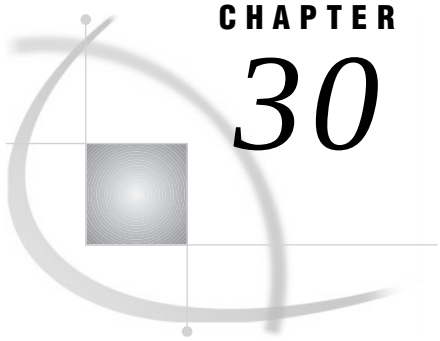


CHAPTER

30

Creating and Executing Stored Compiled DATA Step Programs

Definition

A *stored compiled DATA step program* is a SAS file that contains a DATA step program that has been compiled and then stored in a SAS data library. You can execute stored compiled programs as needed, without having to recompile them. Stored compiled DATA step programs are of member type PROGRAM.

Note: Stored compiled programs are available for DATA step applications only. Your stored programs can contain all SAS language elements except global statements. If you do include global statements in your source program, SAS stores the compiled program but not the global statements, and does not display a warning message in the SAS log. △

Uses for Stored Compiled DATA Step Programs

The primary use of stored compiled DATA step programs is for executing production jobs. The advantage of using these DATA step programs is that you can execute them as needed without investing the resources required for repeated compilation. The savings are especially significant if the DATA step contains many statements. If you install a new version of SAS, you do not need to recompile your source code.

Restrictions and Requirements

The following restrictions and requirements apply for using stored compiled DATA step programs:

- Stored compiled DATA step programs are available for DATA step applications only.
- Stored compiled DATA step program cannot contain global statements. If you do include global statements such as FILENAME, FOOTNOTE, LIBNAME, OPTIONS, and TITLE in your source program, SAS stores the compiled program but not the global statements. SAS does not display a warning message in the SAS log.
- SAS does not store raw data in the compiled program.

  Operating Environment Information: You cannot move a compiled program to an operating environment that has an incompatible machine architecture. You must, instead, recompile your source code and store your new compiled program.

  You can, however, move your compiled program to a different host machine that has a compatible architecture. △

How SAS Processes Stored Compiled DATA Step Programs

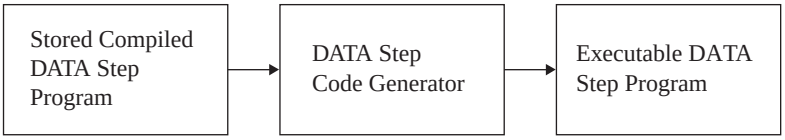
You first compile the SAS source program and store the compiled code. Then you execute the compiled code, redirecting the input and output as necessary.

SAS processes the DATA step through the compilation phase and then stores an intermediate code representation of the program and associated data tables in a SAS file. SAS processes the intermediate code when it executes the stored program. The following figure shows the process for creating a stored compiled DATA step program.

Figure 30.1 Creating a Stored Compiled Program

When SAS executes the stored program, it resolves the intermediate code produced by the compiler and generates the executable machine code for that operating environment. The following figure shows the process for executing a stored compiled DATA step program.

Figure 30.2 Executing a Stored Compiled Program



To move, copy, rename, or delete stored programs, use the DATASETS procedure or the utility windows in your windowing environment.

Creating a Stored Compiled DATA Step Program

Syntax

The syntax for creating a stored compiled DATA step program is as follows:

DATA *data-set-name(s)* / PGM=*stored-program-name*
<(<*password-option*><SOURCE=*source-option*>)>;

where

data-set-name
specifies a valid SAS name for the output data set created by the source program. The name can be a one-level name or a two-level name. You can specify more than one data set name in the DATA statement.

stored-program-name
specifies a valid SAS name for the SAS file containing the stored program. The name can be a one-level name, but it is usually a two-level name. Stored programs are assigned the member type PROGRAM in the SAS data library.

password-option
assigns a password to a stored compiled DATA step program.

source-option
allows you to save or encrypt the source code.

For complete information about the DATA statement, see *SAS Language Reference: Dictionary*.

Process

To compile and store a DATA step program, do the following:

1 Write, test, and debug the DATA step program you want to store.

If you are reading external raw data files or if you output raw data to an external file, use a fileref rather than the actual file name in your INFILE and FILE statements so that you can redirect your input and output when the stored program executes.

2 When the program runs correctly, submit it using the PGM= option in the DATA statement.

The PGM= option tells SAS to compile, but not execute, the program and to store the compiled code in the SAS file named in the option. SAS sends a message to the log when the program is stored.

Note: The default SOURCE=SAVE or SOURCE=ENCRYPT options automatically save your source code. △

Note: If you move your application to another operating environment, you need to recompile your source code and store your new compiled program. △

Example: Creating a Stored Compiled DATA Step Program

The following example uses the information in the input SAS data set IN.SAMPLE to assign a plant type based on a plant code. Note that the global LIBNAME statements are necessary to identify the storage location for your files, but are not part of STORED.SAMPLE, the DATA step that SAS stores.

```
libname in 'SAS-data-library';
libname stored 'SAS-data-library';


data out.sample / pgm=stored.sample;
   set in.sample;
   if code = 1 then
      do;
         Type='Perennial';
         number+4;
      end;
   else
   if code = 2 then
      do;
         Type='Annual';
         number+10;
      end;
   else
      do;
         Type='ERROR';
         Number=0;
      end;
run;
```

Output 30.1 Partial SAS Log Identifying the Stored DATA Step Program

```
.
.
.
NOTE: DATA STEP program saved on file STORED.SAMPLE.
NOTE: A stored DATA STEP program cannot run under a different operating system.
NOTE: DATA statement used:
      real time           1.14 seconds
      cpu time            0.12 seconds
```

Executing a Stored Compiled DATA Step Program

Syntax

The syntax for executing a stored compiled DATA step program, optionally retreiving source code, and optionally redirecting input or output, is as follows:

global SAS statements

DATA PGM=*stored-program-name* <(*password-option*)>;

<**DESCRIBE**;>
<**REDIRECT** INPUT | OUTPUT *old-name-1 = new-name-1*<*. . . old-name-n = new-name-n*>;>
<**EXECUTE**;>

where

global SAS statements
specifies any global SAS statements that are needed by the program when it executes, such as a FILENAME or a LIBNAME statement that points to input files or routes output.

stored-program-name
specifies a valid SAS name for the SAS file containing the stored program. The name can be a one-level name or a two-level name.

password-option
specifies a password that you use to access the stored compiled DATA step program.

DESCRIBE
is a SAS statement that retrieves source code from a stored compiled DATA step program or a DATA step view.

INPUT | OUTPUT
specifies whether you are redirecting input or output data sets. When you specify INPUT, the REDIRECT statement associates the name of the input data set in the source program with the name of another SAS data set. When you specify OUTPUT, the REDIRECT statement associates the name of the output data set with the name of another SAS data set.

old-name
specifies the name of the input or output data set in the source program.

new-name
specifies the name of the input or output data set that you want SAS to process for the current execution.

EXECUTE
is a SAS statement that executes a stored compiled DATA step program.

For complete information about the DATA statement, see *SAS Language Reference: Dictionary*.

Process

To execute a stored compiled DATA step program, follow these steps:

1 Write a DATA step for each execution of the stored program. In this DATA step, specify the name of the stored program in the PGM= option of the DATA statement and include an optional password. You can
   - □ submit this DATA step as a separate program
   - □ include it as part of a larger SAS program that can include other DATA and procedure (PROC) steps
   - □ point to different input and output SAS data sets each time you execute the stored program by using the REDIRECT statement.

2 Submit the DATA steps. Be sure to end each one with a RUN statement or other step boundary.

Using Global Statements

You can use global SAS statements such as FILENAME or LIBNAME when you store or execute a stored compiled DATA step program. However, the global statements that you use to compile and store a DATA step program are not stored with the DATA step code.

Redirecting Output

You can redirect external files using filerefs. You can use the REDIRECT statement for renaming input and output SAS data sets.

You can use the REDIRECT statement to redirect input and output data to data sets you specify. Note that the REDIRECT statement is available only for use with stored compiled DATA step programs.

Note: To redirect input and output stored in external files, include a FILENAME statement at execution time to associate the fileref in the source program with different external files. △

CAUTION:

Use caution when you redirect input data sets. The number and attributes of variables in the input SAS data sets that you read with the REDIRECT statement should match those of the input data sets in the SET, MERGE, MODIFY, or UPDATE statements of the source code. If they do not match, the following occurs:

- If the variable length attributes differ, the length of the variable in the source code data set determines the length of the variable in the redirected data set.
- If extra variables are present in the redirected data sets, the stored program will continue to execute but the results of your program may not be what you expect.
- If the variable type attributes are different, the stored program stops processing and an error message is sent to the SAS log.

△

Printing the Source Code of a Stored Compiled DATA Step Program

If you use both the DESCRIBE and the EXECUTE statements when you execute a stored compiled DATA step program, SAS writes the source code to the log. The following example executes a stored compiled DATA step program. The DESCRIBE statement in the program writes the source code to the SAS log.

```
data pgm=stored.sample;
   describe;
   execute;
run;
```

Output 30.2 Partial SAS Log Showing the Source Code Generated by the DESCRIBE Statement

```
.
.
.
26
27   data pgm=stored.sample;
28      describe;
29      execute;
30   run;
NOTE: DATA step stored program STORED.SAMPLE is defined as:

data out.sample / pgm=stored.sample;
   set in.sample;
   if code = 1 then
      do;
         Type='Perennial';
         number+4;
      end;
   else
      if code = 2 then
         do;
            Type='Annual';
            number+10;
         end;
      else
         do;
            Type='ERROR';
            Number=0;
         end;
run;


NOTE: DATA STEP program loaded from file STORED.SAMPLE.
NOTE: There were 7 observations read from the dataset IN.SAMPLE.
NOTE: The data set OUT.SAMPLE has 7 observations and 4 variables.
NOTE: DATA statement used:
      real time           0.80 seconds
      cpu time            0.15 seconds
```

For more information about the DESCRIBE statement, see *SAS Language Reference: Dictionary*.

Example: Executing a Stored Compiled DATA Step Program

The following DATA step executes the stored program STORED.SAMPLE created in "Example: Creating a Stored Compiled DATA Step Program" on page 468. The REDIRECT statement specifies the source of the input data as BASE.SAMPLE. The output from this execution of the program is redirected and stored in a data set named TOTALS.SAMPLE. Output 30.3 on page 472 shows part of the SAS log.

```
libname in 'SAS-data-library';
libname base 'SAS-data-library';
libname totals 'SAS-data-library';
libname stored 'SAS-data-library';


data pgm=stored.sample;
   redirect input in.sample=base.sample;
```

```
   redirect output out.sample=totals.sample;
run;
```

Output 30.3 Partial SAS Log Identifying the Redirected Output File

```
      cpu time            0.00 seconds
.
.
.
6
7   data pgm=stored.sample;
8      redirect input in.sample=base.sample;
9      redirect output out.sample=totals.sample;
10   run;
NOTE: DATA STEP program loaded from file STORED.SAMPLE.
NOTE: The data set TOTALS.SAMPLE has 7 observations and 4 variables.
NOTE: DATA statement used:
      real time           0.67 seconds
```

Differences between Stored Compiled DATA Step Programs and DATA Step Views

Stored compiled DATA step programs and DATA step views are similar in function. They both store DATA step programs that can retrieve and process data stored in other files. Both have the same restrictions and requirements (see "Restrictions and Requirements" on page 466). For information about DATA step views, see Chapter 29, "SAS Data Views," on page 455.

Stored compiled DATA step programs and DATA step views differ in the following ways:

- A stored compiled DATA step program is explicitly executed when it is specified by the PGM= option on a DATA statement. The stored compiled DATA step is used primarily in production jobs.
- A DATA step view is implicitly executed when the view is referenced as an input data set by another DATA or procedure (PROC) step. Its main purpose is to provide data one record at a time to the invoking procedure or DATA step.
- You can use the REDIRECT statement when you execute a stored compiled DATA step. You can not use this statement with DATA step views.

Examples

Example 1: Quality Control Application

This example illustrates how to use a stored compiled DATA step program for a simple quality control application. This application processes several raw data files. The source program uses the fileref DAILY in the INFILE statement. Each DATA step that is used to execute the stored program can include a FILENAME statement to associate the fileref DAILY with a different external file.

The following statements compile and store the program:

```
libname stored 'SAS-data-library-1';

data flaws / pgm=stored.flaws;
   length Station $ 15;
   infile daily;
   input Station $ Shift $ Employee $ NumberOfFlaws;
   TotalNumber + NumberOfFlaws;
run;
```

The following statements execute the stored compiled program, redirect the output, and print the results:

```
libname stored 'SAS-data-library-1';
libname testlib 'SAS-data-library-2';

data pgm=stored.flaws;
   redirect output flaws=testlib.daily;
run;

proc print data=testlib.daily;
   title 'Quality Control Report';
run;
```

Output 30.4 Quality Control Application Output

Quality Control Report 1

Obs	Station	Shift	Employee	Number OfFlaws	Total Number
1	Cambridge	1	Lin	3	3
2	Northampton	1	Kay	0	3
3	Springfiled	2	Sam	9	12

Note that you can use the TITLE statement when you execute a stored compiled DATA step program or when you print the results.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS Language Reference: Concepts*, Cary, NC: SAS Institute Inc., 1999. 554 pages.

SAS Language Reference: Concepts

ISBN 1–58025–441–1

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

1st printing, November 1999