CHAPTER 18

The EXPORT Procedure

Overview

The EXPORT procedure reads data from a SAS data set and writes it to an external data source. External data sources can include DBMS tables, PC files, spreadsheets, and delimited external files (which are files that contain columns of data values that are separated by a delimiter such as a blank or a comma). For more information about PC file formats, see *SAS/ACCESS Software for PC File Formats: Reference*.

Once you invoke PROC EXPORT, the procedure reads the input data set and writes the data to the external data source. PROC EXPORT exports the data by one of the following methods:

- generated DATA step code
- generated SAS/ACCESS code
- translation engines.

You control the results with options and statements that are specific to the output data source. PROC EXPORT produces the specified output file and writes information about the export to the SAS log. In the log, you see the DATA step or the SAS/ACCESS code generated by PROC EXPORT. If a translation engine is used, then no code is submitted.

PROC EXPORT is available on the following hosts:

- OS/2
- UNIX
- OpenVMS
- Windows 95
- Windows NT
- Windows 98

Procedure Syntax

PROC EXPORT DATA=*SAS-data-set*

OUTFILE=“*filename*” | OUTTABLE=“*table-name*”
<DBMS=*identifier*> <REPLACE>;

<*data-source-statements*;>

PROC EXPORT Statement

Featured in: All examples

PROC EXPORT DATA=*SAS-data-set*
OUTFILE=*filename* | OUTTABLE=*table-name*
<DBMS=*identifier*> <REPLACE>;

Required Arguments

DATA=*SAS-data-set*
identifies the input SAS data set with either a one- or two-level SAS name (library and member name). If you specify a one-level name, PROC EXPORT assumes the WORK library.

Featured in: All examples

OUTFILE=*“filename”*
specifies the complete path and filename of the output PC file, spreadsheet, or delimited external file. If the name does not include special characters (like the backslash in a path), lowercase characters, or spaces, you can omit the quotes.

Featured in Example 1 on page 427 and Example 2 on page 430

OUTTABLE=*“tablename”*
specifies the table name of the output DBMS table. If the name does not include special characters (like question marks), lowercase characters, or spaces, you can omit the quotes. Note that the DBMS table name may be case-sensitive.

Requirement: When you export a DBMS table, you must specify the DBMS= option.

Featured in: Example 3 on page 431

Options

DBMS=*identifier*
specifies the type of data to export. For example, DBMS=DBF specifies to export a dBASE file. For PC files, spreadsheets, and delimited external files, you do not have to specify DBMS= if the filename specified with OUTFILE= contains a valid extension so that PROC EXPORT can recognize the type of data. For example, PROC EXPORT recognizes the filename ACCOUNTS.WK1 as a Lotus 1 spreadsheet and the filename MYDATA.CSV as a delimited external file that contains comma-separated data values. When you specify DBMS=EXCEL, PROC EXPORT assumes that the output data source is an Excel Version 5 spreadsheet. Therefore, to

export to an Excel 4 spreadsheet, specify DBMS=EXCEL4, and to export to an Excel 97 spreadsheet, specify DBMS=EXCEL97.

To export a DBMS table, you must specify DBMS= using a valid database product. For example, DBMS=ACCESS exports a Microsoft Access table.

If you do not specify an identifier or if the extension of the filename is not recognized, an error is returned.

The DBMS= specification can include the values listed in the following table:

Identifier	Output Data Source	Extension
ACCESS	Microsoft Access database	.MDB
DBF	dBASE files	.DBF
WK1	Lotus 1 spreadsheet	.WK1
WK3	Lotus 3 spreadsheet	.WK3
WK4	Lotus 4 spreadsheet	.WK4
EXCEL	Excel Version 5 spreadsheet	.XLS
EXCEL4	Excel Version 4 spreadsheet	.XLS
EXCEL5	Excel Version 5 spreadsheet	.XLS
EXCEL97	Excel 97 spreadsheet	.XLS
DLM	delimited file (default delimiter is a blank)	.*
CSV	delimited file (comma-separated values)	.CSV
TAB	delimited file (tab-delimited values)	.TXT

Restriction: The data sources available to you depend on which SAS/ACCESS products you have licensed. If you do not have any SAS/ACCESS products licensed, then the only type of data source files available to you are .CSV, .TXT, and delimited files.

Restriction: The OS/2 operating environment does not support Excel 5 and Excel 97 spreadsheets.

Comparison: When you specify DBMS=EXCEL

- □ PROC EXPORT identifies the output data source only as an Excel Version 5 spreadsheet.
- □ PROC IMPORT distinguishes between Excel Version 4 and Version 5 spreadsheets.

Featured in: Example 1 on page 427 and Example 3 on page 431

Note: If you export a SAS data set that contains missing values (for either character or numeric values) to a dBASE file (DBF), the missing values are translated to blanks. △

REPLACE

overwrites an existing file. If you do not specify REPLACE, PROC EXPORT does not overwrite an existing file.

Featured in: Example 3 on page 431

Data Source Statements

Featured in: Example 1 on page 427 and Example 3 on page 431

PROC EXPORT provides a variety of statements that are specific to the output data source.

Statement for Delimited Files

The following statement is available to export delimited external files.

DELIMITER=*'char'* | *'nn'x*

specifies the delimiter to separate columns of data in the output file. You can specify the delimiter as a single character or as a hexidecimal value. For example, if you want columns of data to be separated by an ampersand, specify DELIMITER='&'. If you do not specify DELIMITER=, PROC EXPORT assumes that the delimiter is a blank. You can replace the equals sign with a blank.

Interaction: You do not have to specify DELIMITER= if you specify DBMS=CSV or DBMS=TAB or if the output filename has an extension of .CSV or .TXT.

Featured in: Example 1 on page 427

Statements for DBMS Tables

The following statements are available to establish a connection to the DBMS when you export a DBMS table.

DATABASE=*"database"*

specifies the complete path and filename of the database to contain the specified DBMS table. If the database name does not contain lowercase characters, special characters, or national characters, you can omit the quotes. You may replace the equals sign with a blank.

Note: A default may be configured in the DBMS client software; the SAS System does not generate a default value. △

Featured in: Example 3 on page 431

DBPWD=*"database password"*

specifies a password that allows access to a database. You may replace the equals sign with a blank.

Interaction: DBPWD= cannot be used with PWD=.

PWD=*"password"*

specifies the user password used by the DBMS to validate a specific userid. If the password does not contain lowercase characters, special characters, or national characters, you can omit the quotes. You may replace the equals sign with a blank.

Note: The DBMS client software may default to the userid and password that was used to log in to the operating environment; the SAS System does not generate a default value. △

Interaction: PWD= cannot be used with DBPWD=.

UID=*"password"*

identifies the user to the DBMS. If the userid does not contain lowercase characters, special characters, or national characters, you can omit the quotes. You can replace the equals sign with a blank.

Note: The DBMS client software may default to the userid and password that was used to log in to the operating environment; the SAS System does not generate a default value. △

WGDB=*"workgroup-database-name"*
specifies the workgroup (security) database name that contains the USERID and PWD data for the DBMS. If the workgroup database name does not contain lowercase characters, special characters, or national characters, you can omit the quotes. You may replace the equals sign with a blank.

Note: A default workgroup database may be used by the DBMS; the SAS System does not generate a default value. △

Security Levels for Microsoft Access Tables

Microsoft Access tables have three levels of security, for which specific combinations of security statements must be used.

None
Do not specify DBPWD=, PWD=, UID=, or WGDB=.

Password
Specify only DBPWD=.

User-level
Specify only PWD=, UID=, and WGDB=.

Each statement has a default value; however, you may find it necessary to provide a value for each statement explicitly.

Examples

Example 1: Exporting a Delimited External File

Procedure features:

PROC EXPORT statement arguments:

DATA=
DBMS=
OUTFILE=

Data source statement:

DELIMITER=

This example exports the following SAS data set named MYFILES.CLASS and creates a delimited external file:

```
                                The SAS System                    1

 Obs    Name       Sex    Age    Height    Weight

   1    Alice       F      13     56.5       84.0
   2    Becka       F      13     65.3       98.0
   3    Gail        F      14     64.3       90.0
   4    Karen       F      12     56.3       77.0
   5    Kathy       F      12     59.8       84.5
```

```
     6    Mary        F      15     66.5      112.0
     7    Sandy       F      11     51.3       50.5
     8    Sharon      F      15     62.5      112.5
     9    Tammy       F      14     62.8      102.5
    10    Alfred      M      14     69.0      112.5
    11    Duke        M      14     63.5      102.5
    12    Guido       M      15     67.0      133.0
    13    James       M      12     57.3       83.0
    14    Jeffrey     M      13     62.5       84.0
    15    John        M      12     59.0       99.5
    16    Philip      M      16     72.0      150.0
    17    Robert      M      12     64.8      128.0
    18    Thomas      M      11     57.5       85.0
    19    William     M      15     66.5       112
```

Program

DATA= identifies the input SAS data set.

```
proc export data=myfiles.class
```

OUTFILE= specifies the output filename. The filename does not contain an extension.

```
            outfile="/myfiles/class"
```

DBMS= specifies that the output file is a delimited external file.

```
            dbms=dlm;
```

DELIMITER= specifies that the delimiter to separate columns of data in the output file is the ampersand.

```
   delimiter=’&’;
run;
```

SAS Log

The SAS log displays the following information about the successful export. Notice the generated SAS DATA step.

```
12    /**********************************************************************
   12 ! *
   13    *   PRODUCT:   SAS
   14    *   VERSION:   8.00
   15    *   CREATOR:   External File Interface - Version 1.1
   16    *   DATE:      01MAR1999
   17    *   DESC:      Generated SAS Datastep Code
   18    *   TEMPLATE SOURCE:  (None Specified.)
   19    **********************************************************************
   19 ! */
   20       data _null_;
   21       set  MYFILES.CLASS                                   end=EFIEOD;
   22       %let _EFIERR_ = 0;     /* clear ERROR detection macro variable */
   23       %let _EFIREC_ = 0;     /* clear export record count macro variable
   23 ! */
   24       file '/myfiles/class' delimiter='&' DSD DROPOVER
   24 ! lrecl=32767;
   25          format NAME $8. ;
   26          format SEX $1. ;
   27          format AGE best12. ;
   28          format HEIGHT best12. ;
   29          format WEIGHT best12. ;
   30       if _n_ = 1 then        /* write column names */
   31        do;
   32          put
   33          'NAME'
   34          '&'
   35          'SEX'
   36          '&'
   37          'AGE'
   38          '&'
   39          'HEIGHT'
   40          '&'
   41          'WEIGHT'
   42          ;
   43        end;
   44        do;
   45          EFIOUT + 1;
   46          put NAME $ @;
   47          put SEX $ @;
   48          put AGE @;
   49          put HEIGHT @;
   50          put WEIGHT ;
   51          ;
   52        end;
   53       If _ERROR_ then        /* ERROR detection */
   54          call symput('_EFIERR_',1);
   55       If EFIEOD then
   56          call symput('_EFIREC_',EFIOUT);
   57       run;

   NOTE: Numeric values have been converted to character
         values at the places given by: (Line):(Column).
         53:31   55:31

   NOTE: 20 records were written to the file '/myfiles/class'.
         The minimum record length was 17.
         The maximum record length was 26.

   19 records created in /myfiles/class from MYFILES.CLASS.
   NOTE: /myfiles/class was successfully created.
```

Output

The external file produced by PROC EXPORT follows.

```
NAME&SEX&AGE&HEIGHT&WEIGHT
Alice&F&13&56.5&84
Becka&F&13&65.3&98
```

```
Gail&F&14&64.3&90
Karen&F&12&56.3&77
Kathy&F&12&59.8&84.5
Mary&F&15&66.5&112
Sandy&F&11&51.3&50.5
Sharon&F&15&62.5&112.5
Tammy&F&14&62.8&102.5
Alfred&M&14&69&112.5
Duke&M&14&63.5&102.5
Guido&M&15&67&133
James&M&12&57.3&83
Jeffrey&M&13&62.5&84
John&M&12&59&99.5
Philip&M&16&72&150
Robert&M&12&64.8&128
Thomas&M&11&57.5&85
William&M&15&66.5&112
```

Example 2: Exporting to an Excel Spreadsheet

Procedure features:

PROC EXPORT statement arguments:

DATA=
OUTFILE=

This example exports the SAS data set named SASUSER.ACCOUNTS, the first five observations of which follow, and creates an Excel 5 spreadsheet:

```
Product                        Quantity              Price

Dharamsala Tea                 10 boxes x 20 bags    16.00
Tibetan Barley Beer            24 - 12 oz bottles    19.00
Licorice Syrup                 12 - 550 ml bottles   10.00
Chef Anson's Cajun Seasoning   48 - 6 oz jars        22.00
Chef Anson's Gumbo Mix         36 boxes              21.35
```

Program

DATA= identifies the input SAS data set.

```
proc export data=sasuser.accounts
```

OUTFILE= specifies the output file. The filename contains the extension .XLS, which PROC EXPORT recognizes, by default, as the extension for an Excel 5 spreadsheet.

```
            outfile="C:\myfiles\Prices.xls";
      run;
```

Example 3: Exporting a Microsoft Access Table

Procedure features:

PROC EXPORT statement arguments:

DATA=
DBMS=
OUTTABLE=
REPLACE

Data Source Statement:

DATABASE=

This example exports a SAS data set named SASUSER.CUST, the first five observations of which follow, and creates a Microsoft Access table. The security level for this Access table is none, so it is not necessary to specify any of the database security statements (DBPWD=, PWD=, UID=, or WGDB=).

```
Obs     Name                    Street                    Zipcode

 1      David Taylor            124 Oxbow Street          72511
 2      Theo Barnes             2412 McAllen Avenue       72513
 3      Lydia Stirog            12550 Overton Place       72516
 4      Anton Niroles           486 Gypsum Street         72511
 5      Cheryl Gaspar           36 E. Broadway            72515
```

Program

DATA= identifies the input SAS data set.

```
proc export data=sasuser.cust
```

OUTTABLE= specifies the output DBMS table name.

```
            outtable="customers"
```

DBMS= specifies that the output file is a Microsoft Access table.

```
            dbms=access
```

REPLACE specifies to overwrite the output file if it exists.

```
            replace;
```

DATABASE= specifies the path and filename of the database to contain the table.

```
   database="c:\myfiles\mydatabase.mdb";
run;
```

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS® Procedures Guide, Version 8,* Cary, NC: SAS Institute Inc., 1999. 1729 pp.

SAS® Procedures Guide, Version 8

ISBN 1–58025–482–9

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

1st printing, October 1999

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