CHAPTER

3

DBF and DIF Procedures

Introduction

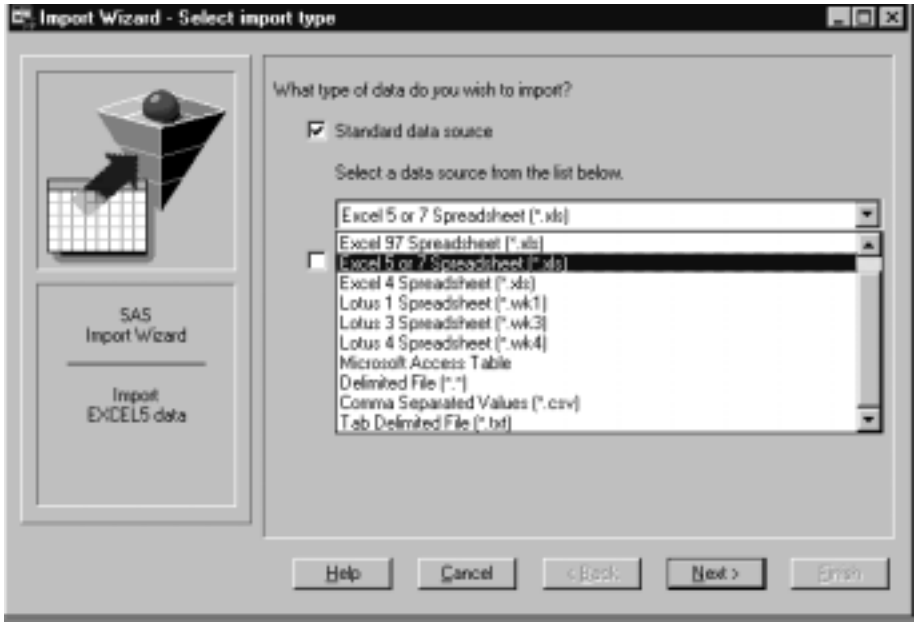
If you are running this SAS/ACCESS interface under the Windows, OS/2, OS/390, or UNIX operating environment, this chapter applies to you. Under UNIX and the PC hosts, you can use the DBF and DIF procedures to convert a DBF or DIF file to a SAS data set or a SAS data set to a DBF or DIF file. Under OS/390, 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 DBF (but not DIF) 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 the **Import Data** or **Export Data** item. Information about how to import or export DBF data is available from the Help button. The following is a sample Import window:

Display 3.1 Import Window



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

PROC DBF

converts a dBASE file to SAS data set or a SAS data set to a dBASE file

Syntax

PROC DBF *options*;

PROC DBF Options

DB2|DB3|DB4|DB5=*fileref* | *filename*
specifies the fileref or filename of a DBF file. When you use the FILENAME statement to assign the fileref, the statement must specify the filename plus a DBF extension (that is, **`filename myref '/my_dir/myfile.dbf'`**).

If you specify a filename instead of a fileref, you can only specify the name itself (omitting the DBF extension) and the file must be in the current directory. For example, this PROC DBF statement creates the EMP.DBF file (uppercase) from the MYLIB.EMPLOYEE data set:

```
proc dbf db5=emp data=mylib.employee;
```

You *cannot* specify **emp.dbf** or a full pathname (**proc dbf db5='/my/unix_directory/emp.dbf'**) in the DB*n*= option.

The DB*n* option must correspond to the version of dBASE with which the DBF file is compatible. You specify a DBF file with the DB*n* option, where *n* is 2, 3, 4, or 5. You can specify only one of these values. If you specify **DB4=myfile**, SAS looks for (and creates, depending on your options) a file called **MYFILE.DBF**, where the name is converted to uppercase.

DATA=<*libref.*>*member*
names the input SAS data set. Use this option if you are creating a DBF file from a SAS data set. If you use the DATA= option, do not use the OUT= option. If you omit the DATA= option, SAS software creates an output SAS data set from the DBF file.

OUT=<*libref.*>*member*
names the SAS data set that is created to hold the converted data. Use this option only if you do not specify the DATA= option.

If OUT= is omitted, SAS creates a temporary data set in the WORK library. (Under UNIX and OS/390, the temporary data set is named DATA1 [...DATA*n*]; under PCs, it is called _DATA_.) If OUT= is omitted or if you do not specify a two-level name in the OUT= option, the SAS data set that is created by PROC DBF remains available during your current SAS session, but it is not permanently saved.

Details

The DBF procedure converts dBASE files to SAS data sets that are compatible with the current release of the SAS System. This procedure can also be used to convert SAS data sets to DBF files.

PROC DBF produces one output file but no printed output. The output file contains the same information as the input file but in a different format.

The DBF procedure works with DBF files created by all the current versions and releases of dBASE (II, III, III PLUS, IV, and 5.0) and with most DBF files that are created by other software products.

Future versions of dBASE files might not be compatible with the current version of the DBF procedure. SAS Institute cannot be responsible for upgrading PROC DBF to support new versions of dBASE with each new version of SAS software. To use the DBF procedure, you must have a SAS/ACCESS interface to PC File Formats license.

Converting DBF Fields to SAS Variables

Numeric variables are stored in character form by DBF files. These numeric variables become SAS numeric variables when converting from a DBF file to a SAS data set. If a DBF numeric value is missing, the corresponding dBASE numeric field is filled with the character **9**, by default.

Character variables become SAS character variables. Any character variable of a length greater than 200 is truncated to 200. Logical fields become SAS character variables with a length of 1. Date fields become SAS date variables. When converting a DBF file to a SAS data set, fields whose data are stored in auxiliary DBF files (Memo and General fields) are ignored.

When a dBASE II file is translated into a SAS data set, any colons in dBASE variable names are changed to underscores in SAS variable names. Conversely, when a SAS data set is translated into a dBASE file, any underscores in SAS variable names are changed to colons in dBASE field names.

Converting SAS Variables to DBF Fields

Numeric variables are stored in character form by DBF files. SAS numeric variables become numeric variables with a length of 16 when converting from a SAS data set to a DBF file. A SAS numeric variable with a decimal value must be stored in a decimal format in order to be converted to a DBF numeric field with a decimal value. In other words, unless you associate the SAS numeric variable with an appropriate format in a SAS FORMAT statement, the corresponding DBF field will not have any value to the right of the decimal point. You

can associate a format with the variable in a SAS data set when you create the data set or by using the DATASETS procedure.

If the number of digits—including a possible decimal point—exceeds 16 a warning message is issued and the DBF numeric field is filled with the character **9**. All SAS character variables become DBF fields of the same length. When converting from a SAS data set to a DBF file that is compatible with dBASE III or later, SAS date variables become DBF date fields. When converting from a SAS data set to a dBASE II file, SAS date variables become dBASE II character fields in the form *YYYMMDD*.

Transferring Other Software Files to DBF Files You might find it helpful to save another software vendor's file to a DBF file and then convert that file into a SAS data set. UNIX users find this especially helpful. For example, you could save an Excel .XLS file to a DBF file (by selecting **File** —> **Save As** —>EMP.DBF from within an Excel spreadsheet) and then use PROC DBF to convert that file into a SAS data set. Or you could do the reverse: use PROC DBF to convert a SAS data set into a DBF file and then load that file into an Excel spreadsheet.

Examples

Example 1: Converting a dBASE II File to a SAS Data Set In this example, a dBASE II file named EMPLOYEE.DBF is converted to a SAS data set. Because no FILENAME statement is specified, the last level of the filename is assumed to be DBF and the file is assumed to be in your current directory and in uppercase.

```
libname save '/my/unx_save_dir';
proc dbf db2=employee out=save.employee;
run;
```

Example 2: Converting a SAS Data Set to a dBASE 5 File In this example, a SAS data set is converted to a dBASE 5 file. A FILENAME statement specifies a fileref that names the dBASE 5 file. You must specify the FILENAME statement before the PROC DBF statement.

```
libname mylib '/my/unix_directory';
filename employee '/sasdemo/employee.dbf';
proc dbf db5=employee data=mylib.employee;
run;
```

In a Windows environment, this example would be:

```
libname mylib 'c:\my\directory';
filename employee 'c:\sasdemo\employee.dbf';
proc dbf db5=employee data=mylib.employee;
run;
```

In an OS/390 environment, this example would be:

```
libname mylib 'sasdemo.employee.data';
filename dbfout 'sasdemo.newemp.dbf' recfm=n;
proc dbf db5=dbfout data=mylib.employee;
run;
```

PROC DIF

converts a DIF file to SAS data set or a SAS data set to a DIF file

Syntax

PROC DIF *options*;

PROC DIF Options

DIF=*fileref* | *filename*
specifies the fileref or filename of a DIF file. When you use the FILENAME statement to assign the fileref, the statement must specify the filename plus a .DIF extension (that is, **filename myref '/my_dir/myfile.dif'**).

If you specify a filename instead of a fileref, you can only specify the name itself (omitting the .DIF extension) and the file must be in the current directory. For example, this PROC DIF statement creates the EMP.DIF file from the MYLIB.EMPLOYEE data set:

```
proc dif dif=emp data=mylib.employee;
```

You *cannot* specify **emp.dif** or a full pathname (**proc dif dif='/my/unix_directory/emp.dif'**) in the DIF option.

DATA=<*libref.*>*member*
names the input SAS data set. Use this option if you are creating a DIF file from a SAS data set. If you use this option, do not use the OUT= option. If you omit the DATA= option, the SAS System creates an output SAS data set from the DIF file.

OUT=<*libref.*>*member*
names the SAS data set to hold the converted data. Use this option only if you omit the DATA= option.

If OUT= is omitted, SAS creates a temporary data set in the WORK library. (Under UNIX, the temporary data set is named DATA1 [...DATA*n*]; under PCs, it is called _DATA_. If OUT= is omitted or if you do not specify a two-level name in the OUT= option, the SAS data set created by PROC DIF remains available during your current SAS session but is not permanently saved.

LABELS
causes PROC DIF to write the names of the SAS variables as the first row of the DIF file and a row of blanks as the second row of the DIF file. The actual data portion of the DIF file begins in the third row. The LABELS option is allowed only when converting a SAS data set to a DIF file.

PREFIX=*name*
specifies a prefix to be used in constructing SAS variable names when converting a DIF file to a SAS data set. For example, if PREFIX=VAR, the new variable names are VAR1, VAR2, ..., VAR*n*. If you omit the PREFIX= option, PROC DIF assigns the names COL1, COL2, ..., COL*n* to the variables in the output SAS data set.

SKIP=*n*
specifies the number of rows, beginning at the top of the DIF file, to be ignored when converting a DIF file to a SAS data set. For example, suppose the first row of your

DIF file contains column headings and the second row of your DIF file is a blank row. The actual data in your DIF file begin in row 3. You should specify SKIP=2 so that PROC DIF ignores the nondata portion of your DIF file. Alternatively, you could delete the first two rows of your DIF file before using PROC DIF.

Details The DIF procedure converts Data Interchange Format (DIF) files to SAS data sets that are compatible with the current release of SAS software. This procedure can also be used to convert SAS data sets to DIF files.

PROC DIF produces one output file but no printed output. The output file contains the same information as the input file but in a different format.

Software Arts, Inc. developed the Data Interchange Format to be used as a common language for data. Originally, DIF was made popular by products such as Lotus 1-2-3 and VisiCalc. Although DIF is not as popular today as it once was, it is still supported by many software products.

Note: Any DIF file that you plan to convert to a SAS data set should be in a tabular form. All items in a given column should represent the *same* type of data. If any rows in the DIF file contain inconsistent data–for example, a row of underscores, dashes, or blanks–delete these rows before converting the DIF file to a SAS data set. It is recommended that you make a backup copy of your DIF table before you make these modifications. △

When converting from a DIF file to a SAS data set, each row of the DIF file becomes an observation in the SAS data set. Conversely, when converting a SAS data set to a DIF file, each SAS observation becomes a row in the DIF file. To use the DIF procedure, you must have a SAS/ACCESS interface to PC File Formats license.

Converting DIF Variables to SAS Variables Character variables in a DIF file (sometimes referred to as *string values*) become SAS character variables of length 20. If a DIF character variable's value is longer than 20 characters, it is truncated to a length of 20 in the SAS output data set. The quotation marks that normally enclose character variable values in a DIF file are removed when the value is converted to a SAS character value.

Numeric variables, which can be represented in either integer or scientific notation in a DIF file, become SAS numeric variables when a DIF file is converted to a SAS data set.

Transferring SAS Data Sets to and from Other Software Products Using DIF DIF files are not generally used as the native file format for a software product's data storage. Therefore, transferring data between SAS and another software product is a two-step process when using DIF files.

To send SAS data sets to another software product using DIF files, you must first run PROC DIF to convert your SAS data set to a DIF file. Use whatever facility is provided by the target software product to read the DIF file. For example, you use the Lotus 1-2-3 Translate Utility to translate a DIF file to a 1-2-3 worksheet file. (This facility might be provided by an import tool or from a **File** —> **Open** dialog box in that software product.) After the application reads the DIF file data, the data can be manipulated and saved in the application's native format.

To transfer data in the opposite direction—for example, from a software product to a SAS data set—the process is reversed. First, export the data to a DIF file and then run PROC DIF to read the DIF file into a SAS data set.

Missing Values The developers of the Data Interchange Format (DIF) suggest that you treat all numeric values that have a value indicator other than V as missing values. PROC DIF follows this convention. When a DIF file is converted to a SAS data set, any numeric value with a value indicator other than V becomes a SAS missing value.

When a SAS data set that has missing values for some numeric variables is converted to a DIF file, the following assignments are made in the DIF file for the variables with missing values:

- the type indicator field value is set to **0**
- the number field value contains a string of 16 blanks
- the value indicator is set to **NA**.

Examples

Example 1: Converting a DIF File to a SAS Data Set In this example, a DIF file named EMPLOYEE.DIF is converted to a SAS data set. Because no FILENAME statement is specified, the last level of the filename is assumed to be DIF, and the file is assumed to be in your current directory and in uppercase.

```
libname save '/my/my_unx_dir';
proc dif dif=employee out=save.employee;
run;
```

Example 2: Converting a SAS Data Set to a DIF File In this example, a SAS data set is converted to a DIF file. A FILENAME statement is used to specify a fileref that names the DIF file. You must specify the FILENAME statement before the PROC DIF statement.

```
filename employee 'c:\sasdemo\employee.dif';
proc dif dif=employee data=save.employee;
run;
```

Or, in a UNIX environment, this example would be:

```
filename employee '/sasdemo/employee.dif';
proc dif dif=employee data=save.employee;
run;
```

See Also

"Programmer's Guide to the DIF," *Software Arts Technical Notes* (SATN-18).

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

Copyright © 1999 by SAS Institute Inc., Cary, NC, USA.

ISBN 1-58025–544–2

All rights reserved. Produced in the United States of America. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

U.S. Government Restricted Rights Notice. Use, duplication, or disclosure of the software and related documentation by the U.S. government is subject to the Agreement with SAS Institute and the restrictions set forth in FAR 52.227–19 Commercial Computer Software-Restricted Rights (June 1987).

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

1st printing, October 1999

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries.® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.

The Institute is a private company devoted to the support and further development of its software and related services.