(5) SELECTING AND RESTRICTING VARIABLES

Just as there are times when you will want to select or restrict the observations in a SAS data set, you may also want to select or restrict the variables. By default, all variables mentioned in a data step are added to the SAS data set being created. The default behavior can be altered by DROP and KEEP statements, or by DROP and KEEP data set options.

...Example 5.1...

```
data males females; ①
infile "k:\epi514\data\cancer99.txt";
input
@01 county $2.
@03 gender $1.
@04 age     3.
@07 cause  $4.
@11 place  $1.
;
if gender eq '1' then output males; ②
else                 output females;
drop gender; ③
run;
```

This is a copy of example 4.10. Two gender-specific data sets are created ① using OUTPUT statements to direct observations to the appropriate data set ②. Since the data sets are gender-specific, the variable gender is not neede in either data set, so it is dropped using a DROP statement ③. The DROP statement in this example is used when reading raw data, but it can also be used when reading a data set. First create a data set that contains data from all records in the raw data file. Then use another data step to create the gender-specific data sets.

...Example 5.2...

```
data cancer99; ①
infile "k:\epi514\data\cancer99.txt";
input
@01 county $2.
@03 gender $1.
@04 age     3.
@07 cause  $4.
@11 place  $1.
;
run;

data males females; ②
set cancer99;
if gender eq '1' then output males;
else                 output females;
drop gender; ③
run;
```

Data set CANCER99 contains observations for both males and females ①. A second data step reads that data set with SET statement and creates the two gender-specific data sets ②. The DROP statement removes the variable GENDER from both of the new data sets, not from CANCER99 ③.

There is also a DROP data set option. That is useful when different variables are to be dropped from various data sets. A DROP statement affects all data sets be created in a data step. The DROP data set option is data set-specific.

...Example 5.3...

```
data
males (drop=gender)  ①
females (drop=gender)
albany (drop=county) ②
;
set cancer99;
if gender eq '1' then output males;  ③
else                 output females;

if county eq '01' then output albany; ④
run;

title 'ORIGINAL DATA SET CANCER99'; ⑤
proc contents data=cancer99 short;
run;

title
'NEW DATA SET MALES-VARIABLES SELECTED WITH DROP';
proc contents data=males short;
run;

title
'NEW DATA SET FEMALES-VARIABLES SELECTED WITH DROP';
proc contents data=females short;
run;

title
'NEW DATA SET ALBANY-VARIABLES SELECTED WITH DROP';
proc contents data=albany short;
run;
```

```
ORIGINAL DATA SET CANCER99
Alphabetic List of Variables for WORK.CANCER99
age cause county gender place

DATA SET MALES
Alphabetic List of Variables for WORK.MALES
age cause county place

DATA SET FEMALES
Alphabetic List of Variables for WORK.FEMALES
age cause county place

DATA SET ALBANY
Alphabetic List of Variables for WORK.ALBANY
age cause gender place
```

Three data sets are created in one data step. The variable GENDER is not needed in data sets MALES and FEMALES ① while the variable COUNTY is not needed in the county-specific data set ALBANY ②. The DROP data set options are data set-specific. Different variables can be dropped from different data sets. If a DROP statement had been used instead ...

```
drop gender county;
```

the two variables would be dropped from all three data sets, not what you wanted. Output statement create data sets MALES and FEMALES ③ and data set ALBANY ④. PROC CONTENTS is run using the SHORT option to look at a list of variables in each data set ⑤. All variables remain in the original data set used in the SET statement, CANCER99. The DROP data set options had the desired results for the remaining three data set.

There are also KEEP statements and KEEP data set options. They both have the opposite effect of DROP, they tell SAS what variables you want in a data set as opposed to specifying the ones you do not want. Just like the DROP statement, a KEEP statement affects all data sets being created in a data step, while a KEEP data set option is specific to a data set.

...Example 5.4...

```
data males females albany; ①
set cancer99;
if gender eq '1' then output males;
else                 output females;
if county eq '01' then output albany;
keep age cause; ②
run;
```

Once again, you are creating three data sets ①. However, you only want two variables in the data sets, AGE and CAUSE. The KEEP statement tells SAS to place only two variables in each of the newly created data sets ②. All variables remain in data set CANCER99.

As with DROP, if you want place different variables in different data sets, you use a KEEP data set option, not a KEEP statement.

...Example 5.5...

```
data
males (keep=age cause)  ①
females (keep=age cause)
albany (keep=gender age cause) ②
;
set cancer99;
if gender eq '1' then output males;  ③
else                 output females;

if county eq '01' then output albany; ④
run;

title 'ORIGINAL DATA SET CANCER99'; ⑤
proc contents data=cancer99 short;
run;

title
'NEW DATA SET MALES-VARIABLES SELECTED WITH KEEP';
proc contents data=males short;
run;

title
'NEW DATA SET FEMALES-VARIABLES SELECTED WITH KEEP';
proc contents data=females short;
run;

title
'NEW DATA SET ALBANY-VARIABLES SELECTED WITH KEEP';
proc contents data=albany short;
run;
```

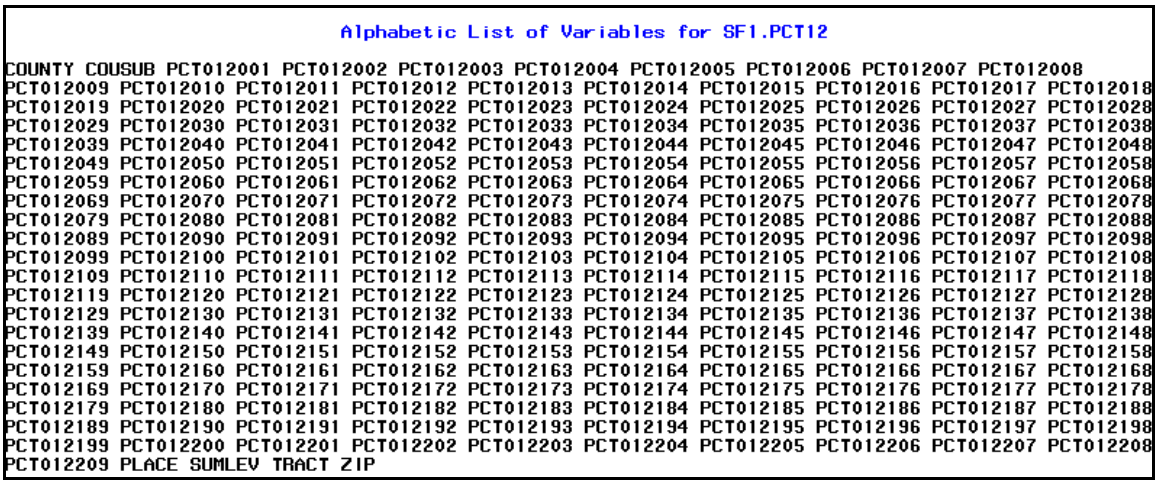
```
                ORIGINAL DATA SET CANCER99
        Alphabetic List of Variables for WORK.CANCER99
              age cause county gender place

          NEW DATA SET MALES - VARIABLES SELECTED WITH KEEP
              Alphabetic List of Variables for WORK.MALES
                          age cause

          NEW DATA SET FEMALES  - VARIABLES SELECTED WITH KEEP
              Alphabetic List of Variables for WORK.FEMALES
                          age cause

          NEW DATA SET ALBANY  - VARIABLES SELECTED WITH KEEP
              Alphabetic List of Variables for WORK.ALBANY
                       age cause gender
```

Three data sets are created in one data step. The variables AGE and CAUSE are kept in data sets MALES and FEMALES ①. Variables GENDER, AGE, and CAUSE are kept in data set ALBANY ②. The KEEP data set options are data set-specific. Different variables can be dropped from different data sets. If a KEEP statement had been used instead ...

```
keep gender age cause;
```

the three variables would be kept in all three data sets, not what you wanted. Output statement create data sets MALES and FEMALES ③ and data set ALBANY ④. PROC CONTENTS is run using the SHORT option to look at a list of variables in each data set ⑤. All variables remain in the original data set used in the SET statement, CANCER99. The KEEP data set options had the desired results for the remaining three data set.

You may now be asking when to use DROP and when to use KEEP. The answer is easy ... use the statement (or data set option) that requires the least amount of writing of SAS code get the desired result. For example, the output shown below lists all the variables in a data set that contains data from the 2000 census. To save you the counting, there are 215 variables is the data set.



Alphabetic List of Variables for SF1.PCT12

COUNTY COUSUB PCT012001 PCT012002 PCT012003 PCT012004 PCT012005 PCT012006 PCT012007 PCT012008
PCT012009 PCT012010 PCT012011 PCT012012 PCT012013 PCT012014 PCT012015 PCT012016 PCT012017 PCT012018
PCT012019 PCT012020 PCT012021 PCT012022 PCT012023 PCT012024 PCT012025 PCT012026 PCT012027 PCT012028
PCT012029 PCT012030 PCT012031 PCT012032 PCT012033 PCT012034 PCT012035 PCT012036 PCT012037 PCT012038
PCT012039 PCT012040 PCT012041 PCT012042 PCT012043 PCT012044 PCT012045 PCT012046 PCT012047 PCT012048
PCT012049 PCT012050 PCT012051 PCT012052 PCT012053 PCT012054 PCT012055 PCT012056 PCT012057 PCT012058
PCT012059 PCT012060 PCT012061 PCT012062 PCT012063 PCT012064 PCT012065 PCT012066 PCT012067 PCT012068
PCT012069 PCT012070 PCT012071 PCT012072 PCT012073 PCT012074 PCT012075 PCT012076 PCT012077 PCT012078
PCT012079 PCT012080 PCT012081 PCT012082 PCT012083 PCT012084 PCT012085 PCT012086 PCT012087 PCT012088
PCT012089 PCT012090 PCT012091 PCT012092 PCT012093 PCT012094 PCT012095 PCT012096 PCT012097 PCT012098
PCT012099 PCT012100 PCT012101 PCT012102 PCT012103 PCT012104 PCT012105 PCT012106 PCT012107 PCT012108
PCT012109 PCT012110 PCT012111 PCT012112 PCT012113 PCT012114 PCT012115 PCT012116 PCT012117 PCT012118
PCT012119 PCT012120 PCT012121 PCT012122 PCT012123 PCT012124 PCT012125 PCT012126 PCT012127 PCT012128
PCT012129 PCT012130 PCT012131 PCT012132 PCT012133 PCT012134 PCT012135 PCT012136 PCT012137 PCT012138
PCT012139 PCT012140 PCT012141 PCT012142 PCT012143 PCT012144 PCT012145 PCT012146 PCT012147 PCT012148
PCT012149 PCT012150 PCT012151 PCT012152 PCT012153 PCT012154 PCT012155 PCT012156 PCT012157 PCT012158
PCT012159 PCT012160 PCT012161 PCT012162 PCT012163 PCT012164 PCT012165 PCT012166 PCT012167 PCT012168
PCT012169 PCT012170 PCT012171 PCT012172 PCT012173 PCT012174 PCT012175 PCT012176 PCT012177 PCT012178
PCT012179 PCT012180 PCT012181 PCT012182 PCT012183 PCT012184 PCT012185 PCT012186 PCT012187 PCT012188
PCT012189 PCT012190 PCT012191 PCT012192 PCT012193 PCT012194 PCT012195 PCT012196 PCT012197 PCT012198
PCT012199 PCT012200 PCT012201 PCT012202 PCT012203 PCT012204 PCT012205 PCT012206 PCT012207 PCT012208
PCT012209 PLACE SUMLEV TRACT ZIP

Imagine that you want to create a new data set that contained only the variables COUNTY, PCT012001 (total population), PCT012002 (male population), and PCT012106 (female population). Would you write a KEEP statement and list the four variables you want, or a DROP statement and list the 211 variables you do not want? Hopefully you answered KEEP. However, in this and other similar situations where you select a small subset from a large number of variables, a KEEP statement is not the most efficient way to create the new data set.

...Example 5.6...

```
libname sf1 "k:\epi514\census";

data censusa;
set sf1.pct12;
keep county pct012001 pct012002 pct012106; ①
run;

data censusb (keep=county pct012001 pct012002 pct012106); ②
set sf1.pct12;
run;

data censusc;
set sf1.pct12 (keep=county pct012001 pct012002 pct012106); ③
run;
```

Each of the three data steps in example 5.6 produce a data set that contains all the observations, but only four of the variables from data set SF1.PCT12. The first data step uses a KEEP statement ①. The SET statement in that data step reads all the variables from each observation, then writes the values of the four kept variables to the new data set. The second data step uses a KEEP data set option on the new data set ②. Once again, the SET statement in that data step reads all the variables from each observation, then writes the values of the four kept variables to the new data set. The last data step uses a KEEP data set option, but this time it is used with the data set being read with the SET statement ③. Now, the SET statement only reads the values of four variables from the large data set, not 215. The third data step is the most efficient method since you only read and use the variables you want.

Using a KEEP data set option with the data set in a SET statement is not always possible even though it might appear that it is the best way to make a data set. In example 5.4, three new data sets were created that each contained only two variables from a larger data set. That situation appears to be perfect for using a KEEP data set option on the large data set in the SET statement ... why read all the variables when you only want two of them.

...Example 5.7...

```
data males females albany;
set cancer99;
if gender eq '1' then output males;
else                  output females;
if county eq '01' then output albany;
keep age cause; ①
run;

data males females albany;
set cancer99 (keep=age cause); ②
if gender eq '1' then output males;
else                  output females;
if county eq '01' then output albany;
run;
```

The first data step is a repeat of example 5.4. The KEEP statement results in there only being two variables in each of the new data sets ①. The second data step with a KEEP data set option on data set CANCER99 ② will run without any errors, but it does not produce the desired result. Here is the LOG showing what happens when each data step is run.

```
209  data males females albany;
210  set cancer99;
211  if gender eq '1' then output males;
212  else                  output females;
213  if county eq '01' then output albany;
214  keep age cause;
215  run;

NOTE: There were 37583 observations read from the data set WORK.CANCER99.
NOTE: The data set WORK.MALES has 18542 observations and 2 variables.
NOTE: The data set WORK.FEMALES has 19041 observations and 2 variables.
NOTE: The data set WORK.ALBANY has 724 observations and 2 variables.
NOTE: DATA statement used (Total process time):
      real time           0.03 seconds
      cpu time            0.03 seconds


216
217  data males females albany;
218  set cancer99 (keep=age cause);
219  if gender eq '1' then output males;
220  else                  output females;
221  if county eq '01' then output albany;
222  run;

NOTE: Variable gender is uninitialized.
NOTE: Variable county is uninitialized.
NOTE: There were 37583 observations read from the data set WORK.CANCER99.
NOTE: The data set WORK.MALES has 0 observations and 4 variables.
NOTE: The data set WORK.FEMALES has 37583 observations and 4 variables.
NOTE: The data set WORK.ALBANY has 0 observations and 4 variables.
NOTE: DATA statement used (Total process time):
      real time           0.04 seconds
      cpu time            0.04 seconds
```

The first data step places the correct number of observations in each data set. But, the second data step places no observations in data sets MALES and ALBANY, and all the observations from CANCER99 in data set FEMALES. Why? The lower portion of the LOG contains two notes about variables being uninitialized, GENDER and COUNTY. When you read the data with SET, you tell SAS to keep only AGE and CAUSE. Then you use the variables GENDER and COUNTY in IF statements. Since you did not read them, you cannot use them in the data step. The variable is not available so GENDER is never equal to '1' (no males). Since the IF statement says if GENDER is not '1', write the observation to data set FEMALES, all observations are written to that data set. The next IF statement is never true since the variable COUNTY cannot be used in the data step. It was never read and no observations are written to data set ALBANY.

What you should remember...

When you create data sets, only keep the variables you really need. That saves both space used to store your data and time to process your data ... examples: if you have a data set with observations for only males, you do not need the variable GENDER; if you have a data set with observations for only the year 2000, you do not need the variable YEAR.

DROP and KEEP statements affect all data sets created in a data step. DROP and KEEP data set options affect only the data set they with which they are used. Whether you use DROP or KEEP very often is just an issue of what list of variables is shorter. As shown in example 5.6, for purposes of efficiency, it is also important to consider where you use a DROP or KEEP data set option. Finally, as shown in example 5.7, BE CAREFUL.