CHAPTER

3

Using SAS Files

Introduction to SAS Files **56**
What Is a SAS File? **56**
Types of SAS Files **57**
Using Short or Long File Extensions in SAS Libraries **58**
SAS Data Sets (Member Type: Data or View) **58**
SAS Catalogs (Member Type: Catalog) **60**
SAS Stored Program Files (Member Type: Program) **60**
Access Descriptor Files (Member Type: Access) **60**
Multiple Engine Architecture **60**
Library Engines **61**
Native Library Engines **61**
Interface Library Engines **62**
Rules for Determining the Engine **62**
Using Data Libraries **62**
Accessing the New Library Dialog Box Using the Graphical User Interface **63**
Assigning SAS Libraries Using the LIBNAME Statement or Function **63**
Assigning a Libref to a Single Folder **64**
Assigning a Libref to the Working Folder **64**
Assigning a Libref to Multiple Folders **64**
Assigning Engines **64**
Using the LIBNAME Statement in SAS Autoexec Files **65**
Assigning Multiple Librefs and Engines to a Folder **65**
Assigning SAS Libraries Using Environment Variables **65**
SET System Option **66**
OS/2 SET Command **66**
Listing Libref Assignments **67**
Clearing Librefs **67**
Understanding How Concatenated SAS Data Libraries Are Accessed **68**
Input and Update Access **68**
Output Access **68**
Accessing Data Sets with the Same Name **68**
Using the SASUSER Data Library **69**
Using the WORK Data Library **69**
Using an Environment Variable **70**
Using the USER Libref **70**
Accessing SAS Files from Multiple SAS Sessions **70**
Using SAS Files from Other Releases with Version 8 for OS/2 **71**
Using Version 6 SAS Data Sets in Version 8 **72**
Using Release 6.08 through Release 6.12 Data Sets **72**
Using Release 6.03 and Release 6.04 SAS Data Sets **73**
Converting Version 6 SAS Data Sets **73**

Creating Version 6 Data Sets **74**
Converting Version 6 SAS Catalogs in Version 8 **74**
Converting Release 6.06 SAS Catalogs **74**
Using Version 8 SAS Files with Previous Releases **75**
Using Version 5 and Other Remote Host SAS Files in Version 8 **75**
Reading BMDP, OSIRIS, and SPSS Files **76**
BMDP Engine **76**
BMDP Engine Examples **76**
OSIRIS Engine **77**
OSIRIS Engine Example **77**
SPSS Engine **78**
SPSS Engine Example **78**
Transferring SAS Files between Operating Environments **78**
Accessing Database Files with SAS/ACCESS Software **78**

Introduction to SAS Files

This section briefly reviews basic concepts about SAS files, specifically those that are stored under OS/2. For additional information about SAS files, see *SAS Language Reference: Concepts*.

What Is a SAS File?

The SAS System creates and uses a variety of specially structured files called *SAS files*. Although OS/2 manages SAS files for the SAS System by storing them, the operating environment cannot process files. For example, you can list SAS files within the OS/2 operating environment, but you cannot use an OS/2 text editor to edit SAS files.

SAS files usually reside in SAS data libraries. Under OS/2, a SAS library is simply a *named collection of SAS files within one or more OS/2 folders that the SAS System can access*. Each SAS data library has an access engine that is associated with it the first time that a file in the library is accessed. The engine name specifies the access method that the SAS System uses to process the files in the data library. SAS data libraries are described in detail in *SAS Language Reference: Concepts*.

Through SAS Multiple Engine Architecture, various engines enable the SAS System to access different formats or versions of SAS files and other vendors' files. Multiple Engine Architecture, combined with conversion utilities, provides access to Version 8 files and SAS files that were created with previous releases of the SAS System (beginning with Version 5), whether they were created under either OS/2 or other operating environments. Multiple Engine Architecture also provides access to files that were created by other vendors' products, including database files. For details, see "Multiple Engine Architecture" on page 60.

The following sections highlight information you need in order to create and use SAS files with the various engines under OS/2.

Note: SAS files are different from external files. While external files can be processed by SAS statements and commands, they are not managed by the SAS System. △

Types of SAS Files

SAS files are stored in SAS data libraries and are referred to as *members* of a library. Each member has a *member type*. The SAS System distinguishes between SAS files and external OS/2 files in a folder by using unique file extensions. The SAS System assigns certain file extensions to a general set of SAS member types. Table 3.1 on page 57 lists the OS/2 file extensions and their corresponding SAS member types for the V6, V7, and V8 engines. For more information about engines, see "Multiple Engine Architecture" on page 60.

Table 3.1 OS/2 File Extensions and Their Corresponding SAS Member Types

V6 File Extension	V7 and V8 File Extension (Short)	V7 and V8 File Extension (Long)	SAS Member Type	Description
.sas	.sas	.sas	none	SAS program
.ss2	.ss7	.sas7bpgm	Program	stored program (DATA step)
.lst	.lst	.lst	none	output file
.log	.log	.log	none	log file
none	.st7	.sas7baud	Audit	audit file
.sd2	.sd7	.sas7bdat	Data	data set
.sv2	.sv7	.sas7bvew	View	data set view
.si2	.si7	.sas7bndx	Index	data set index. Indexes are stored as separate files but are treated by the SAS System as integral parts of the SAS data file.
.sc2	.sc7	.sas7bcat	Catalog	SAS catalog
.sa2	.sa7	.sas7bacs	Access	access descriptor file
.sf2	.sf7	.sas7bfdb	FDB	consolidation database file
.sm2	.sm7	.sas7bmdb	MDDB	multidimensional database file
none	.s7m	.sas7bdmd	DMDB	data mining database file
none	.sr7	.sas7bitm	Itemstor	item store file
.su2	.su7	.sas7butl	Utility	utility file
.sp2	.sp7	.sas7bput	Utility	permanent utility
.stx	.stx	none	none	transport file
none	.sb7	.sas7bbak	none	backup file

CAUTION:

Do not change the file extension of a SAS file; doing so can cause unpredictable results. The file extensions assigned by the SAS System to SAS files are an integral part of how the SAS System accesses these files. Also, you should not change the filename of

a SAS file using operating environment commands. If you want to change the name of a SAS file, use the DATASETS procedure or select the file in the SAS Explorer window and select

Edit ► Rename...

△

Note: In your WORK and SASUSER data libraries, you may see files with file extensions other than those listed in the table. Most of these are temporary utility files that you do not need to access directly; be sure not to delete any of them during your SAS session.

If for some reason your SAS session ends abnormally, you might need to delete these files by using operating environment commands in order to regain disk space. △

Using Short or Long File Extensions in SAS Libraries

Version 8 libraries can be either short file extension libraries or long file extension libraries. Although both the OS/2 HPFS file system and Version 8 SAS software support long filenames, short file extension libraries are necessary to access OS/2 FAT file systems and libraries that reside on servers that support only short file extensions.

You can specify in the LIBNAME statement whether the library supports short or long file extensions. For example, if your SAS library is on a server mapped as the S drive and the server file system supports only short file extensions, your libname statement would look similar to this:

```
libname mylib 's:\sasv8' shortfileext;
```

For information on specifying short or long file extensions using the LIBNAME statement, see "LIBNAME" on page 301.

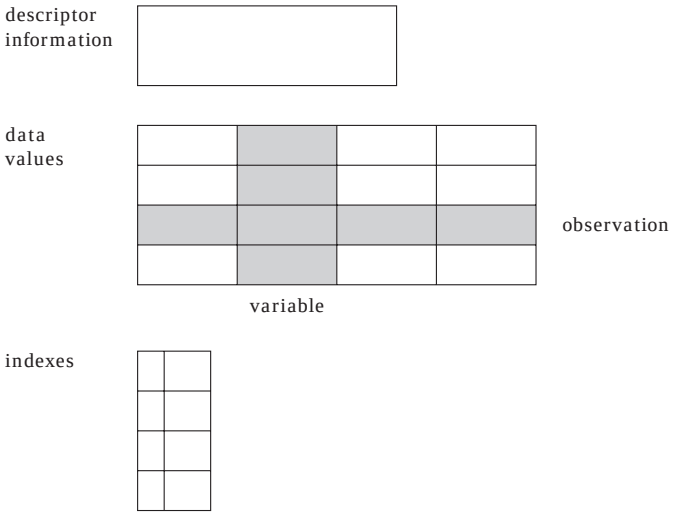
If SAS is not able to create a file with a long file extension the first time it writes to a library, then the library supports only files with short file extensions. If you specify a file with a long file extension for a library that supports only short file extensions, an error message informs you that the member name is too long for the system.

SAS Data Sets (Member Type: Data or View)

SAS data set is an umbrella term for SAS data files and SAS data views, which are both discussed here. This section provides a brief overview of the concept of SAS data sets. For complete details, see the section on data sets in *SAS Language Reference: Concepts*.

Logically, a SAS data set consists of two types of information: descriptor information and data values. The descriptor information includes such things as data set name, data set type, data set label, and number of variables, as well as the names and labels of the variables in the data set, their types (character or numeric), their length, their position within a record, and their formats. The data values contain values for the variables. A SAS data set can be visualized as a table consisting of rows of observations and columns of variable values. Figure 3.1 on page 59 illustrates the SAS data set model:

Figure 3.1 SAS Data Set Model



SAS data files (member type: Data)

The *SAS data file* is probably the most frequently used type of SAS file. SAS data files have a SAS member type of Data and are created in the DATA step and by certain SAS procedures such as the RANK procedure in base SAS software.

The SAS System defines two types of SAS data files, native and interface. Native data files store data values and descriptor information, as described earlier, in files that are formatted by the SAS System. These are the SAS data sets that you may be familiar with from previous versions of the SAS System under other operating environments. In the SAS System under OS/2, native SAS data files can be indexed. The *index* is an auxiliary file that is created in addition to the SAS data file. The index provides fast access to observations within a SAS data file through a variable or key. Indexes are stored as separate files but are treated by the SAS System as integral parts of the SAS data file.

The second type of data file is the interface SAS data file. These files store data that has been formatted by other software. Examples of interface SAS data files are BMDP, OSIRIS, and SPSS files, which the SAS System can access as read-only files. See “Reading BMDP, OSIRIS, and SPSS Files” on page 76 for more information.

In most cases, the maximum file size for a SAS data set is 2 gigabytes (GB). For information about the size limitation of a data set under OS/2, see “Length and Precision of Variables under OS/2” on page 389.

SAS data views (member type: View)

SAS data views have a member type of View. They describe data values and tell the SAS System where to find the values, but they do not contain the actual data values themselves.

Views may be of two kinds, native or interface. A native SAS data view is created with the SQL procedure or with the DATA step and describes a subset or combination of the data in one or more SAS data files or SAS data views. For information on SQL views, see the *SAS Procedures Guide*. For information on DATA step views, see *SAS Language Reference: Concepts*.

Interface SAS data views contain descriptor information for data that has been formatted by other software products, for example, a database management system. Such a view is created with the ACCESS procedure in SAS/ACCESS

software. For more information, see *SAS/ACCESS Software for PC File Formats: Reference* and other SAS/ACCESS documentation.

SAS Catalogs (Member Type: Catalog)

A *SAS catalog* is a special type of SAS file that can contain multiple entries. You can keep different types of entries in the same SAS catalog. For example, catalogs can contain windowing applications, key definitions, toolbox definitions, SAS/GRAPH graphs, SAS/IML matrices, and so on.

If you want to use Version 8 to access catalogs that were created with earlier releases of the SAS System for OS/2, you must first convert the catalogs from the earlier releases to Version 8 format before you can use them in a Version 8 SAS program.

For more information about how to convert SAS catalogs, see *Moving and Accessing SAS Files across Operating Environments*.

SAS Stored Program Files (Member Type: Program)

A *stored program file* is a compiled DATA step that is generated by the Stored Program Facility. For more information about this type of SAS file, see *SAS Language Reference: Concepts*.

Access Descriptor Files (Member Type: Access)

Descriptor files that have been created by the ACCESS procedure in SAS/ACCESS software have a member type of ACCESS and are used when you are creating interface SAS data views. Descriptor files describe the data that have been formatted by other software products supported by the SAS System under OS/2. For more information, see *SAS/ACCESS Software for PC File Formats: Reference* and other available SAS/ACCESS documentation.

Multiple Engine Architecture

All permanent and temporary SAS files are stored in SAS data libraries. To use a SAS data library in your SAS session, you must assign a *libref* (library reference) and an engine to the data library. The libref is the name that you use to refer to the data library during a SAS session or job. You can define the libref with an environment variable or with the LIBNAME statement or function. For a complete explanation of librefs, see *SAS Language Reference: Concepts*. The SAS Explorer window provides an easy way to manage all of your SAS files, including librefs. For information about working with SAS files in the SAS Explorer window, see *Getting Started with the SAS System*. For information about using librefs in the OS/2 environment, see “Using Data Libraries” on page 62.

A *SAS data library* is a collection of SAS files within an OS/2 folder or a concatenation of folders. Although the folder can contain files that are not managed by the SAS System, only SAS files are considered to be part of the SAS data library. Any OS/2 folder can be treated as a SAS data library.

Access methods called engines provide access to many formats of data, giving the SAS System a *Multiple Engine Architecture*. Engines may be used with SAS data sets only.

The engine identifies the set of routines that the SAS System uses to access the files in the data library. With this architecture, data can reside in different types of files, including SAS data files and data formatted by other software products, such as database management systems. By using the appropriate engine for the file type, the

SAS System can write to or read from the file. For some types of files, you need to tell the SAS System what engine to use. For others, SAS automatically chooses the appropriate engine. For more details about engines and Multiple Engine Architecture, see *SAS Language Reference: Concepts*.

Engines are of two basic types, library and view. *Library engines* control access at the SAS data library level and can be specified in the LIBNAME statement or function. *View engines* enable the SAS System to read SAS data views described by the DATA step, SQL procedure, or SAS/ACCESS software. The use of SAS view engines is automatic because the name of the view engine is stored as part of the descriptor portion of the SAS data set. You cannot specify a view engine in the LIBNAME statement or function.

Library Engines

SAS has two types of library engines: native and interface. These engines support the SAS data library model. Library engines perform several important functions, including determining fundamental processing characteristics. For a more detailed description of library engines, see *SAS Language Reference: Concepts*. For examples of using library engines, see "Using Data Libraries" on page 62.

Native Library Engines

Native library engines are those engines that access forms of a SAS file that are created and maintained by the SAS System. Native library engines include the default engine, the compatibility engine, and the transport engine. Table 3.2 on page 61 lists the acceptable names (and nicknames) for these engines.

Table 3.2 Native Library Engines

Category	Engine Names	Description
default	V8, BASE	accesses Version 8 data files
Version 7 compatibility	V7	accesses Version 7 data files
Version 6 compatibility	V6	accesses any data file created by Release 6.06 through Release 6.12
transport	XPORT	accesses transport files

When you are using the default engine, choose the name, either V8 or BASE, to use in your SAS jobs with an eye to the future. If your application is intended for Version 8 of the SAS System only, and you do not want to convert it to later releases, use the name V8. If, however, you plan to convert your application to new releases of the SAS System, use the name BASE because that refers to the latest default engine. Using the name BASE makes your programs easy to convert. The nickname BASE does not refer to base SAS software; rather, it refers to the base, or primary, engine. The BASE engine can be used with more SAS products than base SAS software.

Note: This book uses the term *default engine* to refer to the V8 engine. The V8 engine is the default engine for accessing SAS files under Version 8 of the SAS System unless the default engine is changed with the ENGINE system option. To see the value of the ENGINE system option, select

Tools ► Options ► System

to open the SAS System Options window. Then select

Files ▶ SAS Files

The ENGINE system option displays the default engine for SAS data libraries. △

Interface Library Engines

Interface library engines support access to other vendors' files. These engines allow read-only access to BMDP, OSIRIS, and SPSS files. You must specify as part of the LIBNAME statement or function the name of the interface library engine that you want. Table 3.3 on page 62 lists the interface engine names:

Table 3.3 Interface Library Engines

Name	Description
BMDP	allows read-only access to BMDP files
OSIRIS	allows read-only access to OSIRIS files
SPSS	allows read-only access to SPSS files

For more information about these engines, see "Reading BMDP, OSIRIS, and SPSS Files" on page 76.

Rules for Determining the Engine

If you do not specify an engine name in a LIBNAME statement or function, the SAS System attempts to determine the engine (either the default or a compatibility engine) that should be assigned to the specified data library libref. Under OS/2, the SAS System looks at the file extensions that exist in the given folder and uses the following rules to determine which engine should be assigned to the libref:

- If the folder contains SAS data sets from only one of the supported native library engines (not including XPORT), the libref is assigned to that engine.
- If there are no SAS data sets in the given folder, the libref is assigned to the default engine.
- If the folder contains SAS data sets from more than one engine, this is called a mixed mode library. The libref is then assigned to the default engine. A message is printed in the SAS log that informs you the libref is assigned to a mixed mode library.

Note: It is always more efficient to specify the engine name than to have the SAS System determine the correct engine. △

You can use the ENGINE system option to specify the default engine the SAS System uses when it detects a mixed mode library or a library with no SAS files. The valid values for the ENGINE system option are V8, V7, V6, and BASE. By default, the ENGINE option is set to V8. For more information, see the ENGINE system option in "ENGINE" on page 340.

Using Data Libraries

The libref is a label or alias that is temporarily assigned to a folder so that the storage location (the full path name, including drive and folder) is in a form that is

recognized by the SAS System. A libref exists only during the session in which it is created. It is a logical concept that describes a physical location, rather than something that is physically stored with the file. To name a libref, follow the rules for user-supplied SAS names. See the section about SAS language rules in *SAS Language Reference: Dictionary* for more information. For detailed information about using SAS data libraries. For detailed information about using SAS data libraries see *SAS Language Reference: Concepts*.

There are several ways to specify a libref:

- Use the New Library dialog box, which is described in SAS Online Help.
- Use the LIBNAME statement or function as described in "Assigning SAS Libraries Using the LIBNAME Statement or Function" on page 63.
- Define an environment variable as described in "Assigning SAS Libraries Using Environment Variables" on page 65.

Note: You can eliminate the LIBNAME statement by directly specifying the drive name and the data set name within quotes. An example follows:

```
data "d:\a";
```

△

Accessing the New Library Dialog Box Using the Graphical User Interface

You can assign librefs by using the graphical user interface (GUI) either with the New Library toolbar icon (the file cabinet), the LIBASSIGN command, or from within the Explorer window. All of these options open the New Library dialog box where you can specify librefs and engines for multiple folders.

To open the New Library dialog box by using the toolbar, click on the New Library icon.

To open the New Library dialog box by using a command, type *libassign* in the command box.

To open the New Library dialog box by using the Explorer window:

1. Select the Library folder.
2. Select **New...** from the **File** menu or right-mouse click the Library folder and select **New...** from the pop-up menu to open the New... dialog box.
3. Select the **Library** folder and click on **OK**.

Note: When a second Explorer window is open on the right side of the SAS workspace, you can bypass the New... dialog box to open the New Library dialog box directly if you click on the **Libraries** folder using the right mouse button and select **New...**. △

For more information about using the New Library window and the Explorer, see the SAS online Help.

Assigning SAS Libraries Using the LIBNAME Statement or Function

You can use the LIBNAME statement or function to assign librefs and engines to one or more folders, including the working folder. The examples in this section use the LIBNAME statement. For information about the LIBNAME function, see *SAS Language Reference: Dictionary*.

The LIBNAME statement has the following syntax:

LIBNAME *libref <engine-name> 'SAS-data-library'*

For details about the LIBNAME statement see “LIBNAME” on page 301 and *SAS Language Reference: Dictionary*.

CAUTION:
The words CON, NUL, KBDS, PRN, COM1 - COM3, and LPT1 - LPT3 are reserved words under OS/2. Do not use these reserved words as librefs. △

Assigning a Libref to a Single Folder

If you have Version 8 SAS data sets that are stored in the C:\MYSASDIR folder, you can submit the following LIBNAME statement to assign the libref TEST to that folder:

```
libname test V8 'c:\mysasdir';
```

This statement indicates that Version 8 SAS files that are stored in the folder C:\MYSASDIR are to be accessed using the libref TEST. Remember that the engine specification is optional.

Assigning a Libref to the Working Folder

If you want to assign the libref MYCURR to your current SAS System working folder, use the following LIBNAME statement:

```
libname mycurr '.';
```

Assigning a Libref to Multiple Folders

If you have SAS files that are located in multiple folders, you can treat these folders as a single SAS data library by specifying a single libref and concatenating the folder locations, as in the following example:

```
libname income ('c:\revenue','d:\costs');
```

This statement indicates that the two folders, C:\REVENUE and D:\COSTS, are to be treated as a single SAS data library. When you concatenate SAS data libraries, the SAS System uses a protocol (a set of rules) to access the libraries, depending on whether you are accessing the libraries for READ, WRITE, or UPDATE. Furthermore, you may concatenate multiple libraries by specifying only their librefs, as in the following example:

```
libname sales (income revenue);
```

This statement indicates that two libraries that are identified by the librefs INCOME and REVENUE are treated as a single SAS data library whose libref is SALES. For more information, see “Understanding How Concatenated SAS Data Libraries Are Accessed” on page 68.

Note: The concept of library concatenation also applies when you are specifying system options, such as the SASHELP and SASMSG options. See “Syntax for Concatenating Libraries in SAS System Options” on page 315 for information. △

Assigning Engines

If you want to use another access method or another engine instead of the Version 8 engine, you can specify the engine name in the LIBNAME statement. For example, if you want to access Release 6.10 SAS data sets from your Version 8 SAS session, you can specify the V6 engine in the LIBNAME statement, as in the following example:

```
libname oldlib V6 'c:\sas610';
```

Similarly, if you plan to share SAS files between Version 8 under OS/2 and Version 6 under OS/2, you should use the V6 engine when you are assigning a libref to the SAS data library. Here is an example of specifying the V6 engine in a LIBNAME statement:

```
libname lib6 V6 'c:\sas6';
```

The V6 engine is particularly useful in your Version 8 SAS session if you are going to access the same SAS files from a Version 6 SAS session. Remember that while Version 8 can read Version 6 SAS data sets, Version 6 cannot read Version 8 SAS data sets.

For more information about using engine names in the LIBNAME statement, see "Using SAS Files from Other Releases with Version 8 for OS/2" on page 71 and "Reading BMDP, OSIRIS, and SPSS Files" on page 76. You can also see the section about the LIBNAME statement in *SAS Language Reference: Dictionary*.

Using the LIBNAME Statement in SAS Autoexec Files

If you prefer, you can store LIBNAME statements in your SAS autoexec file so that the librefs are available as soon as the SAS System initializes. For example, your SAS autoexec file may contain the following statements:

```
libname test 'c:\mysasdir';
libname mylib ('c:\mydata','d:\tempdata');
libname oldlib V6 'c:\sas6';
```

For more information about how to create and use a SAS autoexec file, see "SAS Autoexec File" on page 14 .

Assigning Multiple Librefs and Engines to a Folder

If a folder contains SAS files that were created by several different engines, only those SAS files that were created with the engine that is assigned to the given libref can be accessed by using that libref. You can assign multiple librefs with different engines to a folder. For example, the following statements are valid:

```
libname one V6 'c:\mydir';
libname two V8 'c:\mydir';
```

Data sets that are referenced by the libref ONE are created and accessed by using the compatibility engine (V6), whereas data sets that are referenced by the libref TWO are created and accessed using the default engine (V8). You can also have multiple librefs (using the same engine) for the same SAS data library. For example, the following LIBNAME statements assign the librefs MYLIB and INLIB (both using the V8 engine) to the same SAS data library:

```
libname mylib V8 'c:\mydir\datasets';
libname inlib V8 'c:\mydir\datasets';
```

Because the engine name and SAS data library specifications are the same, the librefs MYLIB and INLIB are identical and can be used interchangeably.

Assigning SAS Libraries Using Environment Variables

You can also assign a libref to a SAS data library by using environment variables instead of the LIBNAME statement or function. An *environment variable* equates one string to another within the OS/2 environment. The SAS System recognizes two kinds of environment variables: SAS environment variables and OS/2 environment variables. When you use a libref in a SAS statement, the SAS System first looks to see if that libref has been assigned to a SAS data library by a LIBNAME statement. If it has not,

the SAS System searches first for any defined SAS environment variables and then for any OS/2 environment variables that match the specified libref. There are two ways of defining an environment variable to the SAS System:

- Use the SET system option to define a SAS (internal) environment variable.
- Issue an OS/2 SET command to define an OS/2 (external) environment variable.

CAUTION:
You cannot assign engines to environment variables. If you use environment variables as librefs, you must accept the default engine. △

The availability of environment variables makes it simple to assign resources to the SAS System prior to invocation.

SET System Option

You can use the SET system option to define a SAS environment variable. For example, if you store your permanent SAS data sets in the C:\SAS\MYSASDATA folder, you can use the following SET option in the SAS command or in your SAS configuration file to assign the environment variable TEST to this SAS data library:

```
-set test c:\sas\mysasdata
```

When you assign an environment variable, the SAS System does not resolve the environment reference until the environment variable name is actually used. For example, if the TEST environment variable is defined in your SAS configuration file, the environment variable TEST is not resolved until it is referenced by the SAS System. Therefore, if you make a mistake in your SET option specification, such as misspelling a folder name, you do not receive an error message until you use the environment variable in a SAS statement.

Because OS/2 filenames can contain spaces or single quotes as part of their names, you should enclose the name of the physical path in double quotes when you specify the SET system option. If you use SET in an OPTIONS statement, you must use quotation marks around the filename. For the complete syntax of the SET system option, see “SET” on page 372.

Any environment variable name that you use with a system option in your SAS configuration file must be defined as an environment variable before it is used. For example, in the following example the SET system option must appear before the SASUSER option that uses the environment variable TEST:

```
-set test "d:\mysasdir"
-sasuser !test
```

In the following example, environment variables are used with concatenated libraries:

```
-set dir1 "c:\sas\base\sashelp"
-set dir2 "d:\sas\stat\sashelp"
-sashelp (!dir1 !dir2)
```

Note that when you reference environment variables in your SAS configuration file or in a LIBNAME statement in your SAS programs, you must precede the environment variable name with an exclamation point (!).

It is recommended that you use the SET system option in your SAS configuration file if you invoke the SAS System through an OS/2 shadow.

OS/2 SET Command

You can execute an OS/2 SET command prior to invoking the SAS System to create an OS/2 environment variable. You must define the environment variable prior to

invoking the SAS System; you cannot define environment variables for SAS System use from an OS/2 window from within a SAS session.

Operating Environment Information SAS can recognize environment variables only if they have been assigned in the same context that invokes the SAS session. That is, you must either define the environment variable in the OS/2 CONFIG.SYS file that runs when OS/2 starts (thus creating a global variable), or define the variable in an OS/2 window from which you then start SAS.

If you define an environment variable in an OS/2 window, and then start SAS from a desktop shortcut, SAS will not recognize the environment variable. △

The environment variables that you define with the SET command can be used later within the SAS System as librefs. In the following example, the OS/2 SET command is used to define the environment variables PERM and BUDGET:

```
SET PERM="C:\MYSASDIR"
SET BUDGET="D:\SAS\BUDGET\DATA"
```

When you reference an external environment variable (one that is assigned with the OS/2 SET command) in your SAS programs (such as in a DATA or MERGE statement or in a SAS command), a note that informs you that the environment variable has been used is written to the SAS log. SAS does not recognize the environment variable as a libref until after you use it at least once during your SAS session, so the library name does not appear in the Libraries dialog box or in the LIBNAME window until then.

Listing Libref Assignments

If you need to find out which file a libref points to, you can use the SAS Explorer window, the LIBNAME command, or the LIBNAME statement to list all the assigned librefs.

If you are running the SAS System interactively, use the SAS Explorer window to view the active librefs. The SAS Explorer window lists all the librefs that are active for your current SAS session, along with the engine and the physical path for each libref. Any environment variables that you have defined as librefs are listed, provided that you have used them in your SAS session. If you have defined an environment variable as a libref but have not used it yet in a SAS program, the SAS Explorer window does not list it.

Note: You can use the LIBNAME command to open the LIBNAME window, which lists the active libraries. △

You can use the following LIBNAME statement to write the active librefs to the SAS log:

```
libname _all_ list;
```

Clearing Librefs

You can clear a libref by using the following form of the LIBNAME statement:

LIBNAME *libref*|_all_ <clear>;

If you specify a libref, only that libref is cleared. If you specify the keyword _ALL_, all the librefs that you have assigned during your current SAS session are cleared. (SASUSER, SASHELP, and WORK remain assigned.)

Note: When you clear a libref that is defined by an environment variable, the variable remains defined, but it is no longer considered a libref. (That is, it is no longer

listed in the Libraries dialog box). You can use the variable in another LIBNAME statement to create a new libref. △

The SAS System automatically clears the association between librefs and their respective data libraries at the end of your job or session. If you want to associate the libref with a different SAS data library during the current session, you do not have to end the session or clear the libref. The SAS System automatically reassigns the libref when you use it to name a new library.

Understanding How Concatenated SAS Data Libraries Are Accessed

When you use the concatenation feature to specify more than one physical folder for a libref, the SAS System uses the following protocol to determine which folder is accessed. The protocol that these examples illustrate applies to all SAS statements and procedures that access SAS files, such as the DATA, UPDATE, and MODIFY statements in the DATA step and the SQL and APPEND procedures.

Input and Update Access

When a SAS file is accessed for input or update, the first SAS file that is found by that name is the one that is accessed. For example, if you submit the following statements, and the file OLD.SPECIES exists in both folders, the one in the C:\MYSASDIR folder is printed:

```
libname old ('c:\mysasdir','d:\saslib');
proc print data=old.species;
run;
```

The same would be true if you opened OLD.SPECIES for update with the FSEDIT procedure.

Output Access

If the data set is accessed for output, it is always written to the first folder, provided that the folder exists. If the folder does not exist, an error message is displayed. For example, if you submit the following statements, the SAS System writes the OLD.SPECIES data set to the first folder (C:\MYSASDIR) and replaces any existing data set with the same name:

```
libname old ('c:\mysasdir','d:\saslib');
data old.species;
   x=1;
   y=2;
run;
```

If a copy of the OLD.SPECIES data set exists in the second folder, it is not replaced.

Accessing Data Sets with the Same Name

However, when you use the DATA and SET statements to access data sets with the same name, a different behavior occurs. For example, suppose that you submit the following statements, and suppose that TEST.SPECIES exists only in the second folder, D:\MYSASDIR:

```
libname test ('c:\sas','d:\mysasdir');
data test.species;
   set test.species;
```

```
   if value1='y' then
      value2=3;
run;
```

In this case, the DATA statement opens TEST.SPECIES for output according to the output rules; that is, the SAS System opens a data set in the first of the concatenated libraries (C:\SAS). The SET statement then opens the existing TEST.SPECIES data set in the second (D:\MYSASDIR) folder, according to the input rules. Therefore, the original TEST.SPECIES data set is not updated; rather, two TEST.SPECIES data sets exist, one in each folder.

Using the SASUSER Data Library

The SAS System automatically creates a SAS data library with the libref SASUSER. This library contains, among other SAS files, your user profile catalog. By default under OS/2, the SASUSER libref points to a folder called SASUSER that is located under the working folder of your current SAS session. If a SASUSER folder does not exist, the SAS System creates one. You can use the SASUSER system option to make the SASUSER libref point to a different SAS data library. For more information about your profile catalog, see "Profile Catalog" on page 15. See "SASUSER" on page 370 for more information about the SASUSER system option. The SAS System stores other files besides the profile catalog in the SASUSER folder. For example, the sample data sets that are provided with SAS/ASSIST software are stored in this folder.

CAUTION:

You cannot change the engine that is associated with the SASUSER data library. The SASUSER data library is always associated with the V8 engine. If you try to assign another engine to this data library, you receive an error message. Therefore, even if you have set the ENGINE system option to another engine, any SAS files that are created in the SASUSER data library are Version 8 SAS files. △

Using the WORK Data Library

The WORK data library is the storage place for temporary SAS files. By default under OS/2, the WORK data library is created as a subfolder of the SASWORK folder. This subfolder is named #TD*nnnnn*, as discussed in "WORK Data Library" on page 16. Temporary SAS files are available only for the duration of the SAS session in which they are created. At the end of that session, they are deleted automatically. By default, any file that is not assigned a two-level name is automatically considered to be a temporary file. A special libref of WORK is automatically assigned to any temporary SAS data sets that are created. For example, if you run the following SAS DATA step to create the data set SPORTS, a temporary data set named WORK.SPORTS is created:

```
data sports;
   input @1 sport $10. @12 event $20.;
   cards;
volleyball co-recreational
swimming   100-meter freestyle
soccer     team
;
```

If you display the SAS Explorer window now, you will see the SPORTS data set in the WORK folder.

You can display all the temporary data sets that are created during this session from the SAS Explorer window by double-clicking on the Libraries folder icon and then

double-clicking on the Work folder icon. From the Active Libraries (LIBNAME) window, double-click on the Work folder icon to see the temporary data sets.

CAUTION:

You cannot change the engine that is associated with the WORK data library. The WORK data library is always associated with the V8 engine. If you try to assign another engine to this data library, you receive an error message. Therefore, even if you have set the ENGINE system option to a different engine, any SAS files that are created in the WORK data library are Version 8 SAS files. △

Using an Environment Variable

You can use an environment variable in your WORK data library specification, similar to the method that was described earlier using the SASUSER system option. Use this technique when you do not want to use the default location for your WORK data library. You can put something similar to the following in your SAS configuration file to set up an environment variable to use for your WORK data library:

```
-set myvar c:\ tempdir
-work !myvar
```

The SET option associates the MYVAR environment variable with the C:\TEMPDIR folder. Then the WORK system option tells the SAS System to use that folder for the SAS WORK data library.

Note: Do not assign an environment variable named TEMP or TMP, as these variable names are usually already used by OS/2. △

Using the USER Libref

Although by default SAS files with one-level names are temporary and are deleted at the end of your SAS session, you can use the USER libref to cause SAS files with one-level names to be stored in a permanent SAS data library. For example, the following statement causes all SAS files with one-level names to be permanently stored in the C:\MYSASDIR folder:

```
libname user 'c:\mysasdir';
```

When you set the USER libref to a folder as in the previous example, if you want to create or access a temporary data set you must specify a two-level name for the data set with WORK as the libref. You can also assign the USER libref when you invoke the SAS System by using the USER system option. For more information on the USER system option, refer to “USER” on page 382 and the section about the USER system option in *SAS Language Reference: Dictionary* .

Note: You can assign other engines to the USER libref if you want the data sets that are saved with one-level names to be stored in a format for use with other releases of the SAS System. △

Accessing SAS Files from Multiple SAS Sessions

If you are running multiple SAS sessions, whether on a single machine or across a network, you can have multiple access to the same SAS file when you are reading from it.

If you have SAS/SHARE installed, the VIEWTABLE window and the FSEDIT or FSVIEW windows allow multiple users to edit the same SAS file. When you edit a data

set by using the VIEWTABLE window, you can set the editing mode to either Row Level Edit Access or Table Level Edit Access. When you select Table Level Edit Access, only you have access to the data set. Row Level Edit access allows multiple users to access the same SAS file, but only one user can access and make changes to a single record (observation) at a time.

To open a data set in the VIEWTABLE window, from the SAS Explorer window:

1. double-click on the Libraries icon
2. double-click on the library that contains the data set
3. double-click on the data set.

To edit the data set, select

Edit ▶ Edit Mode

and then select either **Table Level Edit** or **Row Level Edit**.

When you edit a data set by using FSEDIT or FSVIEW, you can set the update mode to either MEMBER or RECORD. When you select MEMBER mode, only you have access to the data set. When you select RECORD mode, multiple users can write to the same SAS file but only one user can update a single record (observation) at a time.

To open a data set by using FSEDIT or FSVIEW:

1. type FSEDIT or FSVIEW in the command bar
2. double-click on the library name in the Select a Member dialog box
3. double-click on the data set name.

To edit the data set, select

Edit ▶ Update...

and then select either the **MEMBER** or **RECORD** radio button.

The RSASUSER system option, described in "RSASUSER" on page 364, allows you to share the SASUSER data library. If multiple users need update access to common SAS data sets, use SAS/SHARE software.

For details about rules for multiple user access to the same data set and its members, see the SAS online Help and *SAS/SHARE User's Guide*.

Using SAS Files from Other Releases with Version 8 for OS/2

This section discusses using SAS files from previous releases of SAS for OS/2. For information about transporting files from other operating environments to the Windows environment, see *Moving and Accessing SAS Files across Operating Environments*.

If you intend to use Version 6 or 7 SAS files only with Version 8, you will see improved performance if you convert your earlier files to Version 8 files.

Note: If you will be using your SAS files only with previous releases of the SAS System, do not convert your files to Version 8 files. △

Table 3.4 on page 72 summarizes the steps that you must complete to use your existing SAS data sets and catalogs with Version 8.

Table 3.4 Migrating Existing SAS Files to Version 8

Release	SAS Data Sets	SAS Catalogs
6.12 for OS/2	Transparent	CPORT and CIMPORT
6.11 for OS/2	Transparent	CPORT and CIMPORT
6.10 for OS/2	Transparent	CPORT and CIMPORT
6.08 for OS/2	Transparent	CPORT and CIMPORT
6.06 for OS/2	V606 engine	CPORT (C16PORT) and CIMPORT
6.03 and 6.04	V604 engine	CPORT and CIMPORT

As the table shows, you can use SAS data sets that were created starting with Release 6.06 without first converting them. All Version 6 SAS catalogs must be transported to be used in Version 8.

Note: Version 8 of SAS under OS/2 does not read SAS data sets under Windows directly as it did in Version 6. To use SAS data sets under Windows in the OS/2 operating environment, you must either transport them or use Cross Environment Data Access (CEDA). For more information, see *Moving and Accessing SAS Files across Operating Environments*. △

Using Version 6 SAS Data Sets in Version 8

This section discusses the use of SAS data sets that were created using Releases 6.06 through 6.12.

Using Release 6.08 through Release 6.12 Data Sets

If your SAS library contains only Version 6 SAS files, Version 8 automatically determines the appropriate engine to use for Release 6.08 and later SAS data sets. If your SAS files are in a mixed-mode library that possibly contains SAS data sets from Version 6 and Version 8, you must specify the *engine* parameter in the LIBNAME statement or else the engine defaults to V8.

For example, if you know that the 'c:\mydata' SAS library contains only Version 6 files, submit the following SAS statements to print a Version 6 SAS data set that is named OS2DATA.SALEFIGS:

```
libname os2data 'c:\mydata';
proc print data=os2data.salefigs;
   title 'Sales Figures';
run;
```

Where all SAS files in the library are Version 6 SAS files, you can omit the engine parameter because SAS automatically detects the V6 engine.

Using the same example, if you are unsure or if you know that the SAS library is a mixed-mode library, you must specify the engine name in the LIBNAME statement to access the V6 files:

```
libname windata v6 'c:\mydata';
proc print data=windata.salefigs;
   title 'Sales Figures';
run;
```

Using Release 6.03 and Release 6.04 SAS Data Sets

The V604 engine enables you to read from and write to Release 6.03 and Release 6.04 SAS data sets directly from your Version 8 SAS session. (Remember that there is no difference between Release 6.04 and Release 6.03 SAS data sets.) This feature is useful when you have SAS data sets that you want to share between Release 6.04 for PCs and Version 8 under Windows. The V604 engine is supported only for SAS data sets (member type DATA). For example, if you want to print a Release 6.04 SAS data set that is named MYLIB.FRUIT, submit the following statements from a Version 8 session:

```
libname mylib v604 'c:\sas604';
proc print data=mylib.fruit;
run;
```

While it is useful to be able to access SAS data sets that were created in a previous release of the SAS System, you cannot take advantage of the full functionality of the Version 8, which includes creating indexes and database views, unless you convert the data sets that were created with earlier releases of the SAS System to Version 8 data sets.

Converting Version 6 SAS Data Sets

While access to Version 6 SAS data sets is quite easy when you use the V6 engines, you may want to consider converting your SAS files to Version 8 format if you access them often and no longer need to read the files from Release 6. The data set format of Version 8 is more efficient than the Version 6 format, and there are new Version 8 features that cannot be used unless the data sets are converted.

The following SAS statements use the COPY procedure to convert all the Version 6 SAS data sets in the OS2DATA SAS data library to Version 8 and to format and store the new data sets in the OS2DATA8 library:

```
libname os2data v6 'c:\mydata';
libname os2data8 V8 'd:\newdata';
proc copy in=os2data out=os2data8 memtype=data;
run;
```

Alternatively, you can use the DATA step to do the conversion, as in the following example. This technique works well if you want to convert only one or two data sets in a particular SAS data library.

```
libname os2data v608 'c:\mydata';
libname os2data8 V8 'd:\newdata';
data os2data8.eggplant;
   set os2data.eggplant;
run;
```

You can also use the CPORT and CIMPORT procedures to perform the conversion. Remember that when you convert a data set to Version 8 format, you can no longer read that data set from Version 6.

Note: Do not convert your Version 6 files to Version 8 if you need to access the files from both versions. △

Creating Version 6 Data Sets

You may need to create Version 6 SAS data sets from your Windows SAS session. This is similar to reading Version 6 data sets in that you use the V6 engine. For example, the following SAS statements use the V6 engine to create a SAS data set that is named QTR1. The raw data are read from the external file that is associated with the fileref MYFILE.

```
libname os2data v6 'c:\mydata';
filename myfile 'c:\qtr1data.dat';
data os2data.qtr1;
   infile myfile;
   input saledate amount;
run;
```

Converting Version 6 SAS Catalogs in Version 8

Differences exist in the internal structures of the operating environments. Therefore, you must use the CPORT and CIMPORT procedures to convert Version 6 SAS catalogs that were created under OS/2 to Version 8 format before you can use the catalogs in your Version 8 SAS session under OS/2. Here are the steps to follow:

1. Using the CPORT procedure in your Version 6 SAS session, create a transport file that contains the SAS catalog to be converted.
2. Transfer the file (perhaps on a network or diskette) to a place that your Version 8 SAS session can read it.
3. Use the CIMPORT procedure from your Version 8 SAS session to read the transport file and create a converted SAS catalog.

For information about using the CPORT and CIMPORT procedures, see *Moving and Accessing SAS Files across Operating Environments* and *SAS Procedures Guide*.

Converting Release 6.06 SAS Catalogs

If you are converting directly from Release 6.06 to Version 8, you can use the CPORT procedure in Release 6.06 to create a transport file, and then use the Version 8 CIMPORT procedure to convert the catalog to a Version 8 catalog. However, the TOOLBOX catalog entries are not portable if you use CPORT from a Release 6.06 session.

An alternative way to convert Release 6.06 catalogs is to use the C16PORT procedure that is provided in Releases 6.10 through 6.12. SAS provided the C16PORT procedure to convert the 16-bit catalogs that were created with Release 6.06 under OS/2 to a 32-bit format that the SAS System can use. You can use the C16PORT procedure from within one of these earlier releases of the SAS System to create a catalog that can later be read by Version 8. (The C16PORT procedure is not available in Version 8.)

Here are the steps to follow to convert your SAS catalogs from Release 6.06 under Windows to Version 8:

1. While in your Release 6.10, Release 6.11, or Release 6.12 session, use the C16PORT procedure (described in the documentation for those releases) to create a transport file that contains the SAS catalog from Release 6.06.
2. Use the CIMPORT procedure from your Release 6.10, Release 6.11, or Release 6.12 SAS session to read the transport file and create a converted SAS catalog.
3. Use a Version 8 SAS session to access the converted catalog.

If you want to convert a catalog that currently exists on another machine running Release 6.06 for OS/2, you must first transfer the file (perhaps on a network or a diskette) to a place where your Version 8 session can read it.

The following example uses the C16PORT procedure in Release 6.12 to create a transport file from the INLIB.CAT catalog, then creates a Release 6.12 catalog (OUTLIB.CAT) by using the CIMPORT procedure.

```
   /* Folder where catalog    */
   /* 'cat.sc2' resides       */
libname inlib 'c:\cat606';
   /* Folder  where catalog   */
   /* 'cat.sc8' will reside   */
libname outlib 'c:\cat612';
proc c16port file='transprt' c=inlib.cat;
run;
proc cimport infile='transprt' c=outlib.cat;
run;
```

The Release 6.12 SAS catalog can now be read by Version 8. For information about the CPORT and CIMPORT procedures, see *SAS Procedures Guide* and *Moving and Accessing SAS Files across Operating Environments*.

Using Version 8 SAS Files with Previous Releases

It is possible to move SAS files between your Version 8 SAS session under OS/2 and earlier releases of the SAS System under OS/2 by using the CPORT and CIMPORT procedures.

Note: When you transport Version 8 files to previous releases, Version 8 features are not available from the transport file △

For information about PROC CPORT, PROC CIMPORT, and about back-porting SAS/AF applications, see *SAS Procedures Guide*. For more information about back-porting SAS/EIS applications, see the SAS online Help for SAS/EIS. For more information about transporting files in general, see *Moving and Accessing SAS Files across Operating Environments*.

Using Version 5 and Other Remote Host SAS Files in Version 8

You can directly access remote host SAS data sets, including Version 5, Release 6.07, and Release 6.08 SAS data sets that were created on a remote host. For example, you may have a Version 5 SAS data set under VSE or a Release 6.07 data set under OS/390. The following explanation uses Version 5 but also applies to these other releases.

To directly access a Version 5 SAS data set that was created on a remote host computer, use SAS/CONNECT software to establish a communication link between your local SAS session under OS/2 and a remote SAS session under the other host. Once you have established a link, you can remotely submit SAS statements to process the Version 5 data on the remote host computer, or you can download the Version 5 data set to a Version 8 data set under OS/2. SAS/CONNECT software provides a convenient one-step way to transport SAS files from system to system. For more information about downloading files, see *SAS/CONNECT User's Guide*.

As an alternative, you can create a transport data set from your Version 5 data set and transport your file from the host to OS/2. Then you can use the XPORT engine to create a Version 8 file from this transport file. If you are accessing Version 6 remote SAS data sets, you can also use PROC CPORT to create the transport file.

The method for transporting data sets differs from the method for transporting SAS catalogs. For more information about transporting SAS files, see SAS Institute's Web page and select Technical Support.

Reading BMDP, OSIRIS, and SPSS Files

Version 8 of the SAS System provides three interface library engines that enable you to access external data files directly from a SAS program: the BMDP, OSIRIS, and SPSS engines. These engines are all READ-only. Because they are sequential engines (that is, they do not support random access of data), these engines cannot be used with the POINT= option in the SET statement or with the FSBROWSE, FSEDIT, or FSVIEW procedures. You can use PROC COPY or a DATA step to copy the system file to a SAS data set and then perform these functions on the SAS data set. Also, because they are sequential engines, some procedures (such as the PRINT procedure) give a warning message that the engine is sequential. With these engines, the physical filename that is associated with a libref is an actual filename, not a folder. This is an exception to the rules about librefs.

You can also use the CONVERT procedure to convert BMDP, OSIRIS, and SPSS files to SAS data files; see "CONVERT" on page 276 for more information.

BMDP Engine

The BMDP interface library engine enables you to read BMDP DOS files from the BMDP statistical software package directly from a SAS program. The BMDP engine is a READ-only engine. The following discussion assumes that you are familiar with the BMDP *save file* terminology.

To read a BMDP save file, you must issue a LIBNAME statement that explicitly specifies that you want to use the BMDP engine. In this case, the LIBNAME statement takes the following form:

LIBNAME *libref* BMDP <'filename'>;

In this example, *libref* is a SAS libref and *filename* is the BMDP physical filename. If the libref appears previously as a fileref, you can omit *filename* because the physical filename that is associated with the fileref is used. This engine can read only BMDP save files that were created under DOS.

Because there can be multiple save files in a single physical file, you reference the CODE= value as the member name of the data set within the SAS language. For example, if the save file contains CODE=ABC and CODE=DEF and the libref is MYLIB, you reference them as MYLIB.ABC and MYLIB.DEF. All CONTENT types are treated the same, so even if member DEF is CONTENT=CORR, it is treated as CONTENT=DATA.

If you know that you want to access the first save file in the physical file, or if there is only one save file, you can refer to the member name as _FIRST_. This is convenient if you do not know the CODE= value.

BMDP Engine Examples

In the following example, the physical file MYBMDP.DAT contains the save file ABC. This example associates the libref MYLIB with the BMDP physical file, then runs the CONTENTS and PRINT procedures on the save file:

```
libname mylib bmdp 'mybmdp.dat';
   proc contents data=mylib.abc;
```

```
run;
proc print data=mylib.abc;
run;
```

The following example uses the LIBNAME statement to associate the libref MYLIB2 with the BMDP physical file. Then it prints the data for the first save file in the physical file:

```
libname mylib2 bmdp 'mybmdp.dat';
proc print dat=mylib2._first_;
run;
```

OSIRIS Engine

Because the Inter-University Consortium on Policy and Social Research (ICPSR) uses the OSIRIS file format for distribution of its data files, the SAS System provides the OSIRIS interface library engine to support the many users of the data from ICPSR and to be compatible with PROC CONVERT. For details, see "CONVERT" on page 276.

The READ-only OSIRIS engine enables you to read OSIRIS data and dictionary files directly from a SAS program. These files must be stored in EBCDIC format. This means you must have downloaded the OSIRIS files from your host computer in binary format. The following discussion assumes that you are familiar with the OSIRIS file terminology. *

To read an OSIRIS file, you must issue a LIBNAME statement that explicitly specifies that you want to use the OSIRIS engine. In this case, the LIBNAME statement takes the following form:

LIBNAME *libref* OSIRIS '*data-filename*' DICT='*dictionary-filename*';

In this form of the LIBNAME statement, *libref* is a SAS libref, *data-filename* is the physical filename of the OSIRIS data file, and *dictionary-filename* is the physical filename of the OSIRIS dictionary file. The *dictionary-filename* argument can also be an environment variable name or a fileref. (Do not put it in quotes if it is an environment variable name or fileref.) The DICT= option must appear because the engine requires both files.

OSIRIS data files do not have member names. Therefore, you can use whatever member name you like. You can use the same OSIRIS dictionary file with different OSIRIS data files. Simply write a separate LIBNAME statement for each one.

The layout of an OSIRIS data dictionary is consistent across operating environments. The reason is that the OSIRIS software does not run outside the OS/390 environment, but the engine is designed to accept an OS/390 data dictionary on any other operating environment under which the SAS System runs. It is important that the OSIRIS dictionary and data files not be converted from EBCDIC to ASCII; the engine expects EBCDIC data. There is no specific file layout for the OSIRIS data file. The file layout is dictated by the contents of the OSIRIS dictionary file.

OSIRIS Engine Example

In the following example, the data file is MYOSIRIS.DAT, and the dictionary file is MYOSIRIS.DIC. The example associates the libref MYLIB with the OSIRIS files and then runs PROC CONTENTS and PROC PRINT on the data:

```
libname mylib osiris 'myosiris.dat'
        dict='myosiris.dic';
proc contents data=mylib._first_;
```

* See the documentation that is provided by the Institute for Social Research for more information.

```
run;
proc print data=mylib._first_;
run;
```

SPSS Engine

The SPSS interface library engine enables you to read SPSS/PC files directly from a SAS program. This is a READ-only engine. The following discussion assumes that you are familiar with the SPSS *save file* terminology. *

To read an SPSS save file, you must issue a LIBNAME statement that explicitly specifies that you want to use the SPSS engine. In this case, the LIBNAME statement takes the following form:

LIBNAME *libref* SPSS <'*filename*'>;

In this example, the *libref* argument is a SAS libref, and *filename* is the SPSS physical filename. If the libref appears also as a fileref, you can omit *filename* because the physical filename that is associated with the fileref is used. The SPSS/PC native file format, as well as the export file format, is supported. The engine determines which format is used and reads the save file accordingly. Native files must be created under PC DOS. Export files can originate from any operating environment.

Because SPSS/PC files do not have internal names, you can refer to them by any member name that you like. (The example in this discussion uses _FIRST_ .)

SPSS Engine Example

The following example associates the libref MYLIB with the physical file MYSPSS.DAT in order to run PROC CONTENTS and PROC PRINT on the save file:

```
libname mylib spss 'myspss.dat';
proc contents data=mylib._first_;
run;
proc print data=mylib._first_;
run;
```

Transferring SAS Files between Operating Environments

For complete information about transferring SAS files between operating environments, see *Moving and Accessing SAS Files across Operating Environments*.

Accessing Database Files with SAS/ACCESS Software

SAS/ACCESS software provides an interface between the SAS System and several database management systems (DBMS) that run under OS/2. The interface consists of three procedures and an interface view engine. With these, you can perform the following tasks:

ACCESS procedure
: creates SAS/ACCESS descriptor files that describe DBMS data to the SAS System. It also can create a SAS data file from DBMS data.

* See the documentation that is provided by SPSS Inc. for more information.

DBLOAD procedure
loads SAS or other DBMS data into a DBMS file.

SQL Procedure Pass-Through Facility
accesses data from several different relational DBMSs, including ODBC and SYBASE.

interface view engine
enables you to use descriptor files in SAS programs to access DBMS data directly and enables you to specify descriptor files in SAS programs to update, insert, or delete DBMS data directly.

For more information about using SAS/ACCESS software under OS/2, see *SAS/ACCESS Software for PC File Formats: Reference* and other SAS/ACCESS documentation.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS® Companion for the OS/2® Environment, Version 8*, Cary, NC: SAS Institute Inc., 1999. 448 pp.

SAS® Companion for the OS/2® Environment, Version 8

ISBN 1-58025-521-3

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

1st printing, October 1999