



CHAPTER

7

Using External Files

<i>Writing to External Files</i>	67
<i>Using the FILE Statement</i>	67
<i>Dynamically Changing Files in a DATA Step</i>	68
<i>Using the FILE Command</i>	68
<i>Writing to Print Files</i>	69
<i>Using the PRINT Option</i>	69
<i>Using File-specification Options</i>	70
<i>Writing to Nonprint Files</i>	70
<i>Reading from External Files</i>	70
<i>Using the INFILE Statement</i>	70
<i>Dynamically Changing Files in a DATA Step</i>	71
<i>Using the %INCLUDE Statement</i>	71
<i>Using the INCLUDE Command</i>	72

Writing to External Files

You can use the FILE statement or the FILE command to write to an external file.

Using the FILE Statement

The FILE statement specifies the current output file for PUT statements in the DATA step. (See *SAS Language Reference: Dictionary* for a complete description of the PUT statement.)

The specified output file must be an external file, not a SAS data library. It cannot be a CMS MACLIB nor an OS/390 data set. If no FILE statement is specified, then any PUT statements in your SAS program write to the SAS log. The FILE statement is executable; therefore, you can use it in conditional processing (in an IF/THEN statement, for example).

When multiple FILE statements are present, the PUT statement builds and writes output lines to the file that was specified in the most recent FILE statement.

For complete information about the FILE statement, see “FILE” on page 204.

The syntax of the FILE statement is:

```
FILE file-specification <options>
```

file-specification

identifies the file. It can be in the following forms:

Form	Example
fileref	<code>report</code>
filename filetype	<code>'report listing'</code>
filename filetype filemode (or SFS directory)	<code>'report listing b'</code>
fileref(filename)	<code>mydir(report)</code>
fileref (filename filetype)	<code>mydir(report listing)</code>
CMS pipeline	<code>> report listing b</code>
reserved filerefs	<code>LOG</code> or <code>PRINT</code>

See “Identifying an External File” on page 57 for details.

options

describe the output file’s characteristics and specify how it is to be written with a PUT statement. Options that are not host-dependent are documented in *SAS Language Reference: Dictionary*. For information about CMS-specific options, see “FILE” on page 204.

You can use options to do the following:

- define variables that will contain information about the external file
- specify special open and close processing
- specify file characteristics.
- indicate that the file specification is a pipeline.

Dynamically Changing Files in a DATA Step

Use the FILEVAR= option in the FILE statement to dynamically change output files in the middle of a DATA step. For example:

```
data _null_;
  length x $20;
  x='old file a';
  file cc filevar=x ;
  put 'line one';
  x='new file b';
  file cc filevar=x ;
  put 'line two';
run;
```

These statements place 'line one' in OLD FILE A while NEW FILE B contains 'line two'.

Using the FILE Command

The FILE command writes the entire contents of the current window to an external file without removing text from the window. You can specify a previously assigned fileref or an external file.

The form of the FILE command is

```
FILE <file-specification> <options-list>
```

The *file-specification* argument is in one of the forms given in “Identifying an External File” on page 57 . The file specification cannot be applied to a CMS MACLIB

or OS/390 data set. For information about the available options, see the help for base SAS or *SAS Language Reference: Dictionary*.

For example, suppose you specify this FILENAME statement:

```
filename sasfile 'myfile saspgm b';
```

The following command-line command will copy the text from the Program Editor window to the disk file MYFILE SASPGM B:

```
file sasfile
```

If you do not give a value for *file-specification*, the file from the previous FILE or INCLUDE command is used. If you have not issued previous FILE or INCLUDE commands, then an error message tells you that no default file exists.

The FILE command does not create a PRINT file, even if A is specified in the RECFM= option. In order to create a PRINT file from any window, you need to use the PRINT command instead.

Aggregate external files can also be used in a FILE statement. Suppose that a FILENAME statement assigns a fileref to an SFS directory as an aggregate external file. To illustrate a further point, suppose that the FILENAME statement consists of an SFS directory specification only, as follows:

```
filename mydir 'fpool:mysuer.dir';
```

When the filemode is not specified, the FILE statement assumes a filetype of SAS. In the following FILE statement, the contents of the Program Editor window are copied to PGM1 SAS FPOOL:MYUSER.DIR:

```
file mydir(pgm1);
```

Note that aggregate syntax can be applied only to SFS directories and CMS minidisks.

Writing to Print Files

When you write SAS output to external files, you need to know the differences between *print* files and *nonprint* files.

A print file contains carriage-control information (also called ASA control characters) in column 1 of each line. These characters (blank, 0, -, +, and 1) control the operation of a printer: skipping lines, beginning a new page, and so on. They do not normally appear on a printout. If you do not expect to print the external file, you do not need to write to a print file.

When you write to a print file in a DATA step, SAS shifts all column specifications in the PUT statement one column to the right to accommodate the carriage-control characters in column 1.

Using the PRINT Option

You can declare an external file as a print file if you specify the PRINT option in the FILE or INFILE statement. For example, the following SAS program writes one line to MYFILE1 FILE, which is declared to be a print file by the FILE statement:

```
filename out1 'myfile1 file a';
data _null_;
  file out1 print;
  put 'line to myfile1';
run;
```

Using File-specification Options

You can declare a file to be a print file if you include an A (for ASA carriage control) in the RECFM= option in the FILENAME statement. The following SAS program shows an example of this second method:

```
filename out2 'myfile2 file a' recfm=va;
data _null_;
  file out2;
  put 'line to myfile2';
run;
```

Note: You can use either of these techniques to read from or write to a file with carriage-control characters. If a print file is being read, the first byte is stripped off and is not returned as part of the data. If you wish to include the carriage-control bytes as part of the data, do not declare the file as a print file. Δ

Writing to Nonprint Files

A nonprint file that is written by SAS does not contain any characters to control printer operation. The NOPRINT option declares the file as a nonprint file, even if an A is specified in the RECFM= option. For example, the following SAS program writes a nonprint file to MYFILE3 FILE, even though an A has been included in the RECFM= option.

```
filename out3 'myfile3 file a' recfm=va;
data _null_;
  file out3 noprint;
  put 'line to myfile3';
run;
```

Whether you create a print file or a nonprint file, SAS provides default values for most characteristics of the file; these defaults are adequate in most cases. “FILENAME” on page 209 lists the default file characteristics for print and nonprint files.

Reading from External Files

You can read from an external file in a SAS DATA step by specifying it in the INFILE statement. Alternatively, you can use the INCLUDE command or the %INCLUDE statement.

Using the INFILE Statement

In a SAS DATA step, the INFILE statement specifies which external file is to be read by a subsequent INPUT statement. Every external file that you want to read must have a corresponding INFILE statement. (See *SAS Language Reference: Dictionary* for a complete description of the INPUT statement.)

This section provides a brief overview of INFILE statement syntax. For complete information about the INFILE statement, see “INFILE” on page 218.

The form of the INFILE statement is

```
INFILE file-specification <options-list>;
```

file-specification

identifies the file. It may be in the following forms:

Form	Example
fileref	<code>indata</code>
fileref(member)	<code>maclib(mem1)</code>
fileref(filename filetype)	<code>mydir(myfile in)</code>
filename filetype filemode (or SFS directory)	<code>'myfile in b'</code>
filename filetype	<code>'myfile in'</code>
CMS pipeline	<code>< myfile in b</code>
reserved fileref	<code>CARDS</code>

options-list

controls how a file is written to or read from. When specifying more than one option, use a blank space to separate each option.

You can use these options to do the following:

- define variables that will contain information about the external file
- specify special open and close processing
- specify file characteristics.

See “INFILE” on page 218 for valid option values under CMS. For information about other options that you can specify in the INFILE statement, see *SAS Language Reference: Dictionary*.

Dynamically Changing Files in a DATA Step

Use the FILEVAR= option in the INFILE statement to dynamically change input files in the middle of a DATA step. For example:

```
data read;
  x='mydata march a';
  y='mylib names b';
  infile a filevar=x;
  input Q1 $1. Q2 $1.;
  infile in filevar=y;
  input name $20.;
run;
```

The data set READ consists of the variables Q1, Q2, and NAME; values for Q1 and Q2 come from the file MYDATA MARCH A, whereas the value of NAME comes from the file MYLIB NAMES B.

Using the %INCLUDE Statement

The %INCLUDE statement includes SAS statements from external files, from earlier points in the same job or session, or from the terminal, and it submits those statements automatically. The form of the %INCLUDE statement is

```
%INCLUDE file-specification </options-list>;
```

The *file-specification* argument is in one of the forms given in “Identifying an External File” on page 57. For information about the available options, see *SAS*

Language Reference: Dictionary. For example, suppose you issue the following FILENAME statement:

```
filename mypgm 'saspgm tastest c';
```

You can issue the following %INCLUDE statement to copy in and execute the SAS statements that are stored in the file SASPGM TASTEST C:

```
%include mypgm;
```

You can also use the %INCLUDE statement in interactive line mode to recall previously entered statements, or in noninteractive mode to allow input from the terminal. See “%INCLUDE” on page 217 and *SAS Language Reference: Dictionary* for more information about the %INCLUDE statement.

Aggregate external files can be specified in a %INCLUDE statement as well:

```
filename mydir 'a';
.
.
%include mydir(saspgm);
```

As shown in the previous example, the filetype can be left out of %INCLUDE specifications of aggregate SFS directories and minidisks. For aggregates other than CMS MACLIBs, an unspecified filetype defaults to SAS. The previous example copies in and executes the file SASPGM SAS A.

Using the INCLUDE Command

The INCLUDE command enables you to display an entire external file in a window. The form of the INCLUDE command is

INCLUDE *file-specification*< *options-list*>

The *file-specification* argument is in one of the forms given in “Identifying an External File” on page 57. For information about the available options, see in *SAS Language Reference: Dictionary*. For example, suppose you specify this FILENAME statement:

```
filename sasfile tape 'tap3';
```

The following command-line command includes the file in the Program Editor window:

```
include sasfile
```

If you do not give a value for *file-specification*, the file from the previous FILE or INCLUDE command is used. If you have not issued any previous FILE or INCLUDE commands, an error message tells you that no default file exists.

If the specified fileref has an A in its record format (as specified by the RECFM= option of the FILENAME statement), the INCLUDE command treats the file as a print file.

When you specify a minidisk or SFS directory as an external aggregate file with the INCLUDE command you can leave off the filetype, which will default to **SAS**.

See in *SAS Language Reference: Dictionary* for more information about the INCLUDE command.