></td>
<td>On remote host: Base SAS SAS/SHARE TCP/IP**</td>
</tr>
<tr>
<td>remote SAS data and</td>
<td>PC on a network</td>
<td>On PC: SAS ODBC driver*** TCP/IP**</td>
</tr>
<tr>
<td>remote DBMS data</td>
<td></td>
<td>On remote host: Base SAS SAS/SHARE SAS/ACCESS interface for each DBMS TCP/IP**</td>
</tr>
</tbody>
</table>

* For local data access, base SAS software includes the SAS ODBC driver, but you install the driver separately.

** Requires a version of TCP/IP that supports the Microsoft Windows socket (“winsock”) API.

*** For remote data access, install the driver on your client PC. The SAS license for the remote host must include SAS/SHARE.
CHAPTER 2

Defining Your Data Sources

Introduction to Defining Data Sources 11
Accessing the SAS ODBC Driver Dialog Boxes 11
Setting Up Your Data Source 14
   Specifying SQL Options 15
Defining Servers 16
   Deleting a Server Definition 20
   Modifying a Server Definition 20
Defining Data Libraries 20
   Defining Libraries at Server Startup Time 22
   Deleting a Data Library Definition 22
   Modifying a Data Library Definition 23
Saving a Data Source Definition 23
Modifying a Data Source Definition 23
Specifying a Different Server for a Defined Data Source 23
Deleting a Data Source Definition 24

Introduction to Defining Data Sources

After you install the SAS ODBC driver,* you must provide information about the data source(s) that you want to access. This chapter explains how to use the SAS ODBC driver dialog boxes to provide the necessary information.

From the SAS ODBC dialog boxes, you can select Help at any time to obtain information about the active dialog box.

Accessing the SAS ODBC Driver Dialog Boxes

To access the SAS ODBC dialog boxes, complete these steps:

1 Access the Microsoft Windows Control Panel window by selecting Start ➤ Settings ➤ Control Panel

Find the ODBC Data Sources or ODBC Administrator icon. This icon might be located in the Control Panel group, in an ODBC group, or, if you have installed a package of other ODBC drivers, it might be in a group that is associated with that package.

* See the installation instructions that are shipped with the driver diskette.
2 Double-click on the icon. Depending on the version of ODBC Administrator that you have installed (Windows NT 4.0 is shown below), you might see a slight variation in the following dialog box:

**Display 2.1** ODBC Data Source Administrator

3 To add a new data source, select either the User DSN tab or the System DSN tab and select [Add]. The Create New Data Source dialog box appears.
To select the SAS ODBC driver, scroll down the list of Installed ODBC drivers (if necessary) and select **SAS**. Click **Finish**.

**Note:** Select **SAS** even if you are going to use a SAS data view to access data that is not SAS. See “SAS Data Sets” on page 5 for information about SAS data views.

The SAS ODBC Driver Configuration dialog box appears.
Display 2.3 SAS ODBC Driver Configuration Dialog Box

At the top of this dialog box are three tabs: General, Servers, and Libraries. Select these tabs to move from one page to the next. On each page, supply information about the data source that you want to be able to access.

After you have supplied all the necessary information, select OK from any page to save your data source definition. Selecting Cancel discards any new or changed information that you have supplied and returns you to the ODBC Data Source Administrator dialog box (Display 2.1 on page 12).

Setting Up Your Data Source

When you first access the SAS ODBC Driver Configuration dialog box, the General page is in the foreground, as shown in Display 2.3 on page 14. Follow these steps:

1. In the Data Source Name field, enter a name for the data source that you want to access. The name must begin with a letter, and it cannot contain commas, semicolons, or any of the following special characters: [ ] { } ( ) ? * = ! @. For example, if you are defining SAS data that is stored on a machine named CICERO, you might call your data source SAS_CICERO. If you or other users are concerned only with the type of data (or with the type of application that uses that data), and not with where the data is stored, then you might have data sources with names like Finance or Payroll.

2. (Optional) In the Description field, supply a description of the data source.

3. The Server field includes a drop-down list that you select in order to expand the list of defined servers. The first time you define a data source, the list is empty.
Define one or more servers (as described in the next section), and then come back to the General page to make a selection from the Server list. You must specify a server for every data source.

4 In the Records to Buffer field specify the number of rows to request from the SAS/SHARE server in a single transmission. The default value is 100 and the maximum value is 32,000.

The size of the server’s transmission buffers limits the actual number of rows returned. Usually you will not change the default value. However, specifying a larger value might improve performance when you retrieve a very large result set.

---

### Specifying SQL Options

SQL options affect the interaction between the SAS ODBC driver, SAS, and applications that are ODBC compliant. The default settings for the options are those that most applications that are ODBC compliant will expect and will work best with. However, you can override the defaults by selecting any of the SQL Options listed. Select the box next to the desired option.

**Preserve trailing blanks**

preserves trailing blanks at the end of character fields. The default action is that trailing blanks are removed, so that each field ends in a null value.

**Support VARCHAR**

causes character fields that are longer than 80 characters to be reported as variable-length fields, and causes the trailing blanks to be removed. See “Support VARCHAR Option” on page 41 for more information.

**Infer INTEGER from FORMAT**

causes SAS numeric data types (typically reported as SQL_DOUBLE) to be reported as SQL_INTEGER. See “Infer INTEGER from FORMAT Option” on page 41 for more information.

**Disable _0 override parsing**

prevents the SAS ODBC driver from removing the _0 string from SQL queries. See “Disable _0 override parsing Option” on page 41 for more information.

**Return SQLTables REMARKS**

causes the SAS ODBC driver to read and return the SAS data set label for each data set in the library you are accessing. (SQLTables is the name of an ODBC function that can be used for this purpose.) For SAS data sets, this can have a negative impact on performance, because each data set must be opened in order to read the label. Therefore, you should not select this option unless there is information in the label that you really need to see.

**UNDO_POLICY=REQUIRED**

implements the UNDO_POLICY option of SAS SQL procedure with a setting of REQUIRED. With this setting, INSERT or UPDATE operations that fail are undone. However, this action is performed only for operations that affect multiple records; a statement that affects a single record behaves the same regardless of the UNDO_POLICY setting. When UNDO_POLICY=REQUIRED, the associated statement handle (hstmt) of an UPDATE or INSERT statement must be the only active hstmt against the table. If another user, or an hstmt within the same user’s application, has an active SELECT statement, then the UPDATE or INSERT statement fails.
Defining Servers

The SAS ODBC driver uses a SAS server to access your data sources. To access local data, the driver uses a SAS ODBC server. To access remote data, it uses a SAS/SHARE server. (See “SAS Servers” on page 6 for more information.)

This section explains how to provide the SAS ODBC driver with necessary information about the server(s) you will be using.

1 Select the Servers tab in the SAS ODBC Driver Configuration dialog box to go to the Servers page. Supply the information described in the following steps. If at any time you want to clear all of the fields on the right side of the dialog box and start again, select Clear.

Display 2.4 Servers Page

2 In the Name field, enter a name for the SAS server that you are defining. If you supply a one-part name such as shr1, the SAS ODBC driver infers that the server is a local SAS ODBC server. Local (Single User) appears in the SAS Server Type field.

Fuzz Numbers at N Places
specifies the degree of precision to use when comparing numbers. See “Fuzz Numbers at N Places Option” on page 41 for more information. By default, this option is selected. You can change the default value, 12, by typing over it.
If you supply a two-part name such as `machine.shr2`, the driver infers that the server is remote. **SAS/SHARE (Multi-User)** appears in the SAS Server Type field.*

The difference between local and remote servers is important and can cause errors if your configuration specifies the wrong type of server. See “SAS Servers” on page 6 for more information.

3 If user access to the server is password protected, then enter the password in the **Password** field. This should be the same password that was specified for the **UAPW=** option in PROC SERVER (for a SAS/SHARE server), or in PROC ODBCserv (for a local access server).

4 Select [Configure]. A configuration dialog box appears.

   If you specified a one-part server name in the **Name** field of the **Servers** page (Display 2.4 on page 16), then the SAS ODBC driver infers that you are using a local SAS server. The Local Options dialog box appears.

**Display 2.5  Local Options Dialog Box**

   ![Local Options Dialog Box]

When you access a local SAS data source from an application that is ODBC compliant, the SAS ODBC driver uses the information in this dialog box to invoke a SAS ODBC server, provided there isn’t one already running. (If a SAS ODBC server is already running, the driver finds it and connects to it.) Each field contains default values that you can change by typing over them.

**Path:**

This field specifies the fully qualified path name for the SAS executable file (SAS.EXE) that you use to start a SAS session. The default path is `C:\Program Files\SAS\SAS System\9.1\sas.exe**`. If this field is empty, then no attempt is made to start a SAS ODBC server when you connect to your data source.

---

* The SAS ODBC driver communicates with either a SAS/SHARE server or an SPD server. If you add a server that uses the SPD SNET server, **SAS Server Type** appears as **SAS/SHARE (Multi-User)**. This will not cause a problem using the SPD server.

The driver interprets the first part of the name as an alias for the TCP/IP network machine name and the second part as the server name. The server name must match both the **ID=** field in PROC SERVER (the SAS/SHARE server that is running on the remote host), and the **server-name** that you (or your server administrator) specified when you defined the server as a service in the TCP/IP Services file. See “The TCP/IP Services File” on page 29 for more information.

** The SAS ODBC driver queries the registry for the current version of SAS and the value of **DefaultRoot**. This value is concatenated with `\sas.exe` to create the default path. The default working directory is the value of DefaultRoot.
Working Directory:
This field specifies the fully qualified path name for the directory that you 
want to use as the SAS working directory. This directory is usually where 
your SAS program files and documents are located. The default is 
C:\Program Files\SAS\SAS System\9.1.

Startup Parameters:
This field specifies the parameters that are used to invoke SAS. The default 
values are: The initialization statement (-initstmt) executes a SAS macro 
(%sasodbc), which in turn invokes the ODBC server. The shr1 value is only 
an example. It is a SAS macro parameter whose value is taken from the 
name that you specified in the Name field of the Servers page (Display 2.4 on 
page 16). The -icon option specifies that the server session should be 
invoked in iconified mode, because no interaction with the server is required. 
The -nologo option (not shown) specifies that the SAS session will be 
invoked without displaying the SAS logo and copyright information.

The %sasodbc macro is shipped with SAS and is found in 
!SASROOT\CORE\SASMACRO\SASODBC.SAS.* The SASODBC.SAS file 
executes the SAS procedure PROC ODBCSERV.

The SASODBC.SAS file can be modified to add additional SAS options or 
SAS statements such as the LIBNAME statements mentioned in “Defining 
Libraries at Server Startup Time” on page 22. You can also specify options 
for PROC ODBCSERV. The available options are the same as those for PROC 

An additional option, LOG=QUERY, is particularly relevant for servers 
that are used by the SAS ODBC driver. This option causes the server to log 
SQL queries. (By default, the server logs update and output operations, but 
not queries.) Hence, this option is useful when you need to see the queries 
that the server receives from an ODBC client application.

If your SAS session is installed on a network drive and is shared by 
multiple users, then you probably don’t want individual users to modify the 
SASODBC.SAS file. Instead, users can make their own copies of the file and 
can store them in their personal libraries. In this case, they must also add 
the -sasautos option either to the Startup Parameters field or to their 
local CONFIG.SAS file to indicate the pathname for the library, as in the 
following example:

-sasautos c:\programs\sas

For more information about SAS system options and SAS statements, see 
SAS Companion for Windows.

Timeout:
This field specifies in seconds how long to wait for the local SAS ODBC server 
to start and to register itself. The default is 60 seconds.

If you specified a two-part server name before selecting [Configure], the 
SAS/SHARE Options dialog box appears.

* !SASROOT is a logical name for the directory in which you install SAS. For more information, see SAS Companion for 
  Windows.
Supply the requested information in these fields:

**Server Address:**
This field is automatically filled with the alias for the TCP/IP network machine name that you specified in the **Name** field of the **Servers** page (Display 2.4 on page 16). In a complex networking environment, you might need to supply a fully qualified domain address for the server (for example, `machine.sas.com`).

**User Name:**
This field is for your user ID on the system where the server is running. It is required if the server is running in secured mode; otherwise, it is ignored.

**User Password:**
This field is for your password on the system where the server is running. If you supply a **User Name** without a **User Password**, then you will be prompted for a password at connection time. The SAS ODBC driver automatically encrypts the password. For more information about password encryption, see “Encrypting Passwords” on page 31.

**Connect Options:**
This field provides additional connection options to use when you define a connection that will point to an SPD SNET server. You must leave **Connect Options** blank for a connection to a SAS/SHARE server to work properly.

An SPD SNET server is an intermediate component between the ODBC driver and an actual SPD server.

Specify the connection options as name value pairs delimited by white space. Each of the values must be in single quotation marks. For example,

`DBQ='tmp' HOST='bigone.unx.sas.com' SERV='5129'`

where:

- **DBQ** is the SPD server libname domain.
- **HOST** is the location of the host computer on your network where the SPD name server is running.
- **SERV** is the port number of the SPD name server running on the HOST. See the documentation shipped with the SPD server for more details.
Userid/Password Override
This field requests that the UID keyword and PWD keyword be used in the ODBC client application. The ODBC driver will pass the value of the PWD keyword as the user login password, and the value of the UID keyword as the user ID. Otherwise, the driver will use the value of the PWD keyword from the server definition as the authentication password. For more information about using this option, see “Userid/Password Override” on page 28.

5 When you have finished making any necessary changes to the Local Options or SAS/SHARE Options dialog box, select OK to return to the Servers page (Display 2.4 on page 16).

6 Important! Select <<Add<< to save your server definition. The server name now appears in the list of defined servers at the left.

7 To define another server, select Clear and then repeat steps 2-7.

Deleting a Server Definition
To delete a previously defined server, complete these steps:
1 Select the server name from the Servers list at the left of the Servers page.
2 Select >>Remove>>.

CAUTION:
If you delete a server, then any data sources that use that server will no longer be accessible until you redefine that server.

Modifying a Server Definition
To change the information for a previously defined server, complete these steps:
1 Select the server name from the Servers list at the left of the Servers page.
2 To make the desired changes, select Configure to make changes in the Local Options or SAS/SHARE Options dialog box (Display 2.5 on page 17 and Display 2.6 on page 19).

The Name field is dimmed to indicate that you cannot change the name of the server. The reason for this is that you may have already defined data sources that use that server—if you changed the server name, then you would no longer be able to access those data sources.
3 Select <<Update<< to save your changes.

For instructions on how to specify a different server for a data source that you have already defined, see “Specifying a Different Server for a Defined Data Source” on page 23.

Defining Data Libraries
Each data source can include multiple data libraries. (See “SAS Data Libraries” on page 6 for information.) Therefore, you provide information about each library that you want to access.

This section describes how to use the Libraries page to define your data libraries. See “Defining Libraries at Server Startup Time” on page 22 for information about an alternate way of defining data libraries.
1 Select the Libraries tab in the SAS ODBC Driver Configuration dialog box to go to the Libraries page (Display 2.7 on page 21). Supply the information described in the following steps. If at any point you want to clear all of the fields on the right side of the dialog box and start again, select Clear.

Display 2.7 Libraries Page

2 In the Name field, enter a name for an existing physical SAS data library that you want to access. (For those who are familiar with SAS, this field corresponds to the libref in the SAS LIBNAME statement.) The name can be up to eight characters long. The first character must be a letter or an underscore. Subsequent characters can be letters, numeric digits, or underscores. Blanks and special characters are not allowed. For example, you might use the name cost to designate a library of cost-accounting data.

The SAS data library can include SAS data files, SAS data views, or both. See “SAS Data Sets” on page 5 for more information.

Note: If you use an ODBC application that exports databases using one-level names, you will need to define a library called user.

3 In the Host File field, enter the physical name of the library. This must be a valid pathname for the operating system that your data library is stored in. For example, for a library that is stored on a PC in the Windows environment, c:\data\costacct would be a valid pathname.

4 (Optional) In the Description field, supply a description of the library, to remind yourself or other users of what the library contains.

5 (Optional) In the Engine field, enter the name of the SAS engine that is required for writing to and reading from this library. This is necessary only if you do not want the SAS server to use the default engine for the version and release number of SAS that you are running. (For SAS 9.1, the default engine would be v9.) For
information about other engines that might be available, see the description of the
LIBNAME statement in the SAS Companion for the operating system under which
your data library is stored.

6 (Optional) In the Options field, you can enter the following option for the library
that you are defining:

ACCESS=READONLY

This option limits users to “read-only” access to the SAS data sets in the
library.

7 Select "Add" to save your library information. The library name is added to
the Libraries list at the left.

8 To include another data library with your data source, repeat steps 2-7.

---

**Defining Libraries at Server Startup Time**

Server administrators might prefer to define SAS data libraries at server startup
time rather than defining them through the SAS ODBC driver dialog boxes. Defining
libraries at server startup time can make opening the data source faster. It also enables
you to avoid hardcoding the physical names of your libraries in your SAS ODBC
data-source definitions.

As explained in “SAS Servers” on page 6, the SAS ODBC driver uses a SAS/SHARE
server (invoked by PROC SERVER) to access remote data sources. It uses a SAS ODBC
server (invoked by PROC ODBCSERV) to access local data sources. To define a data
library at server startup time, you precede the PROC SERVER or PROC ODBCSERV
statement with a SAS LIBNAME statement. For example, you could define a library of
cost-accounting data to a SAS/SHARE server as follows:

```sas
libname cost 'e:\fin\acct';
proc server id=acctserv authenticate=optional;
run;
```

**Note:** Depending on whether the server is running in secured mode or not, the
authenticate=optional option might not be needed.

To define this library to a SAS ODBC server, you would add only the above
LIBNAME statement to the !SASROOT\CORE\SASMACRO\SASODBC.SAS file. See
Startup Parameters on page 18 for more information.

When a user requests access to the particular SAS ODBC data source from an ODBC
client application, the server would automatically make this library available, along
with any libraries that were defined via the SAS ODBC Libraries page.

For more information about the SAS LIBNAME statement, see the SAS Companion
for the operating system under which your data library is stored.

---

**Deleting a Data Library Definition**

To delete a previously defined data library, complete these steps:

1 Select the library name from the Libraries list at the left of the Libraries page
   (Display 2.7 on page 21).

2 Select "Remove".
Modifying a Data Library Definition

To change the information for a previously defined library, complete these steps:
1. Select the library name from the Libraries list on the Libraries page.
2. Make the desired changes to the Host File, Description, Engine, and Options fields.
3. Select <<Update>> to save your changes.

Saving a Data Source Definition

When you have finished defining your data source, select OK to save your data source definition and return to the ODBC Data Source Administrator dialog box (Display 2.1 on page 12). Your newly defined data source now appears in the list of Data Sources on the left.
You can either add additional data sources, or select OK to return to the Microsoft Windows Control Panel (or to the location from which you entered the ODBC Data Source Administrator dialog box).

Modifying a Data Source Definition

To modify a previously defined data source, complete these steps:
1. Select the name of the data source from the ODBC Data Source Administrator dialog box (Display 2.1 on page 12).
2. Select Configure. The SAS ODBC Driver Configuration dialog box (Display 2.3 on page 14) appears.
3. Use the tabs in this dialog box to access other SAS ODBC pages. Use the other pages as described in previous sections to modify server definitions, library definitions, or SQL options.
4. When you are finished, select OK to save your changes.

Specifying a Different Server for a Defined Data Source

Suppose a data source named Payroll has been moved from a server named CICERO to one named DAVINCI. In this case, you need to change the server that you specified for the Payroll data source by completing these steps:
1. Select the Payroll data source from the ODBC Data Source Administrator dialog box (Display 2.1 on page 12).
2. Select Configure. The SAS ODBC Driver Configuration dialog box (Display 2.3 on page 14) appears, with CICERO listed in the Servers field.
3. Use the Servers page to define the DAVINCI server, if it is not already defined.
4. From the General page, if DAVINCI is not already visible in the Servers field, select the drop-down list icon next to that field, and then select DAVINCI from the list.
5 Select **OK** to save your changes.

*Note:* The server name specified must be defined in the TCP/IP Services file. For more information, see “The TCP/IP Services File” on page 29.

---

**Deleting a Data Source Definition**

To delete a previously defined data source, complete these steps:

1. Select the name of the data source from the ODBC Data Source Administrator dialog box (Display 2.1 on page 12).
2. Select **Remove**.
CHAPTER 3

Using the SAS ODBC Driver

Introduction to Using the SAS ODBC Driver

This chapter provides an overview of how to use the SAS ODBC driver to access your SAS data sources. It also provides information about the SAS servers and the communications software that are used by the driver and about SQL syntax and return codes that are supported by the driver.

Accessing Your Data Sources

The details of how you access your SAS data sources depend on which ODBC-compliant application you are using (for example, Microsoft Access or Excel), and on whether you are accessing local or remote data. However, here are the steps that you follow:

1. Install the SAS ODBC driver. (Refer to the installation instructions that are shipped with the driver.) The installation program provides the SAS driver for the ODBC Data Source Administrator to use.

2. Double-click on the Data Sources (ODBC) icon in the Control Panel window to access the ODBC and SAS ODBC Data Source Definition dialog boxes.
3 Use these dialog boxes to provide the SAS ODBC driver with the necessary information about your SAS data source(s). (See Chapter 2, “Defining Your Data Sources,” on page 11 for instructions.) In addition to SAS data files, SAS data sources can include DATA step views, PROC SQL views, or SAS/ACCESS views, all of which contain definitions of data that is stored elsewhere rather than the physical data itself. (See “Types of Data Accessed with the SAS ODBC Driver” on page 3 and “SAS Data Sets” on page 5.)

4 The next step depends on whether you want to access local data or remote data:

   Note: TCP/IP (Winsock) is required for accessing local data sources, remote data sources, or both. △
   □ To access local data see “Accessing Local SAS Data Sources” on page 26.
   □ To access remote data sources, see “Accessing Remote (SAS/SHARE) Data Sources” on page 27.

5 Consult the documentation for your Windows application for instructions on how to access or import data from other sources. From the list of available data sources, select or specify the name that you assigned to your SAS data source.

6 Select or enter the name of the desired SAS data library (if required by your application).

7 Select or enter the name of the desired SAS data file or view.

For information about configuring file DSNs for applications that are ODBC compliant that only support the use of file DSNs (such as Microsoft Excel 97), see the online Help that is available when you configure data sources in the ODBC Data Source Administrator and press [Help].

---

Prerequisites for Communications

The SAS ODBC driver interacts with other software to provide you with access to your data sources. Certain prerequisites apply, depending on whether you want to access local data or remote data. These prerequisites are described in the following sections.

Accessing Local SAS Data Sources

To access local SAS data sources, the SAS ODBC driver uses a SAS ODBC server in conjunction with TCP/IP. You must edit your TCP/IP Services file to define your servers before starting the SAS ODBC server. It is not necessary for the server to be running when you define your data sources. However, the server must be running on your PC in order for you to access your SAS data sources. For information about editing the TCP/IP Services file, see “The TCP/IP Services File” on page 29. For more information about SAS servers, see “SAS Servers” on page 6.

Starting a SAS ODBC Server

If there isn’t already a SAS ODBC server running on your PC, the SAS ODBC driver uses the information that you supplied in the Local Options dialog box (described in Display 2.5 on page 17) to automatically start one for you. That is, you do not need to take any action to start the server.

If you already have a SAS session running on your PC, then you can start the SAS ODBC server in that session by submitting the following statements:
options comamid=tcp;
proc odbcserv id=server-name authenticate=optional;
run;

The server-name must be the same as the name you specified on the Servers page when you defined your local data source, as explained in “Defining Servers” on page 16. Alternatively, you can terminate your SAS session so that the SAS ODBC driver can start a SAS ODBC server for you in a new SAS session.*

Note: When the server is running in a SAS session, the SAS session does not accept user input from the keyboard. △

If the SAS session cannot be started before the SAS Timeout value that you specified in the Local Options dialog box is reached, a time-out error is returned to your ODBC client application. An error message is also returned to the client if the SAS session was started but PROC ODBCSERV could not execute.

Terminating the SAS ODBC Server

When you are finished using your ODBC client application to access your local SAS data sources, the SAS ODBC server continues to execute in case you want to access additional SAS data sources. To terminate the server, do either of the following tasks:

- Open the Windows Task List window and press the Ctrl, Alt and Delete keys simultaneously and select Task Manager from the Windows NT Security dialog box. You can also use the Ctrl, Shift, and Esc keys to immediately invoke the Task Manager. Under the Applications tab, select SAS from the list of tasks, and then select End Task. Alternatively, under the Processes tab, select SAS.EXE in the Image Name column and select End Process.
- Bring the SAS session into focus (make it the active window), and then press the Ctrl and Break keys simultaneously. The Task Manager appears. Select ODBCSERV from the list of tasks and click OK. Click OK again to halt the procedure.

Accessing Remote (SAS/SHARE) Data Sources

To access remote data sources, the SAS ODBC driver uses a SAS/SHARE server. (See “Accessing Remote (SAS/SHARE) Data Sources” on page 27 for more information. See SAS/SHARE User’s Guide for complete information about SAS/SHARE.) It is not necessary for the server to be running when you define your data sources. However, the server must be running on your remote host in order for you to access your data sources.

Because a SAS/SHARE server is used by multiple users, it is usually invoked on the remote host at system startup time. Therefore, users usually do not need to take any local action to invoke the server.

Whoever is responsible for starting the SAS/SHARE server (you or your server administrator) must complete these steps:

1 Define the server(s) in the TCP/IP Services file for the client machine(s) that are running the server(s).* (For more information, see “The TCP/IP Services File” on page 29.)
2 Create a SAS program that contains the following SAS statements:

* Under Windows NT, it is not necessary to terminate the SAS session because you can run multiple SAS sessions in that environment.
* Refer to the documentation for your TCP/IP software to find the path name for the Services file.
options comamid=tcp;
proc server id=server-name authenticate=optional;
run;

Invoke this program to run in the background using proper host-specific command syntax. See Communications Access Methods for SAS/CONNECT and SAS/SHARE for additional information.

Userid/Password Override

The Userid/Password Override option is in the SAS/SHARE Options dialog box of the Servers page. This option enables you to use the UID keyword and PWD keyword in the ODBC client application so that the driver passes the value of the PWD keyword as the user login password and the value of the UID keyword as the user ID. Otherwise, the driver will use the value of the PWD keyword from the server definition as the authentication password.

By default, the SAS ODBC driver uses the value of the ODBC PWD keyword to mean the SAS/SHARE server authentication password (UAPW= option on PROC SERVER). However, most ODBC users intuitively associate the PWD keyword with a user login password such as a Windows NT or UNIX user login password. This can pose a problem if you connect to a remote secure server and you want to specify a different user ID and password at runtime.

The following example starts a SAS/SHARE server and uses the PWD option as the server authentication password:

/* SAS/SHARE User Access password is set */
/* System username and password not required */
proc server id=shr1 UAPW=sapw authenticate= opt;
run;

If an ODBC DSN named is created with the server authentication password field in the server definition set to pwsa, then the connections that use SampleDSN will fail. This is because the password field does not match the PROC SERVER value. ODBC client applications can override keyword defaults at connect time by specifying a PWD= option in the program.

The following SAS code uses SAS/ACCESS for ODBC to connect successfully to the DSN SampleDSN by overriding the PWD keyword:

/* The PWD= option value matches the UAPW= option for remote PROC SERVER, so the connection should succeed. */
proc sql;
   connect to odbc (dsn=SampleDSN pwd=sapw);
   select * from connection to odbc
       {select * from dictionary.columns};
quit;

Instead of using just the PWD value as the SAS/SHARE server access password, you can use the Userid/Password Override option to force the ODBC driver to use the UID and PWD values for authentication at the system level. The following example demonstrates this option when you start a SAS/SHARE server and configure the TCP/IP communications access method to require authentication for all connecting clients:

/* Start a remote secured SAS/SHARE server that requires */
/* requires system username and password from the client. */
/* the SAS/SHARE user access password is not required */
%let tcpsec=_secure_;
proc server id=shr1 authenticate=req;
run;

If the ODBC DSN is named SampleDSN and the user login ID of iamuser and user login password of iampw are stored in the server definition, then you can use the following SAS program to connect to the DSN SampleDSN:

```sas
proc sql;
    connect to odbc (dsn=SampleDSN);
    select * from connection to odbc
    (select * from dictionary.columns);
quit;
```

Perhaps your system administrator requests not to store user names and passwords in DSNs, or your password on the remote system changed from iampw to iarepw. The ODBC specification allows ODBC client applications to override the user ID and password with the UID option and the PWD option in the source code. Now you can use the following SAS program to connect to the DSN SampleDSN:

```sas
proc sql;
    connect to odbc (dsn=SampleDSN uid=iamuser pwd=iarepw);
    select * from connection to odbc
    (select * from dictionary.columns);
quit;
```

If you enable the Userid/Password Override option in the SAS/SHARE Options dialog box for SampleDSN, then the driver will recognize UID and associate PWD with the user login password. The above connection will succeed because the server does not use the server authentication password. However, by default, this program fails because the SAS ODBC driver does not recognize the UID keyword. Instead, the driver associates PWD with the SAS/SHARE server authentication password.

---

**The TCP/IP Services File**

The TCP/IP Services file contains information about the names of the available services on the machine, along with the port number, protocol name, and any aliases corresponding to each service. For the purposes of the SAS ODBC driver, each entry in this file associates a SAS server (service name) with the port number and protocol used by that service. The location of the Services file varies on different platforms. To configure the SAS ODBC driver correctly, you must locate the Services file for your platform. Common locations for the Services file are

- **Windows**: C:\WINNT\SYSTEM32\DRIVERS\ETC\SERVICES
- **UNIX**: /etc/services

Entries in the services file take the following general form:

```
<official service name> <port number/protocol name> <aliases> #
<comments>
```

*Note:* You must update the services file on both the server and ODBC client machines when you access remote SAS data sources.
Editing the TCP/IP Services File

To configure your Services file for use with the SAS ODBC driver, you must add an entry to the services file for each SAS server (either local or remote) that you have configured using the ODBC Administrator.

The port number that you use should be an unused port number in this file (for larger networks, contact your network administrator to obtain an available port number). The port number must be greater than 1024, as any port number equal to or less than 1024 is reserved. The protocol must always be tcp. The server name must be from one to eight characters long. The first character must be a letter or an underscore; the remaining seven characters can include letters, digits, underscores, the dollar ($) sign, or the at (@) sign. You specify the same server name on the Servers page when you define your data source as explained in “Defining Servers” on page 16.

For example, if, in the ODBC Administrator, you have configured a local ODBC server named shr1 and a remote SAS/SHARE server named machine.shr2 you would add entries to the Services file similar to the following entries (substituting the appropriate port numbers):

shr1 5010/tcp #Local ODBC Server
shr2 5011/tcp #Remote SAS/SHARE Server

Note: In the case of shr2, the administrator of the remote system named machine should have already edited the Services file on that system to include the same shr2 entry and should have started the SAS/SHARE server.

Using Data Sets That Have One-Level Names

If you use an ODBC application such as Microsoft Access that exports databases using one-level names, you should use the ODBC administrator to define a USER library. SAS usually places any data set that has a one-level name into the WORK library, which is deleted at the end of the SAS session. But if a USER library has been defined, SAS places all one-level name data sets into the USER library, which is saved at the end of the SAS session. In a multi-user environment, multiple client connections to a SAS server can each have their own USER library defined.

Updating Attached Tables

Some Microsoft products that are based on the JET engine (such as Microsoft Access) have certain requirements in order to be able to update database tables. This might be true of other ODBC applications as well. These requirements might make it necessary for you to specify two SQL options when you define your SAS data sources.

- The attached table must have a unique primary key that is not a floating-point value. You can use the Data Source SQL Option Infer INTEGER from FORMAT to indicate that SAS numeric fields without fractional parts (for example, FORMAT(n,0) where n is less than 12) are actually integer values that can be used to index the table.
All of the values in a row might be used to uniquely select the row for updating. This can be a problem in rows that contain floating-point fields (SAS numerics). Insignificant differences in values can be caused either by differences in floating-point representation on different machines or by conversion between character and binary formats. By specifying the Data Source SQL Option **Fuzz Numbers at 12 places**, you can cause WHERE clauses to select values that are “acceptably close” rather than requiring exact comparisons.

See “Setting Up Your Data Source” on page 14 and “User-Specified SQL Options” on page 40 for more information about these SQL options.

### Using SQL Statements to Access SAS Data Sources

All applications that are ODBC compliant use a variety of the Structured Query Language (SQL) to access and manipulate data. However, most of these applications transform your actions into SQL statements so that you do not need to know anything about SQL.

If your application requires you to use SQL statements, or if you use SQL out of personal preference, then you should refer to the chapter “The SQL Procedure” in *Base SAS Procedures Guide*. The elements of SQL grammar that are supported by the SAS ODBC driver are the same as those described in that book.

### Accessing the SAS Libraries MAPS, SASUSER, and SASHELP

By default, every SAS session (including SAS server sessions) provides access to the SAS libraries MAPS, SASUSER, and SASHELP. However, because these libraries contain sample data sets and other files that are generally not of interest to ODBC users, the SAS ODBC driver does not report the contents of these libraries when it invokes a SAS ODBC server. (From a programming standpoint, when SQLTables, SQLStatistics, or SQLColumns is called, the result set that is returned does not include rows for the SAS libraries MAPS, SASUSER, or SASHELP.) If you want information from these libraries, you can do either of the following tasks:

- Use a LIBNAME statement to define the desired SAS library to the server.
- Use the SAS ODBC Driver Configuration dialog boxes to define the desired library to your ODBC applications.

In both cases you must use a different name (that is, not MAPS, SASUSER, or SASHELP) as your libref or library name.

### Encrypting Passwords

The SAS ODBC driver automatically encrypts passwords in the server definitions that are stored in your system. However, server definitions created before Release 8.1 contain unencrypted passwords. When you install the driver, the program sets up a pwdlog.txt file that lists server definitions that contain unencrypted passwords. You should refer to this file to update your server definitions.
The SAS ODBC driver will display a warning message that the password for a server definition contains an unencrypted value when you

- update a server definition that contains an unencrypted password. The warning message appears when you select OK on the Servers page. You will not receive this message again if passwords are properly updated in the server definition.
- use a server definition with an unencrypted password in an ODBC application. The warning message appears when you connect to the server. You will continue to receive this message until you have updated passwords in the server definition.

For instructions on how to encrypt passwords in an older server definition, see the discussion of updating an unencrypted password in the online Help of the SAS ODBC Driver Configuration dialog box.

---

**Return Codes and Associated Messages**

The SAS ODBC driver uses standard ODBC return codes to notify you of any errors and to provide additional information or warnings. The associated message texts might be generated by the driver itself, by the SAS server, or by your communications access method. See *Microsoft ODBC 2.0 Programmer’s Reference and SDK Guide* and Appendix 1, “Return Codes and Associated Messages,” on page 45 for explanations of these return codes and their associated texts.
Introduction to the Programmer’s Reference

This chapter is intended for applications programmers and others who need information about how the SAS ODBC driver has been implemented. It provides information about the driver’s support for ODBC functions, SQL grammar, and SQL data types.

For complete information about the ODBC standard, see Microsoft ODBC 2.0 Programmer’s Reference and SDK Guide.

Support for and Implementation of ODBC Functions

Microsoft’s ODBC specification defines three levels of support for ODBC functions: CORE, LEVEL 1, and LEVEL 2. The SAS ODBC driver is ODBC 2.0 compliant and supports LEVEL 1 functionality, with the exception of those functions associated with cursors (SQLGetCursorName, SQLSetCursorName) and large data fields (SQLParamData, SQLPutData), which are not supported.
The following tables provide explanations of the functions that are not supported or whose implementation details might be noteworthy to applications developers.

### Table 4.1 CORE Functions

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Purpose</th>
<th>SAS ODBC Driver Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQLBindParameter</td>
<td>Assigns storage for a parameter in an SQL statement</td>
<td>Note that SQL_DATA_AT_EXEC is not supported because SAS does not support large data fields.</td>
</tr>
<tr>
<td>SQLCancel</td>
<td>Cancels an SQL statement</td>
<td>This function is used only in asynchronous mode, which SAS does not support. SAS’ interprocess communication library does not provide any way to interrupt a transaction in process. SQLCancel does not cause an active statement to terminate immediately, and it does not halt an operation that is already in process. This is allowable within the function specification. This call always returns as successful.</td>
</tr>
<tr>
<td>SQLColAttributes</td>
<td>Describes the attributes of a column in the result set</td>
<td>A common reason for applications to call this function is to determine whether a column of data is a dollar amount. With SAS data sets or views, this is inferred from the FORMAT. See “Supported Data Types” on page 37 for more information about SAS FORMATs.</td>
</tr>
<tr>
<td>SQLGetCursorName</td>
<td>Returns the cursor name that is associated with a statement handle</td>
<td>SAS does not support cursors, so this function returns SQL_ERROR with SQLSTATE set to IM001 (“Driver does not support this function”).</td>
</tr>
<tr>
<td>SQLPrepare</td>
<td>Prepares an SQL statement for later execution</td>
<td>Does not check syntax at this point; syntax checking is done later in the SQLExecute call by the server.</td>
</tr>
<tr>
<td>SQLSetCursorName</td>
<td>Specifies a cursor name</td>
<td>SAS does not support cursors, so this function returns SQL_ERROR with SQLSTATE set to IM001 (“Driver does not support this function”).</td>
</tr>
<tr>
<td>SQLTransact</td>
<td>Commits or rolls back a transaction</td>
<td>Always returns SQL_SUCCESS for SQL_COMMIT. Returns an error for SQL_ROLLBACK because SAS does not support transactions.</td>
</tr>
</tbody>
</table>
## LEVEL 1 Functions

### Table 4.2  LEVEL 1 Functions

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Purpose</th>
<th>SAS ODBC Driver Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQLColumns</td>
<td>Returns the list of column names from specified tables</td>
<td>SAS uses a specially formatted query to query the virtual table DICTIONARY.COLUMNS.</td>
</tr>
<tr>
<td>SQLDriverConnect</td>
<td>Connects to a specific driver by connection string or requests that the Driver Manager and the driver display connection dialog boxes for the user</td>
<td>SAS makes use of the same dialog boxes that are used in configuration. If input was adequate, it continues with the connection rather than saving the parameters. However, in many cases input is not required. The connection to the host is made at this time.</td>
</tr>
<tr>
<td>SQLGetData</td>
<td>Returns result data for a single unbound column in the current row. The application must call SQLFetch or SQLExtendedFetch prior to calling SQLGetData</td>
<td>This function enables you to use multiple calls to retrieve data in parts from character variables. Some applications need this functionality to access long character variables. The extension SQL_GD_BLOCK (see SQL_GETDATA_EXTENSIONS under SQLGetInfo) is not supported; this means that you can only call SQLExtendedFetch with a rowset size of 1.</td>
</tr>
<tr>
<td>SQLParamData</td>
<td>Returns the storage value that has been assigned to a parameter for which data will be sent at execution time</td>
<td>This function is used for large data fields, which SAS does not support, so the function returns SQL_ERROR with SQLSTATE set to IM001 (“Driver does not support this function”).</td>
</tr>
<tr>
<td>SQLPutData</td>
<td>Sends part or all of a data value for a parameter</td>
<td>This function is used for large data fields, which SAS does not support, so the function returns SQL_ERROR with SQLSTATE set to IM001 (“Driver does not support this function”).</td>
</tr>
<tr>
<td>SQLSpecialColumns</td>
<td>Retrieves information about the optimal set of columns that uniquely identifies a row in a specified table, or about the columns that are automatically updated when any value in the row is updated by a transaction</td>
<td>The SAS ODBC driver uses a query on the DICTIONARY.INDEXES view to obtain this information.</td>
</tr>
<tr>
<td>Function Name</td>
<td>Purpose</td>
<td>SAS ODBC Driver Implementation</td>
</tr>
<tr>
<td>---------------</td>
<td>---------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>SQLStatistics</td>
<td>Retrieves statistics about a single table and the list of indexes that are associated with the table</td>
<td>The SAS ODBC driver uses a query on the <code>DICTIONARY.INDEXES</code> view to obtain this information.</td>
</tr>
<tr>
<td>SQLTables</td>
<td>Returns the list of table names stored in a specific data source</td>
<td>The SAS ODBC driver uses a query on the <code>DICTIONARY.TABLES</code> and <code>DICTIONARY.MEMBERS</code> views to obtain this information.</td>
</tr>
</tbody>
</table>

**ODBC Cursors and the SQLExtendedFetch Function**

The SAS ODBC driver supports the SQLExtendedFetch function to retrieve multiple rows of result data in a single operation. However, SAS does not support real cursor functionality, so the driver cannot truly support cursors.

The implementation of SQLExtendedFetch uses a forward-only cursor to support the syntax of the function. Rows in the result data cannot be skipped, so the parameter `irow` (which is the number of the row to begin fetching) is ignored. You cannot retrieve an arbitrary subset of the result data. However, because a single operation can fetch multiple rows, there is less overhead compared to when you use multiple calls to the SQLFetch function.

*Note:* In order to support the SQLExtendedFetch function, the SQLSetSmtOption function allows you to set the rowset size and to select row-wise or column-wise binding.

For additional information about the SQLExtendedFetch function, see *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

**Catalog Functions**

You might want to eliminate certain SAS librefs from the results of the catalog functions such as SQLColumns, SQLStatistics, SQLSpecialColumns, and SQLTables. Many client applications (for example, Microsoft Access) use these catalog functions to present a list of available tables. Currently, the results from the catalog functions filter out the MAPS, SASHELP, and SASUSER librefs. The filtering is done primarily so those unwanted tables do not appear in lists based on results from these catalog functions. The filtering can also improve the performance of these functions.

You can replace this filter by adding the LibWhere key to an appropriate data source entry in the system Registry. The LibWhere key includes a logical expression to conditionally specify which results to retrieve from the SAS server. The current filter uses the following logical expression:

```
tbl.libname NE 'MAPS' AND tbl.libname NE 'SASUSER' AND 
tbl.libname NE 'SASADMIN' AND tbl.libname NE 'SASHELP'
```

The `tbl` name is critical if you need to construct your own in these functions.

**CAUTION:**

*When you use the Registry Editor, you can damage the Windows environment* if your changes are invalid or if you accidentally delete information. Do not create a filter unless it is absolutely necessary. Editing the Registry can damage registered programs and make them unusable.

If many librefs are defined on a server, but only specific librefs are needed, you can edit the Registry to specify which librefs to access. For instructions on how to edit the
Registry, see the discussion of catalog functions in the online Help of the SAS ODBC Driver Configuration dialog box.

### ODBC Scalar Functions

The SAS ODBC driver supports the following ODBC scalar functions:

**String Functions:**
- ASCII, CHAR, CONCAT, LCASE, LEFT, LTRIM, REPLACE, RTRIM, SOUNDEX, SPACE, SUBSTRING, UCASE

**Numeric Functions:**
- ABS, ACOS, ASIN, ATAN, CEILING, COS, EXP, FLOOR, LOG, LOG10, MOD, POWER, RAND, ROUND, SIGN, SQRT, TAN

**Timedate Functions:**
- CURDATE, CURTIME, DAYNAME, DAYOFMONTH, DAYOFWEEK, DAYOFYEAR, HOUR, MINUTE, MONTH, MONTHNAME, NOW, QUARTER, SECOND, YEAR

The ODBC `RAND` scalar function is translated to a SAS function. For more information about the usage and parameters of ODBC scalar functions, consult *Microsoft ODBC 2.0 Programmer’s Reference and SDK Guide*.

### Support for SQL Grammar

Microsoft’s ODBC specification defines three levels of support for SQL grammar: MINIMUM, CORE, and EXTENDED. The SAS ODBC driver supports all of MINIMUM and some of the CORE SQL statements and the statement elements. See *Base SAS Procedures Guide* for complete information about supported grammar.

Currently the SAS ODBC driver does not support the following SQL syntax:

```
SELECT FOR UPDATE OF
```

Records are locked during data manipulation operations but cannot be explicitly locked prior to such an operation.

### Supported Data Types

Internally, SAS supports two data types for storing data:

- **CHAR**
  - fixed-length character data, 32767 character maximum

- **NUM**
  - double-precision floating-point number

*Note:* If the data field is longer than 254 characters, this will surface through ODBC with the ODBC data type SQL_VARCHAR. △

By using SAS `format` information, the SAS ODBC driver is able to represent other ODBC data types, both when responding to queries and in CREATE TABLE requests. (A SAS format is a string that describes how data should be printed. SAS associates format information with each column in a table.)

The following sections explain conventions for data type representation that the SAS ODBC driver follows.

For information about user-specified SQL options that can also affect data type representations, see “User-Specified SQL Options” on page 40. For more information about SAS formats, see *SAS Language Reference: Dictionary*. 
ODBC SQL Data Types

The SAS ODBC driver supports the ODBC SQL types listed below. Internally, SAS supports a numeric and character data type using a variety of informats and formats to modify the data representation. The SAS ODBC driver uses the informat and format to determine the corresponding ODBC data type:

<table>
<thead>
<tr>
<th>SQL Data Type</th>
<th>SAS ODBC Driver Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL_CHAR.</td>
<td>Supports text fields up to 200 characters long.</td>
</tr>
<tr>
<td>SQL_VARCHAR.</td>
<td>Interpreted for text columns of length 80 or greater when the Support Varchar option is enabled</td>
</tr>
<tr>
<td></td>
<td>from the SQL Options section of the SAS ODBC Driver Configuration dialog box. The maximum</td>
</tr>
<tr>
<td></td>
<td>length of an SQL_VARCHAR field is 32,767 characters, available when the version of the SAS</td>
</tr>
<tr>
<td></td>
<td>server is Version 7 or later. For SAS Version 6 servers, the SQL_VARCHAR field is limited to</td>
</tr>
<tr>
<td></td>
<td>200 characters.</td>
</tr>
<tr>
<td>SQL_DOUBLE.</td>
<td>1.E-308 to 1.E308</td>
</tr>
<tr>
<td>SQL_FLOAT.</td>
<td>1.E-308 to 1.E308</td>
</tr>
<tr>
<td>SQL_INTEGER.</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td>SQL_DATE.</td>
<td>Valid dates range from 1582 A.D. to 9999 A.D.</td>
</tr>
<tr>
<td>SQL_TIME.</td>
<td>ODBC supports time values within the range of a 24-hour day (00:00:00 to 3:59:59).</td>
</tr>
<tr>
<td>SQL_TIMESTAMP.</td>
<td>Valid dates range from 1582 A.D. to 9999 A.D. with a 24-hour day (00:00:00 to 23:59:59) time</td>
</tr>
<tr>
<td></td>
<td>portion.</td>
</tr>
</tbody>
</table>

Data Types Reported on Queries

When the SQLDescribeCol and SQLColAttributes functions are called against active queries, the SAS ODBC driver reports data types as follows:

- When the SQLDescribeCol function is called, the SAS ODBC driver reports CHAR data types as SQL_CHAR. NUM data types are generally reported as SQL_DOUBLE.

  However, SAS stores dates and times as numbers, and the SAS ODBC driver uses SAS format information to infer the following additional SQL data types from NUM data types:* 

---

* For a complete list of date and time formats that the SAS ODBC driver supports, see the table of formats listed by categories in *SAS Language Reference: Dictionary.*
In each of the previous FORMAT= strings, \( n \) is a number that selects the printable representation by specifying a width for printing. The value of \( n \) is not relevant to the driver.

When the SQLColAttributes function is called, if a NUM column has a format of DOLLAR\( n \), the SAS ODBC driver identifies it as financial data (having a column attribute of SQL_COLUMN_MONEY).

**Retrieving Native SAS Format Information**

You can find the SAS format for a particular column by using the SQLColumns function. The SAS ODBC driver returns the additional column FORMAT. The 13th column in the result set that the SQLColumns function returns contains a string with the format information for a SAS column of data.

**Creating or Comparing Date, Time, and Datetime Values**

When you create or compare date, time and datetime values in SAS data sets from an ODBC application, you must consider the following:

- A SAS time value is the number of seconds since the current day began. That is, 0 is 00:00:00 or 12:00:00 AM and 86399 is 11:59:59 PM.

  *Note:* ODBC does not support negative time values or values greater than one day’s worth of seconds. The SAS ODBC driver returns an error for time values that are less than 0 or greater than 86399 (the last second of the day).

- A SAS date value is the number of days since January 1, 1960. That is, 0 is 01jan1960 and -1 is 31dec1959.

- A SAS datetime value is the number of seconds since midnight on January 1, 1960. That is, 0 is 01jan1960:00:00:00 and -1 is 31dec1959:11:59:59.

Both ODBC and SAS date, time and datetime literals are supported by the SAS ODBC driver.

**CAUTION:**

You can only compare equivalent literals against SAS date, time or datetime values since they each have a different unit of measure.

For example, you cannot compare a SAS data set value that has been defined with a datetime format against a date literal using

```
select * where hiredate = {d'1995-01-02'}
```

or

```
select * where hiredate = '02jan1995'd
```

Instead, use a datetime literal such as

```
select * where hiredate = {ts'1995-01-02 00:00:00'}
```
or

```sql
select * where hiredate = '02jan1995:00:00:00'dt
```

## Interpreting Data Types in CREATE TABLE Requests

In CREATE TABLE requests, the SAS ODBC driver interprets certain column-type specifications by creating NUM variables and associating SAS formats with them, as shown in the following table:

### Table 4.3 Correspondence of CREATE TABLE Data Types and SAS Data Types

<table>
<thead>
<tr>
<th>CREATE TABLE ODBC Data Type</th>
<th>SAS Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>char(w)</td>
<td>SQL_CHAR</td>
</tr>
<tr>
<td>num(w, d)</td>
<td>SQL_DOUBLE</td>
</tr>
<tr>
<td>integer</td>
<td>SQL_INTEGER</td>
</tr>
<tr>
<td>date9x</td>
<td>SQL_DATE</td>
</tr>
<tr>
<td>datetime19x</td>
<td>SQL_TIMESTAMP</td>
</tr>
<tr>
<td>time8x</td>
<td>SQL_TIME</td>
</tr>
</tbody>
</table>

The data type names listed in the first column of the table are the values that are returned by SQLColAttributes (with the parameter SQL_COLUMN_TYPE_NAME) and by SQLGetTypeInfo. For all CREATE TABLE statements, the SAS ODBC driver translates these data type names into the respective SAS data types shown under the SAS Data Type heading. Do not try to use the ODBC data types directly in SAS.

In a CREATE TABLE statement, any FORMAT= specification is passed on to SAS unmodified, so a column within a table (or data set) can be created according to any exact specification that is required for its use within SAS. For example, in the following CREATE TABLE statement, variable B’s data type and format are passed directly to SAS.

```sql
CREATE TABLE
SASUSER.TABLE1
(A INTEGER,
 B NUM FORMAT=9.5,
 C CHAR(40) );
```

### User-Specified SQL Options

This section describes two SQL options that affect how other default conversions of data types or data values can be made: Infer INTEGER from FORMATS and Support VARCHAR. The SQL option Disable _0 override parsing will prevent a SAS error by ensuring that the SAS ODBC driver keeps the _0 string at the end of a table name. Fuzz Numbers at N Places is important in comparison operations. You can specify these SQL options on the General page of the SAS ODBC Driver Configuration dialog box. (See “Setting Up Your Data Source” on page 14.)
**Infer INTEGER from FORMAT Option**

Even when no FORMAT string is specified for SAS data, SAS assigns a default width and number of decimal places to the data. If the SQL Option **Infer INTEGER from FORMAT** is selected, then the SAS ODBC driver reports SAS columns of NUM\((n,0)\) data types as SQL_INTEGER, where \(n\) is less than 12. This can be important, because some PC products do not use indexes on floating-point columns. If those columns actually contain only integer values, then using this option enables these products to honor the index and allow updates. See “Updating Attached Tables” on page 30 for more information.

**Support VARCHAR Option**

The SQL option **Support VARCHAR** causes the SAS ODBC driver to report the data type CHAR\((n)\) as SQL_VARCHAR, where \(n\) is greater than 80. Because SAS is fixed width, CHAR fields are often specified at the maximum. For example, for a list of messages the text width might be specified as 200 characters, even though the average width is much less. Reporting it as SQL_VARCHAR enables some PC products to use less memory.

**Disable _0 override parsing Option**

Sometimes the _0 string occurs at the end of names in SQL queries constructed by certain applications. For example, the string _0 is added to table aliases in SQL queries that MS Query constructs so that the table name `mytable` becomes `mytable_0`. This table name is nine characters in length, which SAS Version 6 cannot handle because table names are limited to eight characters.

To prevent a SAS error, the SAS ODBC driver removes the _0 string found in the eighth or greater position of a table name. However, if SAS variable names contain the _0 string at the eighth or greater position, a query might not execute properly. A variable name such as `aaaaaa_0` will not cause a problem but variable names such as `aaaaaaa_0` and `aaaaaa_0` will cause SAS errors. The SQL option **Disable _0 override parsing** prevents the SAS ODBC driver from removing the _0 string.

*Note:* You can use this option if the data on your SAS server has been created with Version 7 or later of the ODBC server, and you are running Version 7 or later of the SAS server. This is because Version 7 can handle table names and variable names up to 32 characters in length. If your SAS server is Version 6, or if you use Version 7 or later of the ODBC server to access data in Version 6 format, parsing errors might occur depending on the length of your table names, or whether the _0 string exists in a variable name.

**Fuzz Numbers at N Places Option**

This option addresses a problem that arises from the conversion of floating-point numbers. Floating-point numbers are stored in different binary representations on different computer hardware. Even when data is transferred between different applications on the same type of hardware, the precision of floating-point numbers might be affected slightly due to conversion between ASCII and binary representations.

This effect is usually so slight that it is insignificant when a number is used in calculations. For example, the numbers 65.8 and 65.799999999999 are practically identical for mathematical purposes, and the difference between them might be the
result of conversion between representations rather than any purposeful change in value.

However, such a slight difference in value can keep a number from comparing correctly. For example, many ODBC applications include a WHERE clause that lists every column in a record at its current value whenever the application performs an UPDATE. This is done to ensure that the record has not been changed since the last time it was read. Sometimes a comparison might fail because of the aforementioned problem with floating-point conversion.

To solve this problem, SAS “fuzzes” numbers (standardizes the degree of precision to use, overriding the hardware-specific representations). Instead of using exact comparisons, SAS checks to make sure that the numbers are acceptably close.

By default, the degree of precision is 12 decimal places. Given a number $N$, if $N_1$ were to be checked for equality with $N$, then the SAS ODBC driver would use the SQL BETWEEN function to determine whether $N_1 > (N - (\text{ABS}(N \times 10^{-12})))$ AND $N_1 < (N + (\text{ABS}(N \times 10^{-12})))$.

If $N=0$, the driver checks for BETWEEN $-(10^{-12})$ AND $(10^{-12})$.

---

**Security Notes on Windows NT, Windows 2000, and Windows XP**

You can set up a secure SAS/SHARE server on Windows NT, Windows 2000, and Windows XP. To specify a secure SAS/SHARE server, you submit code similar to the following code on your Windows system:

```plaintext
/* Start a remote secured SAS/SHARE server that requires user login */
/* name and password specified by the client. */
/* ADOMAIN is the NT domain that authenticates username and password */
%let tcpsec=_secure_; /* Require user loginid/password */
options authserver=adomain;
proc server id=shr1 authenticate=req;
run;
```

However, for this program to work you must assign Act as part of the operating system rights to the user account that is running the SAS/SHARE server. You must also assign Log as a batch job rights to the account that wants to connect to the SAS/SHARE server.

For example, suppose your Windows system is named TRISTAN, and the user account adomain\stephmc wants to connect to the SAS/SHARE server. Also, suppose the user logged in to TRISTAN (and running the SAS/SHARE server) is the user account adomain\joshua. On the system named TRISTAN, you must assign the user rights. For Windows NT or Windows 2000, use the utility User Manager or User Manager for Domains. For Windows XP, use the group policy.

For Windows NT or Windows 2000 complete the following steps:

1. Start --> Programs --> Administrative Tools --> User Manager.
2. Select User Rights... from the Policies menu.
4. Grant Act as part of the operating system rights to adomain\joshua.
5. Grant Logon as a batch job rights to adomain\stephmc.
For Windows XP, complete the following steps:

1. Select Start --> Run, type gpedit.msc, and then click [OK].
3. Right-click Act as part of the operating system and then select Properties. Select Add User or Group, type adomain\joshua, and then click [OK].
4. Right-click Log on as a batch job and then select Properties. Select Add User or Group, type adomain\stephmc, and then click [OK].

After the system TRISTAN is shut down and restarted (rebooting is usually required), you submit the above SAS/SHARE code on TRISTAN.

To log on to the remote Windows system, the user account adomain\stephmc specifies the user name adomain\stephmc in the SAS ODBC Driver Configuration dialog box and the password in the SAS/SHARE Options dialog box. The user will also specify an address such as tristan.mynet.com. This will enable the user account adomain\stephmc to connect to the secure SAS/SHARE server.

This information is also documented in SAS/SHARE User's Guide.

### SAS ODBC Driver Error Codes

See Microsoft ODBC 2.0 Programmer’s Reference and SDK Guide for information about the SQLSTATE values (return codes) and associated texts that can be returned for the SQLError function.

For explanations of messages that might be returned by your communications software, see Appendix 1, “Return Codes and Associated Messages,” on page 45.
Return Codes and Associated Messages

SAS ODBC Driver Return Codes

See the *Microsoft ODBC 2.0 Programmer’s Reference and SDK Guide* for information about the SQLSTATE values (return codes) and the associated text that can be returned for the SQLException function. The messages might be generated by the driver itself, by the SAS server, or by your communications software. The ODBC Driver Manager passes these codes and messages on to client applications.

S1000 Communications Access Method Errors

The S1000 (SAS API Error) return code is often accompanied by error messages that are returned by your communications software. The following tables list some of these message texts and provide explanations for them.

In addition to these error messages and return codes, some additional information can sometimes be found in a “trace” file that is created in the working directory of the ODBC client application that fails a connection to a SAS server. This trace file has the name `WQExxxxx.TRCE`, where `xxxxx` is the process ID of the ODBC client application at the time of failure.

<table>
<thead>
<tr>
<th>Message Text</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory failure</td>
<td>Not enough memory is available.</td>
</tr>
<tr>
<td>Network failure</td>
<td>An unspecified network failure occurred.</td>
</tr>
<tr>
<td>No server found</td>
<td>The remote server was not found.</td>
</tr>
<tr>
<td>Remote closed connection</td>
<td>The SAS server disconnected.</td>
</tr>
<tr>
<td>Remote refused connection</td>
<td>The remote system disallowed a connection. Check the remote services file.</td>
</tr>
<tr>
<td>Start SAS failure—please check your SAS server parameters</td>
<td>The ShellExecute statement failed when starting SAS. Check to see whether the SAS paths are specified properly.</td>
</tr>
</tbody>
</table>
A TCP/IP Winsock return code (rc) was returned. Even though it is impossible to document all the possible reasons you might get one of these return codes, they are listed in Table A1.2 on page 46 to provide some indication of where the Winsock transport is having a problem.

A SAS server did not register itself as a server within the specified time period.

TCP/IP could not find the remote host name.

TCP/IP could not find the server name in the services file.

User ID and password verification failed on the remote machine.

You must select the SAS/SHARE button on the SAS ODBC Servers page (Display 2.4 on page 16) in order to connect to a remote machine.

<table>
<thead>
<tr>
<th>Table A1.2 TCP/IP Winsock Return Codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Code</td>
</tr>
<tr>
<td>-------------</td>
</tr>
<tr>
<td>10004</td>
</tr>
<tr>
<td>10013</td>
</tr>
<tr>
<td>10014</td>
</tr>
<tr>
<td>10022</td>
</tr>
<tr>
<td>10024</td>
</tr>
<tr>
<td>10035</td>
</tr>
<tr>
<td>10036</td>
</tr>
<tr>
<td>10037</td>
</tr>
<tr>
<td>10038</td>
</tr>
<tr>
<td>10039</td>
</tr>
<tr>
<td>10040</td>
</tr>
<tr>
<td>10041</td>
</tr>
<tr>
<td>10042</td>
</tr>
<tr>
<td>10043</td>
</tr>
<tr>
<td>Return Code</td>
</tr>
<tr>
<td>------------</td>
</tr>
<tr>
<td>10044</td>
</tr>
<tr>
<td>10045</td>
</tr>
<tr>
<td>10046</td>
</tr>
<tr>
<td>10047</td>
</tr>
<tr>
<td>10048</td>
</tr>
<tr>
<td>10049</td>
</tr>
<tr>
<td>10050</td>
</tr>
<tr>
<td>10051</td>
</tr>
<tr>
<td>10052</td>
</tr>
<tr>
<td>10053</td>
</tr>
<tr>
<td>10054</td>
</tr>
<tr>
<td>10055</td>
</tr>
<tr>
<td>10056</td>
</tr>
<tr>
<td>10057</td>
</tr>
<tr>
<td>10058</td>
</tr>
<tr>
<td>10059</td>
</tr>
<tr>
<td>10060</td>
</tr>
<tr>
<td>10061</td>
</tr>
<tr>
<td>10062</td>
</tr>
<tr>
<td>10063</td>
</tr>
<tr>
<td>10064</td>
</tr>
<tr>
<td>10065</td>
</tr>
<tr>
<td>10066</td>
</tr>
<tr>
<td>10067</td>
</tr>
<tr>
<td>10068</td>
</tr>
<tr>
<td>10069</td>
</tr>
<tr>
<td>10070</td>
</tr>
<tr>
<td>10071</td>
</tr>
<tr>
<td>10091</td>
</tr>
<tr>
<td>10092</td>
</tr>
<tr>
<td>Return Code</td>
</tr>
<tr>
<td>-------------</td>
</tr>
<tr>
<td>10093</td>
</tr>
<tr>
<td>11001</td>
</tr>
<tr>
<td>11002</td>
</tr>
<tr>
<td>11003</td>
</tr>
<tr>
<td>11004</td>
</tr>
</tbody>
</table>
Recommended Reading

Here is the recommended reading list for this title:

- Base SAS Procedures Guide
- SAS/ACCESS for PC Files: Reference
- SAS/ACCESS for Relational Databases: Reference
- SAS Companion for Windows
- SAS Language Reference: Concepts
- SAS/SHARE User's Guide

For a complete list of SAS publications, see the current SAS Publishing Catalog. To order the most current publications or to receive a free copy of the catalog, contact a SAS representative at

SAS Publishing Sales
SAS Campus Drive
Cary, NC 27513
Telephone: (800) 727-3228*
Fax: (919) 677-8166
E-mail: sasbook@sas.com
Web address: support.sas.com/pubs
* For other SAS Institute business, call (919) 677-8000.

Customers outside the United States should contact their local SAS office.
access descriptor
a SAS/ACCESS file that describes data that is managed by a database management system. After creating an access descriptor, you can use it as the basis for creating one or more view descriptors.

application programming interface (API)
a set of software functions that facilitate communication between applications and other kinds of programs, services, or devices.

access method
the communications protocol that the SAS ODBC driver uses to exchange data with a SAS server. The driver currently supports the use of TCP/IP and Network DDE for remote data exchange, and DDE for local exchange.

API
See application programming interface (API).

client
a workstation or application that requests services, data, or other resources from a server.

DBMS
See database management system (DBMS).

database management system (DBMS)
a software application that enables you to create and manipulate data that is stored in the form of databases.

data source name (DSN)
a name that is associated with a data source definition. The data source definition specifies how to locate and access a data source, including any authentication (such as a user name and password) that a user must supply in order to access the data.

dialog window
a window that prompts a user for additional information in order to perform a specified action.

driver
See ODBC driver.
DSN
See data source name (DSN).

data source name (DSN)

engine
a component of SAS software that reads from or writes to a file. Each engine enables SAS to access files that are in a particular format. There are several types of engines.

file dsn
a data source name that is stored completely within a file (unlike a user DSN or a system DSN, which are stored in the Windows registry).

libref
a name that is temporarily associated with a SAS data library. For example, in the name SASUSER.ACCOUNTS, the name SASUSER is the libref. You assign a libref with a LIBNAME statement or with an operating system command.

ODBC driver
a loadable library module that provides a standardized interface to disparate databases or data sources.

SPD server
a SAS Scalable Performance Data server. An SPD server restructures data in order to enable multiple threads, running in parallel, to read and write massive amounts of data efficiently.

system DSN
a data source name that can be accessed by any user of the system on which the data source is stored. System DSNs are stored in the Windows registry.

user DSN
a data source name that can be accessed by any user of the system on which the data source is stored. System DSNs are stored in the Windows registry.
Numbers

_Zero override parsing 15, 41

A

attached tables
updating 30

C

catalog functions 36
CHAR data type 37
character field length 15
connection options 19
CORE functions 25
Create New Data Source dialog box 12
CREATE TABLE requests
data types in cursors 36

D
data files 9
data libraries 6
defining 20
defining at server startup time 22
MAPS 31
read-only access 22
returning data set labels 15
SASHelp 31
SASUSER 31
data library definitions 20
defining at server startup time 22
deleting engines 21
modifying 23
naming 21
data sets 30
one-level names 30
data source definitions 11
accessing dialog boxes 11
defining libraries 11
defining servers 11
deleting 24
modifying 23
saving 23
server specification 23
setting up data sources 14
data sources 3
accessing 25
defining 25
naming 24
setting up 14
DATA step views 5
data types 37
38
date values 39
datetime values 39
event in CREATE TABLE requests 10
reported on SQL queries 38
time values 39
data views 5
date values 39
datetime values 39
DDE servers 7
converting to TCP servers 7
dialog boxes 11
accessing 11
Disable _0 override parsing option 15, 41

E
crypting passwords 31
engines 21
error codes 21

F
failed SQL updates 15
formats 39

G

I
Infer INTEGER from FORMAT option 15, 41

L
LEVEL 1 functions 35
librefs 3 21

O

ODBC 2
terminology 8
ODBC Data Source Administrator dialog box 12
ODBC driver 12
accessing data sources 35
communications prerequisites 26
error codes 21
functionality 3
local data sources 26
remote data sources 26
return codes 32 35
SAS software and software requirements 8
ODBC functions 33
catalog functions 36
CORE functions 34
LEVEL 1 35
scalar functions 37
SQLExtendedFetch 36
one-level data set names 30
Open Database Connectivity (ODBC) 2
See ODBC driver
owner 8
Index

P
passwords 17, 19, 20, 28
encrypting 19
precision, numeric 16
Preserve trailing blanks option 15
PROC SQL views 5

Q
qualifiers 8
queries
  data types reported 16
  logging 18
  removing _0 string 15

R
read-only access 22
remote data 5
  accessing 27
  servers and 7
return codes 12, 45
  S1000 communications access methods 45
  TCP/IP Winsock 46
Return SQLTables REMARKS option 15
rows
  number to return 15

S
S1000 communications access method errors 45
SAS/ACCESS software 4
SAS/ACCESS views 5
SAS ODBC driver
  See ODBC driver
SAS ODBC Driver Configuration dialog box 13, 14
  Libraries tab 20
  Servers tab 16
SAS/SHARE data sources 27
SAS/SHARE Options dialog box 18
SAS/SHARE servers 5
SAS software
  ODBC driver and 5
  startup parameters 18
terminology 8
SASHELP data library 31
%sasodbc macro 15
SASODBC.SAS file 13
SASUSER data library 31
Scalable Performance Data (SPD) server 7
scalar functions 37
security
  Windows 2000 12
  Windows NT 42
server definitions
  connection options 19
  deleting 20
  local data 16
  logging SQL queries 18
  modifying passwords 20
  path for executable file 17
  server address 19
  startup parameters 18
  timeout limit 18
  user name 19
  working directory 15
  servers
    address of 19
    changing 23
    converting DDE servers to TCP servers 7
    defining 16
    local data and 7
    remote data and 7
    Scalable Performance Data server 7
    starting 18
    terminating 26
    software requirements 3
    SPD server 7
    SQL data types 38
    SQL grammar 37
    SQL options 15
    Disable _0 override parsing 15
    Fuzz Numbers at N Places 16
    Infer INTEGER from FORMAT 15
    Preserve trailing blanks 15
    Return SQLTables REMARKS 15
    Support VARCHAR REMARKS 15
    UNDO_POLICY=REQUIRED 15
    user-specified 40
SQL statements
  accessing data sources with 31
  SQLExtendedFetch function 36
  SQL_INTEGER 15
  SQLTables function 15
  Support VARCHAR option 41

T
tables 5
  updating attached tables 50
TCP/IP servers 7
TCP/IP Services file 29
editing 30
TCP/IP Winsock
  return codes 46
TCP servers
  converting DDE servers to 7
terminology 8
time values 39
timeout limit 18
trailing blanks 15

U
UNDO_POLICY=REQUIRED option 15
userid 19, 20, 28
Userid/Password Override option 28

V
VARCHAR option 15, 41
views 5

W
Windows 2000
  security 42
Windows NT
  security 42
  working directory 15
Your Turn

If you have comments or suggestions about *SAS® ODBC Driver 9.1: User's Guide and Programmer’s Reference*, please send them to us on a photocopy of this page, or send us electronic mail.

For comments about this book, please return the photocopy to

SAS Publishing  
SAS Campus Drive  
Cary, NC 27513  
E-mail: yourturn@sas.com

For suggestions about the software, please return the photocopy to

SAS Institute Inc.  
Technical Support Division  
SAS Campus Drive  
Cary, NC 27513  
E-mail: suggest@sas.com