CHAPTER

9

WK*n* Chapter

Note to UNIX and OS/390 Users

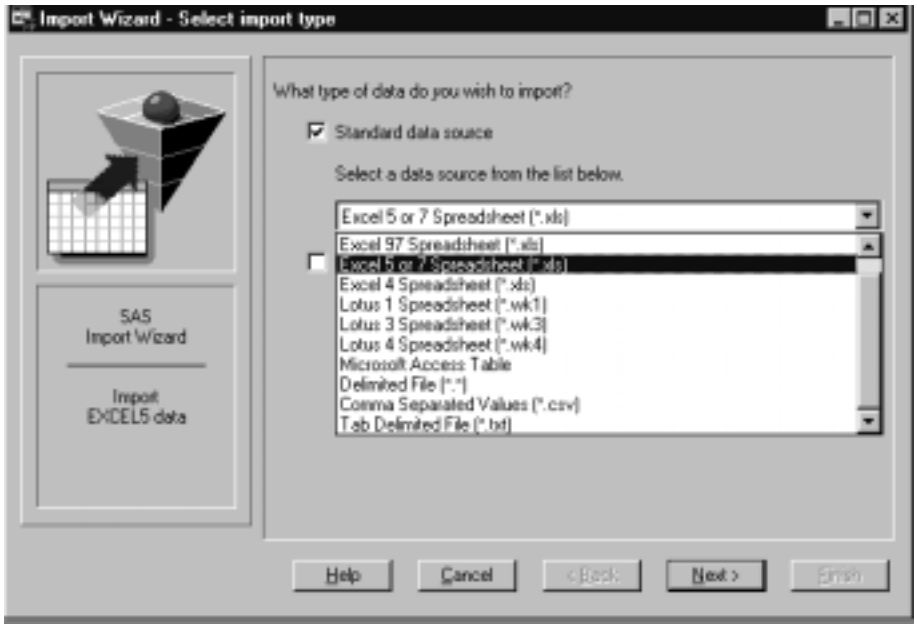
If you are running this SAS/ACCESS interface under the UNIX or OS/390 operating environment, this chapter does not apply to you. Instead, see Chapter 3, "DBF and DIF Procedures," on page 33. Under UNIX and PC hosts, you can use these procedures to convert a DBF or DIF file to a SAS data set or a SAS data set to a DBF or DIF file. Under OS/390, you can use PROC DBF *only* to convert a DBF file to a SAS data set or a SAS data set to a DBF file.

Import/Export Facility

UNIX and PC users can access WK*n* data through the Import/Export facility or by using the IMPORT and EXPORT procedures. An overview is included in Chapter 5, "Import/Export Facility and Procedures," on page 51.

To use the point-and-click interface from a SAS PROGRAM EDITOR window, select the **File** menu and then select the **Import Data** or **Export Data** item. Information about how to import or export WK*n* data is available from the Help button. The following is a sample Import window:

Display 9.1 Import Window



To write code to import or export WK*n* data, refer to the detailed descriptions of the IMPORT and EXPORT procedures in the *SAS Procedures Guide*. This documentation also includes several examples.

Understanding WK*n* Essentials

This chapter introduces SAS System users to WK*n* files. It focuses on the terms and concepts that help you use the SAS/ACCESS interface and includes descriptions of

- WK*n* files
- WK*n* file naming conventions
- WK*n* file data types
- how the SAS/ACCESS interface works.

WK1, WK3, and WK4 files contain data in the form of Lotus 1-2-3 spreadsheets. The SAS/ACCESS interface supports Releases 4 and 5 of the Lotus 1-2-3 WK4 file format. Unless otherwise noted, WK1, WK3, and WK4 files are referred to collectively throughout this report as WK*n* files, where *n* stands for 1, 3, or 4.

Note: The SAS/ACCESS interface does not support the .123 format for files from Lotus SmartSuite 97 software. △

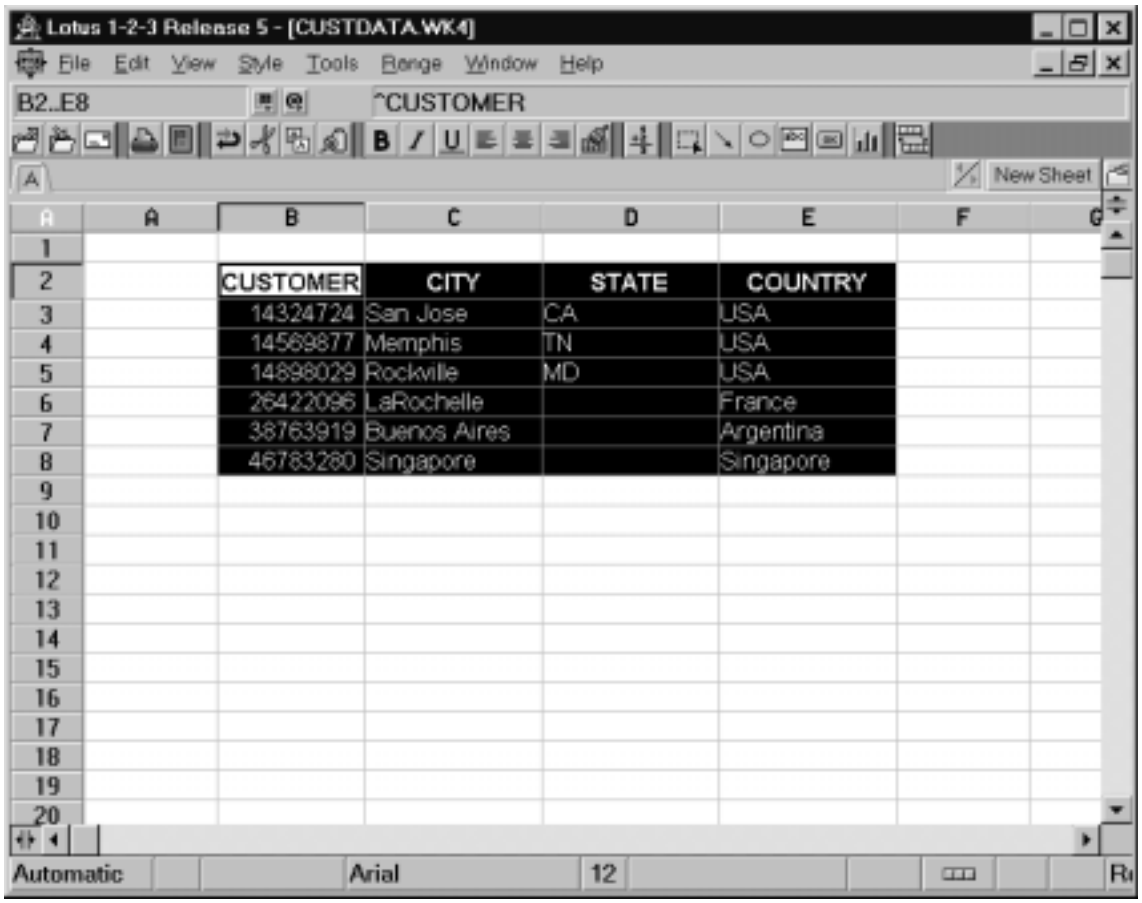
WK*n* Files

Various software products, such as the Lotus 1-2-3 spreadsheet and database system, enable you to use spreadsheet or database files to enter, organize, and perform calculations on data. Spreadsheets are most often used for general ledgers, income statements, and other types of financial record keeping. Database files also enable you to organize related information, such as, the data in an accounts-receivable journal.

In both spreadsheets and database files, the data are organized according to certain relationships among data items. These relationships are expressed in a tabular form, in columns and rows. Each *column* represents one category of data, and each *row* can hold one data value for each column.

A Lotus 1-2-3 worksheet is an electronic spreadsheet consisting of a grid of 256 columns and 8,192 rows. The intersection of a column and a row is called a *cell*. Display 9.2 on page 119 illustrates a portion of a standard 1-2-3 worksheet.

Display 9.2 Columns and Rows of Data in a WK*n* File



Column letters for each column appear above the worksheet. Columns are lettered A through IV (A to Z, AA to AZ, BA to BZ, and so on to IV). Row numbers for each row appear to the left of the worksheet. Rows are numbered 1 to 8,192. For .WK1 files, only one worksheet (worksheet A) is allowed per file. For .WK3 and .WK4 files, up to 256 worksheets (worksheets A-IV) are allowed. The SAS/ACCESS interface to WK*n* files uses only one worsheet, however, and defaults to worksheet A.

A *range* is a subset of cells in a worksheet. A range is identified by its address, which begins with the name of the top left cell and ends with the name of the bottom right cell separated by two periods. For example, the range B3..D6 is the range address for a rectangular block of 12 cells whose top left cell is B3 and whose bottom right cell is D6 (as shaded in the figure).

You can give a name to a range and use the name in commands and formulas instead of the range address in Lotus 1-2-3. A range name can be up to 15 characters long and should not contain any spaces. For example, if the range B3..D6 is named

GRADE_TABLE, then the formula @AVG(GRADE_TABLE) has the same value as @AVG(B3..D6) and identifies the data in the range.

For more information on ranges and their naming conventions, see the documentation that accompanies your Lotus 1-2-3 software.

WK*n* File Naming Conventions

Filenames must also follow operating environment specific conventions, so check the documentation that comes with your Lotus 1-2-3 product or other software products for further information. The following conventions apply to WK*n* filenames:

- Under Windows 95, Windows 98, Windows NT, and OS/2, the ACCESS and DBLOAD procedures support long names that are specified in the PATH= statement (such as **`path= 'c:\sasdemo\library\new_customer_1999.wk4';`**). However, WK*n* files with long names might not be accepted by some versions of Lotus 1–2–3.
- Filenames can contain up to eight characters, unless the file was stored using the high-performance file system (HPFS) format that is supported by OS/2. HPFS filenames are supported by the ACCESS and DBLOAD procedures, but they might not be accepted by some versions of Lotus 1-2-3.
- Filenames start with a letter, and they can contain any combination of the letters A through Z, the digits 0 through 9, the underscore (_), the hyphen (-), and spaces (blanks) within filenames.
- Filenames can contain spaces. Filenames that contain spaces or lowercase letters are supported by the ACCESS and DBLOAD procedures, but they might not be accepted by some versions of Lotus 1-2-3.

WK*n* Data Types

Lotus 1-2-3 software has two data types: character and numeric. Lotus 1-2-3 character data may be entered as labels or formula strings; Lotus 1-2-3 numeric data may be entered as numbers or formulas.

Character data are generally considered text and can include dates and numbers if prefixes are used to indicate character data and to align the data in the cell. For example, in Lotus 1-2-3, the value **`"110 Maple Street`** uses the double quote prefix and aligns the label on the right side of the cell.

Numeric data can include numbers (0 through 9), formulas, and cell entries that begin with one of the following symbols: +, $, @, –, or #.

Numeric data also can include *date and time values*. In Lotus 1-2-3 software, a date value is the integer portion of a number that can range from 01 January 1900 to 31 December 2099, that is, 1 to 73050. A Lotus 1-2-3 software time value is the decimal portion of a number that represents time as a proportion of a day. For example, 0.0 is midnight, 0.5 is noon, and 0.999988 is 23:59:59 (on a 24-hour clock). While a number can have both a date and a time portion, the formats in Lotus 1-2-3 display a number only in a date format or a time format. For information on how the SAS/ACCESS interface handles date and time values and formats, see “How the SAS System Handles Date and Time Values” on page 121.

When you create an access descriptor, the interface software uses the column types and formats in the .WK *n* file to determine the corresponding SAS variable formats. The SAS System generates its default formats based on the values that you specify for the SCANTYPE, SKIPROWS, and GETNAMES statements. You can change the formats generated by the software interface. For more information, see “How the SAS/ ACCESS Interface to WK*n* Files Works” on page 129.

When you browse a view descriptor, any data value that does not match the column type (character or numeric) specified in the descriptor is treated as a missing value. This is the default action. However, you can use the MIXED=YES statement to convert numeric data values in a character column to their character representation when you create an access descriptor.

You can also set the SS_MIXED environment variable to **YES** in your SAS configuration file so that both numeric and character data are displayed as SAS character data. Add this line to your SAS configuration file:

```
-SET SS_MIXED YES
```

See "Setting Environment Variables" on page 124 for more information on environment variables. For more information on changing the column type, refer to the sections on WK-specific procedure statements later in this chapter.

How the SAS System Handles Date and Time Values

The conversion of date and time values between SAS data sets and Lotus 1-2-3 spreadsheets is transparent to users. However, you are encouraged to understand the differences between them.

Lotus 1-2-3 date and time values and formats are described in "WK*n* Data Types" on page 120.

Datetime Conversions in the ACCESS Procedure

As described earlier in this chapter, a Lotus date value is the integer portion of a number that represents the number of days between January 1, 1900 and a specified date. A Lotus time value is a decimal portion of a number that represents time as a portion of the day. For example, 0.0 is 12:00:00 a.m. and 0.9999884 is 11:59:59 p.m. While a number can have both a date and a time portion, the formats in Lotus 1-2-3 display a number only in a date format or in a time format. For example, for 1:00 p.m., March 12, 1994, the Lotus 1-2-3 date value is 34405, the time value is 0.5416667, and the datetime value is 34405.5416667.

The SAS System handles date and time values differently than Lotus. A SAS date value is an integer that represents the number of days between January 1, 1960 and a specified date. A SAS time value is an integer that represents the number of seconds since midnight of the current day. When a date and a time are both present, the SAS System stores the value as the number of seconds since midnight, January 1, 1960. For example, for 1:00 p.m., March 12, 1994, the SAS date value is 12489, and the SAS time value is 46800. Therefore, the SAS datetime value is 1079096400. *

To convert a Lotus 1-2-3 datetime format to a SAS datetime format, you need a SAS datetime format in the view descriptor. For example, changing the default SAS numeric format (15.2) to a SAS date format in the descriptor causes the Lotus 1-2-3 date value (based on January 1, 1900) to be converted to an equivalent SAS date value (based on January 1, 1960). In other words, the Lotus 1-2-3 numeric value for January 1, 1960 (which is 21916) is converted to the equivalent SAS representation of January 1, 1960 (which is 0) only if a SAS datetime format is assigned in the descriptor for that column. Otherwise, the Lotus 1-2-3 value of 21916 is treated as a SAS numeric value of 21916.

The table below shows how the SAS System uses a Lotus 1-2-3 internal datetime value to convert to a SAS internal datetime value.

---

* In this description, datetime (in lowercase) refers to any value or format that represents a date, a time, or both a date and time.

Table 9.1 Value-to-Format Conversions

For a SAS format	SAS System uses
date	if the Lotus datetime value is less than 60: integer portion of the Lotus 1-2-3 datetime value - 21915
	if the Lotus datetime value is greater than 60: integer portion of the Lotus 1-2-3 datetime value - 21916
time	decimal portion of the Lotus 1-2-3 datetime value * 86400
date-and-time	if the Lotus datetime value is less than 60: (integer and decimal portion of the Lotus 1-2-3 datetime value - 21915) * 86400
	if the Lotus datetime value is greater than 60: (integer and decimal portion of the Lotus 1-2-3 datetime value - 21916) * 86400

Datetime Conversions in the DBLOAD Procedure

If a SAS variable is specified with a date, time, or datetime format in the FORMAT statement, the interface-view engine converts that SAS datetime format into the equivalent Lotus 1-2-3 datetime format when the new WK*n* file is created.

However, if a SAS datetime format is not specified in the input SAS data set, you have to assign a format by using a PROC DBLOAD FORMAT statement. Doing so assigns a Lotus 1-2-3 datetime format to the SAS variable when the variable is loaded into a WK*n* file. If you do not assign a SAS datetime format, the SAS numeric-datetime value is written to the WK*n* file. Because SAS dates are based on January 1, 1960, and Lotus 1-2-3 dates are based on January 1, 1900, the datetime value in the WK*n* file will be inaccurate.

To maintain a SAS variable format in the input data set, yet change it only while the DBLOAD procedure is in progress, use the FORMAT statement in PROC DBLOAD. This statement enables you to assign a temporary format to a SAS variable for the duration of the procedure without affecting the input SAS data set.

For example, if the SAS format for the BIRTHDAT variable in the MYDATA.SASEMPS access descriptor is left at the default 15.2 format, you can specify the FORMAT statement in the PROC DBLOAD statement. This specification changes the variable's format to DATE7. while you are creating and loading the WK*n* file. When you load the WK*n* file, the DATE7. format becomes an equivalent Lotus column format, *DD-MON-YY*. When the DBLOAD procedure has completed, the SAS format for the BIRTHDAT variable returns to the 15.2 format.

You can specify the FORMAT statement when you invoke the DBLOAD procedure to assign a temporary format to the variables in your input SAS data set. For more information, see the description of the FORMAT statement in "DBLOAD Procedure Statements for WK*n* Files" on page 128.

ACCESS Procedure Data Conversions

Use PROC ACCESS to define descriptors that identify spreadsheet data and the conversions necessary to use the data in SAS programs. The Lotus label data type is formatted as a SAS character type, and the Lotus 1-2-3 number data type is formatted as a SAS numeric type.

Fonts, attributes, and colors in the WK*n* files are not read into the SAS data sets. However, the ACCESS procedure supports most of the WK*n* number formats and

automatically converts them to the corresponding SAS formats. Any WK*n* data strings longer than 200 characters are truncated while being converted into SAS data sets, and any SAS data file created from WK*n* files can only contain up to 256 variables and 8,192 observations.

Table 9.2 on page 123 shows the default SAS System variable formats that the ACCESS procedure assigns to each type of WK*n* file data. WK*n* file numeric data include date and time values. See "How the SAS System Handles Date and Time Values" on page 121 for more information.

If WK*n* file data fall outside of the valid SAS data ranges, you receive an error message in the SAS log when you try to access the data.

The SAS/ACCESS interface does not fully support the Lotus 1-2-3 hidden and text formats. WK*n* data in hidden format are displayed in SAS data sets; however, you can drop the hidden column when you are creating the access descriptor. If you want to display the formula in the text format, add a label prefix character to indicate that the formula entry is a label. Otherwise, the results of the formula are displayed.

If you have set the SS_MIXED environment variable to **YES**, the numerical values in WK*n* files are converted to character strings in SAS data sets if the corresponding SAS variable type is specified as character.

DBLOAD Procedure Data Conversions

This section explains how SAS data are read into Lotus 1-2-3 data when a table is loaded. In this conversion, the SAS character data type is converted into the Lotus 1-2-3 label type and the SAS numeric type is converted into the Lotus 1-2-3 number type.

The SAS/ACCESS interface automatically converts SAS formats to the same or associated Lotus 1-2-3 formats and column widths. However, you can temporarily assign other formats and column widths to SAS variables by using the FORMAT statement. Table 9.2 on page 123 shows the SAS System variable types and formats and the WK*n* data types, formats, and column widths that you can assign them to.

Note: The FORMAT statement in PROC DBLOAD only changes the format of SAS variables while you are creating and loading the WK*n* files. When the procedure is completed, the formats of SAS variables return to their original settings. △

WK*n* date and time values are numeric data. See "How the SAS System Handles Date and Time Values" on page 121 for more information.

Table 9.2 Converting SAS System Variable Formats to WK*n* File Data

SAS Variable Format		WK*n* File Data			
Type	Data Format	Data Type	Column Format	Column Width	Number
Char	$*w*.	LABEL	DEFAULT	*w*	
Char	$CHAR*w*.	LABEL	DEFAULT	*w*	
Num	*w.d*	NUMBER	FIXED	*w*	*d*
Num	F*w.d*	NUMBER	FIXED	*w*	*d*
Num	E*w.d*	NUMBER	SCIENTIFIC	*w*	*d*
Num	DOLLAR*w.d*	NUMBER	CURRENCY	*w*	*d*
Num	PERCENT*w.d*	NUMBER	PERCENT	*w*	*d*
Num	COMMA*w.d*	NUMBER	COMMA	*w*	*d*

SAS Variable Format		WK*n* File Data			
Type	Data Format	Data Type	Column Format	Column Width	Number
Num	BEST*w*.	NUMBER	DEFAULT	*w*	
Num	BEST*w*.	NUMBER	GENERAL	*w*	
Num	DATE5.	NUMBER	DD-MON	7	
Num	DATE7.	NUMBER	DD-MON-YY	10	
Num	MONYY5.	NUMBER	MON-YY	7	
Num	MMDDYY5.	NUMBER	MM-DD	6	
Num	MMDDYY8.	NUMBER	MM-DD-YY	9	
Num	TIME5.	NUMBER	HH-MM-SS	6	
Num	TIME8.	NUMBER	HH-MM-SS	9	
Num	TIME9.	NUMBER	HH-MM AM/ PM	9	
Num	TIME12.	NUMBER	HH-MM-SS AM/PM	12	

Setting Environment Variables

You can change the default behavior of the SAS/ACCESS interface by setting environment variables in your SAS configuration file. You can set four SAS/ACCESS environment variables: SS_MISS NULLS, SS_MIXED, SS_NAMES, and SS_SCAN. Setting these variables in your SAS configuration file changes how the interface works by default .

The configuration file omits the following environment variables. When the environment variables are omitted, the default value for them is NO.

SS_MISS NULLS
By default, the DBLOAD procedure loads Lotus @NA cell values for missing values. Use this option to specify a null cell value instead. If set, missing values in a SAS data set will be displayed as blanks in the Lotus table.

SS_MIXED YES | NO
YES allows both Lotus 1-2-3 numeric and character data in a column to be displayed as SAS character data. The Lotus 1-2-3 numeric data are converted to their character representation when their corresponding SAS variable type is defined as character.

NO does not convert Lotus 1-2-3 numeric data in a column into SAS character data. Lotus 1-2-3 numeric data are read in as SAS missing values when their corresponding SAS variable type is defined as character. NO is the default.

Setting the SS_MIXED environment variable changes the default value of the MIXED statement in PROC ACCESS.

SS_NAMES YES | NO
YES in PROC ACCESS generates SAS variable names from column names in the first row of the worksheet or the specified range of the worksheet and reads data from the second row. YES in PROC DBLOAD writes column names using SAS variable names or SAS variable labels to the first row of the new WK*n* file, reads data from the data set, and writes them to the WK*n* file beginning with the second row.

NO in PROC ACCESS generates the SAS variable names VAR0, VAR1, VAR2, and so on, and reads data from the first row of the worksheet or specified range.

NO in PROC DBLOAD reads the data from the data set and writes them to the WK*n* file beginning with the first row. NO is the default.

Setting the SS_NAMES environment variable changes the default value of the GETNAMES statement in PROC ACCESS and the PUTNAMES statement in PROC DBLOAD.

SS_SCAN YES | NO | *number-of-rows*

YES scans the data type and format of rows in a worksheet or specified range after skipping the number of rows specified in the SKIPROWS statement. SS_SCAN finds the most common Lotus 1-2-3 data type and format in order to generate the default SAS data type and format. If a number of rows is specified, SAS/ACCESS software scans only the data type and format from these rows.

NO uses the type and format of the first row in a worksheet or specified range after skipping the number of rows specified in SKIPROWS to generate the default SAS data type and format. NO is the default.

Number-of-rows scans the type and format of the specified number of rows only. Setting the number of rows is more efficient because data are read only from the specified number of rows rather than from the entire file.

Setting the SS_SCAN environment variable changes the default value of the SCANTYPE statement in PROC ACCESS.

ACCESS Procedure: WK*n* Specifics

Chapter 2, "ACCESS Procedure Reference," on page 11 describes the generic options and procedure statements that enable you to create access descriptors, view descriptors, and SAS data files from PC file format data. The following section describes the PC file-specific statements you use in the SAS/ACCESS interface to WK*n* data.

ACCESS Procedure Statements for WK*n* Files

To create an access descriptor, you use the DBMS=WK*n* option and six database-description statements: PATH=, GETNAMES, RANGE, SCANTYPE, SKIPROWS, and WORKSHEET. These database-description statements supply WK*n*-specific information to the SAS System and must immediately follow the CREATE or UPDATE statement that specifies the access descriptor to be created or updated. In addition to the database-description statements, you can use editing statements when you create an access descriptor. These editing statements must follow the database-description statements.

Database-description statements are only required when you create access descriptors. Because WK*n* information is stored in an access descriptor, you do not need to repeat this information when you create view descriptors.

The SAS/ACCESS interface to WK*n* uses the following procedure statements in batch mode:

PROC ACCESS DBMS=WK1|WK3|WK4;
CREATE *libref.member-name.*ACCESS | VIEW ;
UPDATE *libref.member-name.*ACCESS | VIEW ;

PATH= '*path-and-filename*<.WK1 | .WK3 | .WK4>'| <'>*filename*<'>| *fileref*;
GETNAMES <=> YES | NO | Y | N;
RANGE <=><'>*range-name*<'>| '*range-address*';
SCANTYPE <=> YES | NO | Y | N | <*number-of-rows*>;

SKIPROWS <=> *number-of-rows-to-skip*;
WORKSHEET <=> *worksheet-letter*| <'>*worksheet-name*<'>;
ASSIGN <=> YES | NO | Y | N ;
DROP <'>*column-identifier-1*<'> <...<'>*column-identifier-n*<'>>;
FORMAT <'>*column-identifier-1*<'> <=> *SAS-format-name-1*
<...<'>*column-identifier-n*<'><=> *SAS-format-name-n*>;
LIST <ALL | VIEW | <'>*column-identifier*<'>>;
MIXED <=> YES | NO | Y | N;
RENAME <'>*column-identifier-1*<'> <=> *SAS-variable-name-1*
<...<'>*column-identifier-n*<'> <=> *SAS-variable-name-n*>;
RESET ALL | <'>*column-identifier-1*<'> <...<'>*column-identifier-n*<'>>;
SELECT ALL | <'>*column-identifier-1*<'> <...<'>*column-identifier-n*<'>>;
SUBSET *selection-criteria* ;
TYPE *column-identifier-1*<=> C | N <...*column-identifier-n* <=> C | N>;
UNIQUE <=> YES | NO | Y | N ;

RUN ;

The QUIT statement is also available in PROC ACCESS. However, its use causes the procedure to terminate. QUIT is used most often in the interactive line and noninteractive modes to exit the procedure without exiting SAS.

GETNAMES <=> YES | NO | Y | N;
determines whether SAS variable names are generated from column names in the first row of the range when an access descriptor is created. When you update a descriptor, you are not allowed to specify the GETNAMES statement.

The GETNAMES statement is optional. If you omit it, the default value GETNAMES=NO is used, and the WK*n* interface generates the SAS variable names VAR0, VAR1, VAR2, and so on. If you specify GETNAMES=YES, SAS variable names are generated from column names in the first row of the range. GETNAMES=YES also sets the default value of SKIPROWS statement to 1.

You can change the default value from NO to YES by setting the SS_NAMES environment variable. See “Setting Environment Variables” on page 124 for more information on setting and changing environment variables.

The GETNAMES statement is a database-description statement, and it must follow the CREATE statement and precede any editing statements when you create a descriptor.

MIXED <=> YES | NO | Y | N;
determines whether to convert Lotus 1-2-3 numeric data values in a column to their character representation when the corresponding SAS variable is expecting a character value.

The MIXED statement is optional. You use it if you have both Lotus 1-2-3 numeric and character data in a column. Specifying YES allows both numeric and character data to be displayed as SAS character data. NO, the default, treats any data in a column that does not match the specified type as missing values.

You can change the default value to YES by setting the SS_MIXED environment variable. See “Setting Environment Variables” on page 124 for more information on setting and changing environment variables.

The MIXED statement is an editing statement and must follow any database descriptions when you create an access descriptor.

RANGE <=> <'>*range-name*<'> | '*range-address*';
subsets a specified section of a WK*n* file worksheet. The *range-name* is the name that is assigned to a range address within the worksheet. Range names can be up

to 15 characters long and are not case-sensitive. If you specify a range name, the name must have been previously defined in the WK*n* file. The *range-address* is identified by the top left cell that begins the range and the bottom right cell that ends the range within the WK*n* worksheet file. The beginning and ending cells are separated by two periods; for example, the range address C9..F12 indicates a cell range that begins at cell C9, ends at cell F12, and includes all cells in between.

The RANGE statement is optional. If you omit RANGE, the entire worksheet is accessed as the default range.

The RANGE is a database-description statement, and it must follow the CREATE statement and precede any editing statements when you create a descriptor.

SCANTYPE <=> YES | NO | Y | N | <*number-of-rows*>;

finds the most common 1-2-3 format for each column in a specified number of rows in an WK*n* worksheet to generate the SAS format. By default, SAS variable formats are generated from the 1-2-3 formats found in the first row of the worksheet or in the range of the worksheet if you specified a range.

The SCANTYPE statement is optional, and its default value is NO. If you specify YES, the ACCESS procedure scans the Lotus formats of all rows in each column of the range and uses the most common one to generate the default SAS format for each column. If you specify a number of rows, PROC ACCESS scans the specified number of rows only and returns the most common format.

If you specify the SKIPROWS statement, the ACCESS procedure skips the specified rows and starts scanning the 1-2-3 format from the next row. For example, if you specify SKIPROWS=3, PROC ACCESS skips the first three rows and begins scanning the format on the fourth row.

You can change the default value to YES by setting the SS_SCAN environment variable. See “Setting Environment Variables” on page 124 for more information on setting and changing environment variables.

Specifying SCANTYPE=0 is equivalent to specifying SCANTYPE=NO.

The SCANTYPE statement is a database-description statement, and must follow the CREATE statement and precede any editing statements when you create a descriptor.

SKIPROWS <=> *number-of-rows-to-skip*;

specifies the number of rows, beginning at the top of the range in the WK*n* file, to ignore when reading data from the WK*n* file. The default value for SKIPROWS is 0. The skipped (or ignored) rows often contain information such as column labels or names or underscores rather than input data.

If GETNAMES=YES, the default value of SKIPROWS automatically changes to 1. The first row of data and formats after SKIPROWS in a range is used to generate the SAS variable types and formats. However, you can use the SCANTYPE statement to scan the formats of specified rows and use the most common type and format to generate the default SAS variable types and formats.

The SKIPROWS statement is a database-description statement, and it must follow the CREATE statement and precede any editing statements when you create a descriptor.

TYPE *column-identifier-1* <=> C | N <...*column-identifier-n* <=> C | N >;

changes the expected data types of SAS variables. SAS data sets have two data types: character (C) and numeric (N). Spreadsheet files have the same two data types: character (for labels and formula strings) and numeric (for numbers and

formulas). Changing the default data type of a SAS variable in a descriptor file also changes its associated default format in the loaded file.

If you omit the TYPE statement, the database field types are generated from the PC file data types. You can change as many database field types as you want in one TYPE statement.

WORKSHEET <=> *worksheet-letter* | <'>*worksheet-name*<'>;

identifies a particular worksheet when reading from a WK*n* file that contains more than one worksheet. You can specify a worksheet name or a worksheet letter using the WORKSHEET statement. Worksheet names can be up to 15 characters long and are not case-sensitive. A worksheet letter is a one- or two-letter alpha character. For WK1 files, there is only one worksheet letter: worksheet A. For WK3 and WK4 files, there can be up to 256 different worksheet letters: worksheet A through worksheet Z and worksheet AA through worksheet IV. The default value is A. For example, specifying WORKSHEET=B identifies worksheet B from a group of worksheets.

The WORKSHEET statement is optional. The WORKSHEET statement is a database-description statement and must follow the CREATE statement and precede any editing statements when you create an access descriptor.

DBLOAD Procedure: WK*n* Specifics

Chapter 4, "DBLOAD Procedure Reference," on page 41 describes the generic options and procedure statements that enable you to create a PC file table and to insert data in it. The following section describes the file-specific statement you use in the SAS/ACCESS interface to WK*n*.

DBLOAD Procedure Statements for WK*n* Files

To create and load a WK*n* table, the SAS/ACCESS interface to WK*n* uses the following statements in batch mode:

PROC DBLOAD <DBMS=WK1|WK3|WK4>
<DATA= <*libref.*>*SAS-data-set*>;

PATH='*path-and-filename*<.WK1|.WK3|.WK4>' | <'>*filename*<'> | *fileref*;

PUTNAMES <=> YES | NO | Y | N;

ACCDESC= <*libref.*>*access-descriptor*;

DELETE *variable-identifier-1* <*...variable-identifier-n*>;

ERRLIMIT= *error-limit*;

FORMAT *SAS-variable-name-1 SAS-format-1* <=>
<*...SAS-variable-name-n SAS-format-n*>;

LABEL;

LIMIT= *load-limit* ;

LIST <ALL | COLUMNS | FIELDS | *variable-identifier*>;

RENAME *variable-identifier-1* <=> <'>*column-name-1*<'>
<*...variable-identifier-n* = <'>*column-name-n*<'>>;

RESET ALL | *variable-identifier-1* <*...variable-identifier-n*>;
<*...column-identifier-n* <=> C | N>;

WHERE *SAS-where-expression*;

LOAD;

RUN;

The QUIT statement is also available in PROC DBLOAD. However, its use causes the procedure to terminate. QUIT is used most often in the interactive line and noninteractive modes to exit the procedure without exiting SAS.

PUTNAMES <=> YES | NO | Y | N;
writes column names to the first row of the new WK*n* file. The column names can be default SAS variable names or, if you specify the LABEL statement, SAS variable labels. You can modify the column names using the RENAME statement.

The PUTNAMES statement is optional. If you omit PUTNAMES, data are read from the data set and written to the WK*n* file beginning in the first row of the WK*n* file, and no column names are written to the file.

You can change the default value to YES by setting the SS_NAMES environment variable. See "Setting Environment Variables" on page 124 for more information on setting and changing environment variables.

FORMAT *SAS-variable-name-1 SAS-format-1 <SAS-variable-name-n SAS-format-n>*;
assigns a temporary format to a SAS variable in the input SAS data set. This format temporarily overrides any other format for the variable. The assignment lasts only for the duration of the procedure. Assign formats to as many variables as you want in one FORMAT statement.

Use FORMAT when you want to change the format, column width, or the number of decimal digits for columns being loaded into the PC file. For example, if you change the SAS variable format 12.1 to DOLLAR15.2, the column format of the loaded data changes from a fixed numeric format with a column width of 12 and one decimal digit to a currency format with a column width of 15 and two decimal digits.

How the SAS/ACCESS Interface to WK*n* Files Works

The SAS/ACCESS interface accesses data in the Lotus 1-2-3 WK*n* files directly. It enables you to create SAS data sets from WK*n* files or directly read the WK*n* file data without creating SAS data sets. The interface does not allow you to update, add, or delete data in WK*n* files.

Accessing the Data

To access the data, the interface accesses a range in a worksheet as a table. If the range is not specified, the interface accesses the entire worksheet as a table. By default, the interface uses the Lotus 1-2-3 formats of columns in the first row of the range to determine the formats of variables in SAS/ACCESS descriptors.

However, you can manipulate where the interface begins to read data and what format the interface generates by using the SKIPROWS and SCANTYPE statements in the ACCESS procedure. SKIPROWS skips a specified number of rows before reading data. SCANTYPE finds the most common data type from among a specified number of rows within a WK*n* range (after skipping the number of rows specified in SKIPROWS) and uses it to generate the default format for SAS variables.

The ACCESS procedure enables you to create access descriptors and view descriptors for WK*n* files. You then can use the view descriptors as SAS data sets.

You can retrieve a subset of data using the WHERE statement .

To sort WK*n* file data, you must first extract the data from a WK*n* file and place them in a SAS data file, unless you are using the SQL procedure. (The SQL procedure enables you to present output data in a sorted order using the ORDER BY clause of the SELECT statement.) You can extract and sort WK*n* file data in one step with the OUT= option in the SORT procedure, using a view to the WK*n* file as input to PROC SORT.

Creating and Loading the Data

When you use PROC DBLOAD to create and load WK*n* files, the procedure translates the SAS data set into a WK*n* file. The file is stored in the location specified by the PATH= statement. Only one SAS data set can be loaded into a WK*n* file at one time. The loaded WK*n* file can contain only one worksheet. Lotus 1-2-3 then reads data from the loaded WK*n* file directly.

In the DBLOAD procedure, you can specify the PUTNAMES statement to place the SAS variable names in the first row of the spreadsheet and the first observation in the second row, and so on. If PUTNAMES is not specified, the first observation is placed in the first row, the second observation is placed in the second row, and so on. Columns do not have names. The formats for SAS variables are automatically converted to the corresponding Lotus 1-2-3 types and formats. See the descriptions of individual statements for more information on how the data and columns are read.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS/ACCESS® Software for PC File Formats: Reference, Version 8*, Cary, NC: SAS Institute Inc., 1999.

SAS/ACCESS® Software for PC File Formats: Reference, Verison 8

ISBN 1-58025–544–2

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

1st printing, October 1999