Using PROC DATASETS for Efficient SAS Processing

Ken Friedman
L. L. Bean, Inc.

Hind's Law:
Any given program will expand to fill all available memory.

Small's Second Law of Programming:
You can write a bad program in any language.

PROC DATASETS, an Overview

The DATASETS procedure is used to manage SAS datasets. With this procedure, you can list, change, append, and repair datasets and create and maintain indexes. Incorporated within the DATASETS procedure are all of the capabilities of the APPEND, CONTENTS, and COPY procedures. Procedure commands execute with a RUN command or upon the entry of a DATASETS command. The procedure remains active until another procedure, dataset statement, or QUIT command is executed.

Unless specified, all DATASETS commands work on datasets found in the temporary work library. To affect a different library, use the LIBRARY option to specify a library name.

```
LIBNAME input 'SAS-data-library';

PROC DATASETS LIBRARY=input;
   DATASETS commands
RUN;
```

Scope of this Paper

The purpose of this paper is to introduce new users to the DATASETS procedure. Advanced topics such as working with indexes, generational datasets, and repairing damaged datasets are not included in this paper.

APPEND vs. SET

The APPEND function is used to append observations from one dataset to another. Unlike the SET command which reads in all observations from the datasets being concatenated, the APPEND function only reads in the observations from the dataset being appended. By using the APPEND function over the SET command you will save processing time and work space.

To use the APPEND command, you must specify the dataset you want to append **to** with the OUT= or BASE= statements. To specify the name of the dataset containing the records you want to append **from** use the DATA= or NEW= statement. (While not required, this makes good coding sense. If you do not specify the dataset to append from, the last dataset created is used.)

If the two datasets you are concatenating do not contain the same variable names, types, or lengths, you can use the FORCE option to force the append to take place.

The following examples both do the same thing, albeit the DATASETS procedure does it more efficiently (note the use of the WHERE statement to limit observations for appending.)

```
PROC DATASETS;
   APPEND OUT  = membr_a
            DATA = membr_b (WHERE=(year=2000));
   RUN;

   DATA membr_a;
      SET          membr_a  membr_b
(WHERE=(year=2000));
```

CHANGE

Used to rename one or more members within a SAS library. To use the CHANGE command, specify the old name on the left side of the equals sign and the new name on the right.

The following example renames the dataset temp1 to new_a and temp2 to new_b:

```
PROC DATASETS;
   CHANGE temp1=new_a  temp2=new_b;
   RUN;
```

COPY

Used to copy or move a SAS member from one library to another.

To limit copying to specific members, use either the SELECT or EXCLUDE options. To specify a different library to copy from use either the DATASETS LIBRARY option to specify a default library or use the IN= option.

To move a member from one library to another and then delete the original member, use the MOVE option

The following example moves two members from lib1 to lib2:

```
LIBNAME lib1 'SAS-data-library';
LIBNAME lib2 'SAS-data-library';

PROC DATASETS;
   COPY IN=lib1 OUT=lib2  MOVE;
   SELECT member1 member2;
RUN;
```

SAVE, DELETE, and KILL

The SAVE command is used to specify which members should be kept within a SAS library with all others being deleted.

The following example shows how to save two members from within a permanent SAS library:

```
LIBNAME input 'SAS-data-library';

PROC DATASETS LIBRARY=input;
      SAVE member1 member2;
RUN;
```

The DELETE command and DATASETS procedure KILL option are used to remove members from a SAS library. Using the KILL option will remove all members from a SAS library upon execution. DELETE allows you to specify specific members to delete.

The following example shows how to delete all the members within a permanent SAS library using the KILL option:

```
LIBNAME input 'SAS-data-library';

PROC DATASETS LIBRARY=input KILL;
RUN;
```

MODIFY

Used to change specific dataset or variable attributes. This command allows you to specify formats, informats, and labels, rename variables, and create and delete indexes. The MODIFY command only works on one dataset at a time.

The following example modifies the dataset income, adding a dataset label, renaming the variable old to new, adding a label to this variable and setting a format for income:

```
LIBNAME input 'SAS-data-library';

PROC DATASETS LIBRARY=input;
   MODIFY income (LABEL='household income');
      RENAME old=new;
      LABEL new='originally called old';
      FORMAT income comma11.2;
RUN;
```

Note that in the above example that the original variable name old is renamed to new by specifying the original name on the left side of the equal sign and the new name on the right.

Because no observations are read in or written out during processing, for an existing dataset, the MODIFY command is the best way to add a label, rename a variable, or change a format or informat.

Another way to do the same thing as above with less efficiency is the following:

```
LIBNAME input 'SAS-data-library';

DATA input.income (LABEL='household income');
   SET input.income;
      RENAME old=new;
      LABEL new='originally called old';
      FORMAT income comma11.2;
RUN;
```

In the above example, all of the observations within input.income are read in and written out during processing. Depending on the size of the file being read, the amount of time and storage needed to make these modifications could be significant.

Conclusion

Until recently, I used a myriad of techniques for managing SAS datasets, most of them bad. Using the DATASETS procedure is a good way to manage SAS datasets. It's simple to use and in many cases more efficiently manipulates SAS datasets.

It is important to remember that the DATASETS procedure is an interactive procedure, all commands execute immediately and in the order that they appear. Consequently, caution should be taken when working with this procedure.

For more information on the DATASETS procedure, see the SAS Procedures Guide or check out the on-line SAS documentation.

Address

Questions pertaining to this article should be addressed to:

Ken Friedman
144 Pownal Rd
Freeport, Maine 04032
Kfriedman@llbean.com