Paper PS06_05

Renaming SAS® Variables

Imelda C. Go, South Carolina Department of Education, Columbia, SC

ABSTRACT

This paper discusses a number of ways to rename variables. Its topics include the RENAME statement used in DATA steps, the RENAME= data set option, the AS keyword for PROC SQL, using macros, and using the DATA _NULL_ step.

A "quick and dirty" way to change the name of a variable is shown below. The variable x is assigned to variable y and x is dropped from the data set. This is *not* really renaming a variable because it involves creating a copy of the variable to be renamed and the copy has the desired new name. This practice, although it achieves the desired result, is not optimal.

```
data two;
set one;
y=x;
drop x;
```

Changing a variable name is an inevitable fact of programming. There are several ways to accomplish this depending on the types of SAS statements involved. Consider the following SAS code.

```
data one;
input x z;
y=x+1;
cards;
1 2
;
```

The variables acquire their names through the INPUT statement (variables x and z in data set one), an assignment statement (variable y in data set one), or automatically in procedures when the user does not specify variable names. It is important to be aware of a PROC's features and how a PROC might accommodate variable name changes.

```
proc means data=one n mean;
  var x y;
  output out=stats;
```

The PROC MEANS results are:

```
     The MEANS Procedure
Variable    N            Mean
------------------------------
x           1       1.0000000
y           1       2.0000000
------------------------------
```

Data set stats is shown below. The n and mean values are listed under variables named after the corresponding analysis variable. The type of statistic in each observation is indicated by the _STAT_ variable.

```
Obs  _TYPE_  _FREQ_  _STAT_  x  y

 1      0       1     N      1  1
 2      0       1     MIN    1  2
 3      0       1     MAX    1  2
 4      0       1     MEAN   1  2
 5      0       1     STD    .  .
```

Suppose that the following code is used instead.

```
proc means data=one;
  var x y;
  output out=stats
         n=countx county
         mean=avex avey;
```

The results are shown below and even the data set structure differs from the previous one due to the user-specified options.

```
Obs _TYPE_ _FREQ_ countx county avex avey

 1     0      1      1      1     1    2
```

RENAME (DATA Step Statement)

A straightforward way of renaming a variable is to use the RENAME statement. The syntax for *n* variables is:

```
rename    oldvarname1=newvarname1
          oldvarname2=newvarname2
          …
          oldvarnamen=newvarnamen;
```

In the example below, the variable x is renamed to variable y and the variable z is renamed to variable a.

```
data two;
set one;
rename x=y z=a;
```

RENAME= (The Data Set Option)

Another way of renaming the variables is to use the RENAME option with the DATA option in a SET statement.

```
data two;
set one (rename=(x=y z=a));
```

Procedures provide default names for variables in output data sets.

```
proc freq data=one;
  tables x/out=xcounts;
```

By default, `count` is the variable name for the frequencies.

```
            The FREQ Procedure

                     Cumulative  Cumulative
x  Frequency Percent  Frequency    Percent
----------------------------------------
1         1  100.00          1    100.00


Obs    x    COUNT    PERCENT

 1     1      1        100
```

Suppose the default name is not satisfactory, one can do the following.

```
data xcounts;
set xcounts;
rename count=n;
```

However, the variable `count` can be renamed within PROC FREQ using the RENAME= data set option.

```
proc freq;
  tables x/out=xcounts (rename=(count=n));
```

PROC SQL

The AS keyword can be used in PROC SQL to change the variable name or assign a name to a computed value. In the example below, several variables are renamed and the difference of `group1.ss-group2.ss` is named `ssdiff`.

```
proc sql;
  select group1.grade, group1.name as name1,
         group2.name as name2, group1.ss as ss1,
         group2.ss as ss2, sex,
         group1.ss-group2.ss as ssdiff
    from group1, group2
   where group1.grade=group2.grade and
         group1.lunch=group2.lunch;
```

Sample results are shown below:

```
GRADE  NAME1            NAME2           SS1   SS2  SEX    SSDIFF
-----------------------------------------------------------------
    2  Taylor, Liz      Ryan, Meg       245   234  F          11
```

Macros Can Facilitate Renaming

One can also use a macro to rename variables that are part of a range, such as `x1-x100`.

```
%macro rename;
rename
   %do i=1 to 100;
       x&i=y&i
   %end;
;
%mend rename;
```

This macro writes the code to rename `x1-x100` to `y1-y100` respectively. However, the variable names are not always in such a convenient sequence.

Variable Names from PROC CONTENTS Output Data

If the variable names need to be renamed in a predictable manner, data from PROC CONTENTS and the DATA _NULL_ step can be used to create the RENAME statement.

Consider the following data set.

```
data three;
infile externalfile;
input x y z f a1-a3;
```

When PROC CONTENTS is applied to a data set and an output data set is created, the output data set will have the variable names of the data set the PROC was applied to. In the example below, data set `varnames` contains the variable names from data set `three`.

```
proc contents data=three out=varnames;
```

There are many other variables in data set `varnames`. The variable characteristics and labels are shown by the PROC CONTENTS output below.

```
      -----Alphabetic List of Variables and Attributes-----

 # Variable Type Len Pos Format      Label
-----------------------------------------------------------------
32 CHARSET  Char   8 802             Host Character Set
33 COLLATE  Char   8 810             Collating Sequence
28 COMPRESS Char   8 791             Compression Routine
20 CRDATE   Num    8  80 DATETIME16. Create Date
22 DELOBS   Num    8  96             Deleted Observations
                                     in Data Set
36 ENCRYPT  Char   8 824             Encryption Routine
19 ENGINE   Char   8 760             Engine Name
27 FLAGS    Char   3 788             Update Flags (Protect
```

```
                                     Contribute Add)
10 FORMAT   Char   8 744             Variable Format
12 FORMATD  Num    8  32             Number of Format Decimals
11 FORMATL  Num    8  24             Format Length
38 GENMAX   Num    8 128             Maximum Number
                                     of Generations
40 GENNEXT  Num    8 144             Next Generation Number
39 GENNUM   Num    8 136             Generation Number

25 IDXCOUNT Num    8 104             Number of Indexes
                                     for Data Set
23 IDXUSAGE Char   9 768             Use of Variable in Indexes
13 INFORMAT Char   8 752             Variable Informat
15 INFORMD  Num    8  48             Number of Informat Decimals
14 INFORML  Num    8  40             Informat Length
16 JUST     Num    8  56             Justification
 9 LABEL    Char 256 488             Variable Label
 7 LENGTH   Num    8   8             Variable Length
 1 LIBNAME  Char   8 152             Library Name
 3 MEMLABEL Char 256 192             Data Set Label
 2 MEMNAME  Char  32 160             Library Member Name
24 MEMTYPE  Char   8 777             Library Member Type
21 MODATE   Num    8  88 DATETIME16. Last Modified Date
 5 NAME     Char  32 456             Variable Name
18 NOBS     Num    8  72             Observations in Data Set
34 NODUPKEY Char   3 818             Sort Option: No
                                     Duplicate Keys
35 NODUPREC Char   3 821             Sort Option: No Duplicate
                                     Records
17 NPOS     Num    8  64             Position in Buffer
37 POINTOBS Char   3 832             Point to Observations
26 PROTECT  Char   3 785             Password Protection
                                     (Read Write Alter)
29 REUSE    Char   3 799             Reuse Space
30 SORTED   Num    8 112             Sorted and/or Validated
31 SORTEDBY Num    8 120             Position of Variable
                                     in Sortedby Clause
 6 TYPE     Num    8   0             Variable Type
 4 TYPEMEM  Char   8 448             Special Data Set
                                     Type (From TYPE=)
 8 VARNUM   Num    8  16             Variable Number
```

PROC PRINT output for data set `varnames` is shown below. The variable names are in a field called `name`.

```
           M                    I
   L    M  E T              F F N I I
   I    E  M Y      L V   F O O F N N        E
   B    M  L P      E A L O R R O F F        N
   N    N  A E N  T N R A R M M R O O J  N N G
O  A    A  B M A  Y G N B M A A M R R U  P O I
b  M    M  E E M  P T U E A T T A M M S  O B N
s  E    E  L M E  E H M L T L D T L D T  S S E

1 WORK ONE     a1 1 8 5     0 0   0 0 1 32 0 V8
2 WORK ONE     a2 1 8 6     0 0   0 0 1 40 0 V8
3 WORK ONE     f  1 8 4     0 0   0 0 1 24 0 V8
4 WORK ONE     x  1 8 1     0 0   0 0 1  0 0 V8
5 WORK ONE     y  1 8 2     0 0   0 0 1  8 0 V8
6 WORK ONE     z  1 8 3     0 0   0 0 1 16 0 V8

                                                I
                                                D       M
                 C                   M    D     X       E
                 R                   O    E     U       M
                 D                   D    L     S       T
O                A                   A    O     A       Y
b                T                   T    B     G       P
s                E                   E    S     E       E

1 22MAR03:14:25:48    22MAR03:14:25:48    0    NONE    DATA
2 22MAR03:14:25:48    22MAR03:14:25:48    0    NONE    DATA
3 22MAR03:14:25:48    22MAR03:14:25:48    0    NONE    DATA
4 22MAR03:14:25:48    22MAR03:14:25:48    0    NONE    DATA
5 22MAR03:14:25:48    22MAR03:14:25:48    0    NONE    DATA
6 22MAR03:14:25:48    22MAR03:14:25:48    0    NONE    DATA

  I            C          S        N   N        P
  D   P        O          O  C  C  O   O   E    O         G
  X   R        M       S  R  H  O  D   D   N    I   G  G  E
  C   O    F   P   R   O  T  A  L  U   U   C    N   E  E  N
  O   T    L   R   E   R  E  R  L  P   P   R    T   N  N  N
O U   E    A   E   U   T  D  S  A  K   R   Y    O   M  N  E
b N   C    G   S   S   E  B  E  T  E   E   P    B   A  U  X
s T   T    S   S   E   D  Y  T  E  Y   C   T    S   X  M  T

1 0  ---  ---  NO  NO  .  .        NO  NO  NO  YES  0  .  .
2 0  ---  ---  NO  NO  .  .        NO  NO  NO  YES  0  .  .
3 0  ---  ---  NO  NO  .  .        NO  NO  NO  YES  0  .  .
4 0  ---  ---  NO  NO  .  .        NO  NO  NO  YES  0  .  .
5 0  ---  ---  NO  NO  .  .        NO  NO  NO  YES  0  .  .
6 0  ---  ---  NO  NO  .  .        NO  NO  NO  YES  0  .  .
```

Using a DATA _NULL_ Step

Suppose there are pre-test data in a data set and the data set has exactly the same variable names as in data set `varnames` above. The task is to put the suffix of `pre` to all variable names for the pre-test data.

The results of the following DATA _NULL_ step are written in the SAS log.

```
data _null_;
set varnames end=eof;
if _n_=1 then put "rename ";
newvarname=trim(name)||'pre';
put name '= ' newvarname;
if eof then put ';';
```

The LOG will have the RENAME statement created by the code above.

```
466  data _null_;
467  set varnames end=eof;
468  if _n_=1 then put "rename ";
469  newvarname=trim(name)||'pre';
470  put name '= ' newvarname;
471  if eof then put ';';

rename
a1 = a1pre
a2 = a2pre
f = fpre
x = xpre
y = ypre
z = zpre
;
NOTE: There were 6 observations read from the dataset WORK.VARNAMES.
NOTE: DATA statement used:
      real time           0.05 seconds
```

%INFILE Statement

This RENAME statement can be copied from the SAS log into a SAS program. This would not be ideal because of the manual intervention. One may choose to write the results into an external file (`pretest.txt`). Later this external file can be referenced in the program using the %INFILE statement.

```
filename pre 'c:\example\pretest.txt';

data _null_;
file pre;
set varnames end=eof;
if _n_=1 then put "rename ";
newvarname=trim(name)||'pre';
if eof then put ';';

data pretest;
  set unrenamedpretestdata;
  %infile pre;
```

The previous example showed how to use the DATA _NULL_ step to create the SAS statements needed to rename variables. However, the example involved a simple renaming rule where a suffix is systematically added to each old variable name. If the renaming rules are not so simple but a pattern can be detected, then it is a programmable situation.

An alternative to programming the type of renaming described above is to use a data set with the old and new variable names. The data set is then matched by the old variable name with the data set obtained from PROC CONTENTS. After that, a DATA _NULL_ step, similar to the above, can be used to write a RENAME statement for renaming the variables.

REFERENCES

SAS Institute Inc., *SAS® Language Reference, Version 8*, Cary, NC: SAS Institute Inc., 1999. 1256 pp.

SAS Institute Inc., *SAS OnLineDoc®, Version 8*, Cary, NC: SAS Institute Inc., 1999.

TRADEMARK NOTICE

Imelda C. Go (icgo@juno.com)
South Carolina State Department of Education
Columbia, SC