(6) SAS PROCEDURES

Up to now, only three SAS procedures (PROC CONTENTS, PROC PRINT, PROC MEANS) have been used without much discussion as to how they work, options available, or why you would use them in lieu of other procedures. Also, there has been a lot of explanation of how to convert raw data into SAS data sets, how to read SAS data sets, the rules for naming variables and data sets, etc. All of the discussion of SAS rules and SAS statements will have more "grounding" if put into the context of what you can do with data once it is in a SAS data set.

Some of the main tasks performed by BASE SAS procedures (PROCs) are rearranging data, reporting, counting, and computing descriptive statistics. These tasks are demonstrated using PROC PRINT and PROC REPORT (reporting), PROC FREQ and PROC TABULATE (counting), PROC SORT (rearranging), PROC MEANS, PROC UNIVARIATE, PROC TABULATE, and PROC REPORT (descriptive statistics). The data to be used in discussing these procedures are all cancer deaths among residents of New York State from the 1999 New York State vital statistics death file. There are 37,583 records in the raw data file and four variables: county, gender, age, and cause of death. The first ten records in the raw data file are as follows...

```
011 70C349D
012 62C189E
012 36C56 F
011 61C911D
012 45C80 D
012 88C259E
462 63C519E
012 76C539E
012 82C349E
412 89C509D
```

The record layout for these data is:

```
column 1-2     county number
column 3       gender (1=male, 2=female)
column 4-6     age (with 0 indicating less than 1 year of age)
column 7-10    cause of death (ICD-10 cause of death code)
column 11      place of death (A Hospital - DOA            B Hospital - ER
                               C Hospital - outpatient     D Hospital - inpatient
                               E Other institution         F Decedent's residence
                               G Other private home        H Other non-institution
                               N Not in hospital           Z Unknown or not classifiable)
```

Before we do anything with any of the PROCs, the raw data must be converted to a SAS data set using a data step.

...Example 6.1...

```
libname x 'e:\';

data x.cancer99;                                                    1
infile "e:\cancer99.dat";                                           2
input
county $  1-2
gender $  3
age       4-6
cause  $  7-10
place  $ 11
;                                                                   3
run;
```

1 A PERMANENT SAS data set named CANCER99 will be created in the directory E;\
2 An INFILE statement is used to tell SAS where the raw data are located.
3 Column input is used to read the data.

...PROC PRINT/REPORTING

There are numerous PROCs that can be used to create reports in SAS, You can also use a data step to create reports. One of the easiest ways to create a report is with PROC PRINT. As stated in the SAS online help...

The PRINT procedure prints the observations in a SAS data set for simple reports. PROC PRINT prints a listing of the values of some or all of the variables in a SAS data set. You can produce customized reports using procedure options and statements.

A portion of the data set CANCER99 can be printed by using the FIRSTOBS and OBS options discussed earlier.

...Example 6.2...

```
proc print data=x.cancer99 (firstobs=501 obs=510);                    1
run;

  Obs    county    gender    age    cause    place                     2
  501      33        2        79    C56        B
  502      33        2        77    C509       D
  503      33        1        72    C349       D
  504      33        2        79    C183       D
  505      33        1        52    C349       F
  506      33        2        73    C259       F
  507      05        1        76    C819       D
  508      33        1        65    C349       D
  509      33        1        80    C64        F
  510      26        2        45    C449       D
```

1 The FIRSTOBS and OBS data set options are used to restrict the output from PROC PRINT to observations 501 through 510
2 This is the default output from PROC PRINT: an OBS (observation number) column is provided; the variables are displayed in the order that they are stored in the SAS data set CANCER99; all variables are displayed; the variable names are used to label the columns.

If you wanted to print observations base on the value of one or more variables, you could use a WHERE statement. Deaths from cancer of the LIP include any cause of death code that begins with C00.

...Example 6.3...

```
proc print data=x.cancer99 obs='observation number';                  1
where cause eq : "C00";                                               2
run;

observation
   number      age    county    gender    cause
   10343        83      14        2       C001
   26262        91      96        1       C009
```

1 The OBS= option can be used to label the column that shows the observation number.
2 The WHERE statement restricts PROC PRINT to only those observations with a cause of death that begins with the characters 'C00'.

NOTE: The EQ followed by a colon (:) tells SAS to select all observations with a cause of death the starts with 'C00' regardless of the fourth character in the cause. Notice that both causes have a fourth character. If you had used...

```
where cause eq 'C00';
```

you would see the following message in the LOG...

```
NOTE: No observations were selected from data set X.CANCER99.
NOTE: There were 0 observations read from the data set X.CANCER99.
      WHERE cause='C00';
```

The EQ without the colon tells SAS to select all observations with a cause of death that starts with 'C00' and with a fourth character than is BLANK. The colon in example 6.3 has the effect of limiting the comparison to the length of the argument on the right side of the comparison, in this case 'C00'.

There are many options available within PROC PRINT that allow you to customize the output. A few of most common are illustrated in the next example.

...Example 6.4...

```
proc print data=x.cancer99 (obs=5) noobs label;                    1
var cause age;                                                     2
label                                                              3
cause = "CAUSE OF DEATH"
age = "AGE AT DEATH"
;
run;
```

```
CAUSE OF    AGE AT
 DEATH       DEATH
  C349         70
  C189         62
  C56          36
  C911         61
  C80          45
```

1 The NOOBS option eliminates the observation numbers. The LABEL option tells SAS to use variable labels (not variable names) for column headings.
2 A VAR statement is used to restrict the output to the designated variables, in the specified order.
3 A LABEL statement is used to provide LABELS for the variables (used as the column headings). These labels may be up to 200 characters and are only associated with the variables for the duration of the procedure (NOT PERMANENTLY as is the case when labels are created in a data step).

The default behavior of PROC PRINT is to use variable names rather than labels as column headings. You must ask PROC PRINT to use variable labels as column headings with the LABEL option as shown in example 33, or with the SPLIT option. The SPLIT option of PROC PRINT allows you to specify a character that SAS will use to split the column labels. Modifying example 6.3...

...Example 6.5...

```
proc print data=x.cancer99 (obs=5) noobs split='~';                1
var cause age;
label                                                              1
cause = "CAUSE~OF DEATH"
age = "AGE~AT DEATH";
run;
```

```
 CAUSE         AGE
OF DEATH    AT DEATH

  C349         70
  C189         62
  C56          36
  C911         61
  C80          45
```

1 Placing a '~' in the label and specifying SPLIT='~' changes the appearance of the column labels...

PROC REPORT/REPORTING

There a many more PROC PRINT options and some will be shown in later examples. Even with all the extra options in PROC PRINT, PROC REPORT has many more capabilities than PROC PRINT. The SAS procedures manual describes PROC REPORT as follows...

The REPORT procedure combines features of the PRINT, MEANS, and TABULATE procedures with features of the DATA step in a single report-writing tool that can produce a variety of reports.

PROC REPORT is an interactive procedure in that can be run in both interactive and batch mode. Interactive mode opens a REPORT window in addition to the standard EDITOR/LOG/OUTPUT windows in the SAS display manager. In batch mode, you write and submit statements as with most other SAS procedures. All examples shown here specify the NOWD (no window) option and use PROC REPORT in batch mode.

...Example 6.6...

```
proc report data=x.cancer99 (obs=5) nowd;                                  1
run;

      g
      e                   p
      n                   l
  co  d                   a
  un  e              caus c
  ty  r         age  e     e
  01  1          70  C349  D                                               2
  01  2          62  C189  E
  01  2          36  C56   F
  01  1          61  C911  D
  01  2          45  C80   D
```

1 PROC REPORT is used in batch mode (option NOWD is specified).
2 The default output uses the length of character variables to determine column width, and the value of the COLWIDTH option (default is 9) for numeric varaibles.

Notice that the default output does not include an observation number as in PROC PRINT. There is no option in PROC REPORT to add an observation number to the output. If you want modify the output, you can use a combination of COLUMN and DEFINE statements. The COLUMN statement takes the place of the VAR statement in PROC PRINT, allowing you to specify variables to be printed and their order, though it is not limited to those tasks. The DEFINE statement allows you to change the display of any given column.

...Example 6.7...

```
proc report data=x.cancer99 (obs=5) split='~' nowd;                        1
columns county age gender                                                  2
define county / 'COUNTY' width=6;                                          3
define cause  / 'CAUSE OF~DEATH'  center width=8;
run;

                    g
                    e
                    n
                    d
                    e  CAUSE OF
  COUNTY        age r   DEATH
  01             70 1    C349
  01             62 2    C189
  01             36 2    C56
  01             61 1    C911
  01             45 2    C80
```

1 The SPLIT option can be used to specify a character that breaks a column header at the desired location. As opposed to PROC PRINT, PROC REPORT uses variable LABELS by default.
2 Columns are chosen for the report.
3 DEFINE statements assign attributes to the columns.
4 the output differs from that in example 6.5.

The county data are left-justified, the default for character variables, the header COUNTY is used, and the width of the column is six (as specified in the define statement). Age is right-justified, the default for numeric variables. Neither AGE nor GENDER were specified in a DEFINE statement so they both look as they did in example 6.5 (though the order was changed in the COLUMN statement). Values of the variable CAUSE are centered and the column header and width from the DEFINE statement are used.

...Example 6.8...

```
proc report data=x.cancer99 (obs=5) split='~' nowd;
columns county age gender cause;
define county / 'COUNTY' center width=6;                                    1
define age    / 'AGE AT~DEATH'  width=6;
define gender / 'GENDER' center width=6;
define cause  / 'CAUSE OF~DEATH'  center width=8;
run;
```

```
          AGE AT          CAUSE OF
  COUNTY   DEATH  GENDER   DEATH
    01        70    1       C349
    01        62    2       C189
    01        36    2       C56
    01        61    1       C911
    01        45    2       C80
```

1 Two more DEFINE statements are added to control the column attributes. The CENTER option has no effect on numeric variables is only used for the variable CAUSE.

PROC REPORT can also compute basic statistics.

...Example 6.9...

```
proc format;                                                                1
value $gnd
'1' = 'MALES'
'2' = 'FEMALES'
;
run;

proc report data=x.cancer99 split='~' nowd headskip panels=99;              2
columns county (gender N) age;                                              3
define county / group 'COUNTY' center width=6;
define gender / across 'GENDER' right width=7 format=$gnd.;                 4
define age    / 'MEAN AGE AT DEATH'   width=8 analysis mean format=4.1;     5
break after county / skip;                                                 6
run;
```

```
             GENDER                MEAN AGE                GENDER                MEAN AGE
  COUNTY  FEMALES    MALES          N  AT DEATH     COUNTY  FEMALES    MALES          N  AT DEATH

    01        350      374        724      72.9
                                                       58         85      113        198      70.8
    02         49       62        111      71.0
                                                       59        938      954       1892      71.8
    03        294      260        554      73.0
                                                       60         35       61         96      72.9
<MORE OUTPUT>
```

1 A user-written format is created named $GND, It will be used to control the display of values of the variable gender (PROC FORMAT is covered in chapter 7).
2 Two new options are specified. HEADSKIP adds a blank line below each column header. PANELS controls how the output is displayed. The default of PANELS=1 mimics the behavior of PROC PRINT, i.e. values for one observation printed per line. PANELS=99 asks SAS to fit as many observations as possible on any given line. PANELS=<#> is the general form of this option where you can specify a value for #.

The output from PROC REPORT shows that in addition to being able to display the values of variables in a give data set, PROC REPORT can also compute both the frequency of values of variables (N) and basic statistics (in this example, the MEAN).

NOTE: The PANELS option alone makes PROC REPORT a good alternative to PROC PRINT. For example, if you wanted to print the variables gender, age, and cause for the first 200 observations in the data set, you could use...

```
proc print data=x.cancer99 (obs=200);
var gender age cause;
run;
```

This would print three variables on a line and extend over several pages. If you use PROC REPORT and the PANELS option as follows...

```
proc report data=x.cancer99 (obs=200) panels=99 nowd;
columns gender age cause;
define gender / center width=6;
define age    / width=6;
define cause  / center width=8;
run;
```

the output would most likely fit on one page.

PROC FREQ/COUNTING

Just as there are many ways to create reports with SAS, there are also a number of procedures that can produce counts. Just as PROC PRINT is perhaps the most elemental way to create reports, PROC FREQ is the basic counting procedure. PROC FREQ also has a number of options that allow you to compute statistics (chi-square, Mantel-Haenszel, etc.). As stated in the SAS online help...

The FREQ procedure produces 1-way to n-way frequency and crosstabulation tables. Frequency tables show the distribution of variable values. Crosstabulation tables show combined frequency distributions for two or more variables. For one-way tables, PROC FREQ can compute chi-square tests for equal or specified proportions. For two-way tables, PROC FREQ computes tests and measures of association. For n-way tables, PROC FREQ does stratified analysis, computing statistics within as well as across strata.

...Example 6.10...

```
proc freq data=x.cancer99;                                                    1
table gender;                                                                 2
format gender $gnd.;                                                          3
run;
```

gender	Frequency	Percent	Cumulative Frequency	Cumulative Percent
MALES	18542	49.34	18542	49.34
FEMALES	19041	50.66	37583	100.00

1 PROC FREQ is used to produce counts based on the values of variables in the data set CANCER99.
2 The variable GENDER is specified in a TABLE statement.
3 The format $GND. is used to control the appearance of values of the variable gender (see example 6.9).

All SAS PROCs have a default behavior. If you do not specify the variables to be used in PROC PRINT by using a VAR statement, all variables are printed. If you do not use a TABLE statement in PROC FREQ, you will get a table for each variable in your data set. More than one variable can be specified is a TABLE statement. If you use...

```
table age gender;
```

two tables will be produced, with counts for each level of variables gender and age..

The next example creates counts of data classified by two variables, gender and age, with the observations used restricted to those with an age at death in the range 10 to 17.

...Example 6.11..

```
proc freq data=x.cancer99;
where age between 10 and 17;                                    1
table gender*age;                                               2
format gender $gnd.;
run;

Table of gender by age                                          3
gender     age

Frequency|
Percent  |
Row Pct  |
Col Pct  |      10|      11|      12|      13|      14|      15|      16|      17|  Total
---------+--------+--------+--------+--------+--------+--------+--------+--------+
MALES    |       2 |       2 |       5 |       3 |       4 |       8 |       5 |       6 |      35
         |    2.99 |    2.99 |    7.46 |    4.48 |    5.97 |   11.94 |    7.46 |    8.96 |   52.24
         |    5.71 |    5.71 |   14.29 |    8.57 |   11.43 |   22.86 |   14.29 |   17.14 |
         |   40.00 |   33.33 |   62.50 |   37.50 |   57.14 |   61.54 |   50.00 |   60.00 |
---------+--------+--------+--------+--------+--------+--------+--------+--------+
FEMALES  |       3 |       4 |       3 |       5 |       3 |       5 |       5 |       4 |      32
         |    4.48 |    5.97 |    4.48 |    7.46 |    4.48 |    7.46 |    7.46 |    5.97 |   47.76
         |    9.38 |   12.50 |    9.38 |   15.63 |    9.38 |   15.63 |   15.63 |   12.50 |
         |   60.00 |   66.67 |   37.50 |   62.50 |   42.86 |   38.46 |   50.00 |   40.00 |
---------+--------+--------+--------+--------+--------+--------+--------+--------+
Total           5        6        8        8        7       13       10       10       67
              7.46     8.96    11.94    11.94    10.45    19.40    14.93    14.93   100.00
```

1 Observations used in the PROC are restricted to those in the age range 10 to 17. The BETWEEN/AND are only available in a WHERE data set option or statement.
2 A table is requested with two variables. The asterisk tells SAS to include both variables in one table, i.e to create a table with rows and columns.
3 This is the default output of PROC FREQ with two variables. Notice that the first variable in the TABLE statement is displayed on the left (as rows), while the second variable is displayed across the top (as columns). In addition to counts, percentages are provided and the contents of each cell is explained in the upper left of the table (frequency, and the percentages). The topmost percentage in each cell is the OVERALL percentage, i.e. the percentage of the grand total of 66 in the lower right of the table

If you only want counts (frequencies), you can suppress one or more of the percentages with options on the TABLE statement.

...Example 6.12...

```
proc freq data=x.cancer99;
where age between 17 and 10;                                    1
table gender*age/nocol nopercent;                               2
table gender*age/norow nopercent;                               3
table gender*age/norow nocol nopercent;                         4
format gender $gnd.;
run;
```

Table of gender by age 5

gender / Frequency / Row Pct	10	11	12	13	14	15	16	17	Total
MALES	2 5.71	2 5.71	5 14.29	3 8.57	4 11.43	8 22.86	5 14.29	6 17.14	35
FEMALES	3 9.38	4 12.50	3 9.38	5 15.63	3 9.38	5 15.63	5 15.63	4 12.50	32
Total	5	6	8	8	7	13	10	10	67

Table of gender by age

gender / Frequency / Col Pct	10	11	12	13	14	15	16	17	Total
MALES	2 40.00	2 33.33	5 62.50	3 37.50	4 57.14	8 61.54	5 50.00	6 60.00	35
FEMALES	3 60.00	4 66.67	3 37.50	5 62.50	3 42.86	5 38.46	5 50.00	4 40.00	32
Total	5	6	8	8	7	13	10	10	67

Table of gender by age

gender / Frequency	10	11	12	13	14	15	16	17	Total
MALES	2	2	5	3	4	8	5	6	35
FEMALES	3	4	3	5	3	5	5	4	32
Total	5	6	8	8	7	13	10	10	67

1 The WHERE statement with BETWEEN recognizes a range with the highest value written first.
2 The NOCOL option eliminates COLUMN percentages, while the NOPERCENT option eliminates the OVERALL percentages in each cell, plus the MARGINAL percentages.
3 The NOROW option eliminates COLUMN percentages. You can use more than one table statement (you will want to if the tables have different options).
4 All percentages are eliminated.
5 The three tables show the results of the various options.

A table with three variables would look as follows, restricted to age in the range 55 to 64 and two causes: lung cancer, any cause that starts with C34; breast cancer, any cause that starts with C50.

...Example 6.13...

```
proc freq data=x.cancer99;
where (age between 55 and 64) and (cause eq: 'C34' or cause eq: 'C50');     1
table cause*gender*age/norow nocol nopercent;                               2
format cause $3. gender $gnd.;                                              3
run;
```

Table 1 of gender by age
Controlling for cause=C34

gender / Frequency	55	56	57	58	59	60	61	62	63	64	Total
MALES	72	99	86	99	106	106	134	121	134	143	1100
FEMALES	67	56	69	74	72	71	89	74	96	115	783
Total	139	155	155	173	178	177	223	195	230	258	1883

```
Table 2 of gender by age
Controlling for cause=C50

gender     age

Frequency|      55|      56|      57|      58|      59|      60|      61|      62|      63|      64|  Total
---------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+
MALES    |       1 |       0 |       0 |       0 |       1 |       1 |       0 |       0 |       1 |       1 |       5
---------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+
FEMALES  |      63 |      63 |      55 |      42 |      48 |      59 |      62 |      47 |      51 |      67 |     557
---------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+
Total           64       63       55       42       49       60       62       47       52       68      562
```

1 A WHERE statement is used to restrict observations used by the procedure. The EQ: is used to limit the comparison to the first three characters in the values of the variable CAUSE.
2 A table using three variables is requested. Note where the three variables appear in the table.
3 The format $3. is used to group all the causes starting with C34 and C50 into two tables. Without the format, there would be on table for every cause starting with C34 and C50 (e.g. C501, C502, C503, ... , C509);
4 This is the output. All percentages are suppressed. One table is produced for each value of the variable that appears first in the TABLE statement, while the last two variables are used as rows and columns.

The are few restrictions on the number of variables in the TABLE statement of PROC FREQ. However, if you request a table that requires an "extreme" number of combinations of the variables in the TABLE statement, you may run into a problem since PROC FREQ attempts to create counts in memory, not on disk. One way to restrict the memory needed to produce a table is to use a BY statement. The two tables that were just produced using CAUSE*GENDER*AGE could have also been produced by using...

```
table gender*age/norow nocol nopercent;
by cause;
```

The same tables would have been created with just a slight change in the labels that SAS produces. However, PROC FREQ would only create the GENDER*AGE tables in memory, one such table for each CAUSE in the tables that are produced. When variables are used in a BY statement within either a PROC, there is an important rule: the data set must be sorted in ascending order according to the variable(s) in the BY statement.

...PROC SORT/REARRANGING OBSERVATIONS

PROC SORT will rearrange the order of observations in a SAS data set according to the values of one or more variables. When three variables were used to produce a table using PROC FREQ, the last method used a BY variable and it was stated that to use a variable in a BY statement, the data set must be sorted according to that variable...

```
proc sort data=x.cancer99;
by cause;
run;
```

PROC SORT rearranges the observations in ASCENDING order. An option is avaiable to place the observations in descending order...

```
proc sort data=x.cancer99;
by descending cause;
run;
```

Data sets can be sorted by more than one variable at a time...

```
proc sort data=x.cancer99;
by cause gender;
run;
```

The data set will be sorted in ascending order according to the first variable (cause), then by the second variable (gender) within values of the first variable. If a DESCENDING option is used, it ONLY applies to the variable that follows the option. If the data set is to be sorted in descending order according to both variables, the option must be repeated...

```
proc sort data=x.cancer99;
by descending cause descending gender;
run;
```

Instead of sorting the data set in place, a new data set can be produced by PROC SORT, leaving the original data set unsorted.

...Example 6.14...

```
proc sort data=x.cancer99 out=temp (keep=cause age gender);                    1
where (age between 55 and 64) and (cause eq: 'C34' or cause eq: 'C50');        2
by cause;
run;

* two-way table with a by-variable;
proc freq data=temp;
table gender*age/norow nocol nopercent;                                        3
by cause;                                                                      4
format cause $3. gender $gnd.;
run;
```

```
cause=C34
Table of gender by age
gender     age

Frequency|      55|      56|      57|      58|      59|      60|      61|      62|      63|      64|  Total
---------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+
MALES    |     72 |     99 |     86 |     99 |    106 |    106 |    134 |    121 |    134 |    143 |   1100
---------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+
FEMALES  |     67 |     56 |     69 |     74 |     72 |     71 |     89 |     74 |     96 |    115 |    783
---------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+
Total         139      155      155      173      178      177      223      195      230      258     1883


cause=C50
Table of gender by age
gender     age

Frequency|      55|      56|      57|      58|      59|      60|      61|      62|      63|      64|  Total
---------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+
MALES    |      1 |      0 |      0 |      0 |      1 |      1 |      0 |      0 |      1 |      1 |      5
---------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+
FEMALES  |     63 |     63 |     55 |     42 |     48 |     59 |     62 |     47 |     51 |     67 |    557
---------+--------+--------+--------+--------+--------+--------+--------+--------+--------+--------+
Total          64       63       55       42       49       60       62       47       52       68      562
```

1 The data set X.CANCER99 remains unsorted, while a new data set TEMP is created that has ONLY three variables from data set X.CANCER99.
2 The WHERE statement restricts the observations that are sorted and placed in the new data set.
3 Only two variables appear in the table statement.
4 A BY statement is used to produce one table for each level of the by-variable.

...PROC MEANS/DESCRIPTIVE STATISTICS

PROC MEANS is any way to compute descriptive statistics for numeric variables in a SAS data set. As stated in SAS online help...

PROC MEANS computes statistics for an entire SAS data set or for groups of observations in the data set. If you use a BY statement, PROC MEANS calculates descriptive statistics separately for groups of observations. Each group is composed of observations having the same values of the variables used in the BY statement. The groups can be further subdivided by the use of the CLASS statement. PROC MEANS can optionally create one or more SAS data sets containing the statistics calculated.

PROC MEANS is the easiest and most direct descriptive procedure for computing univariate statistics.

Example 1.1 shows the default output from PROC MEANS. Several options can be used to control the appearance of the output and to request additional statistics.

...Example 6.15...
```
proc means data=x.cancer99 maxdec=1 mean lclm uclm min p25 median p75 max;     1
run;
```

Analysis Variable : age

Mean	Lower 95% CL for Mean	Upper 95% CL for Mean	Minimum	25th Pctl	Median	75th Pctl	Maximum
70.6	70.4	70.7	0.0	62.0	73.0	80.0	112.0

1 No variables are specified in the procedure. By default, SAS will analyze all numeric variables in the data set and the only numeric variable is AGE. The number of decimal places is controlled by the MAXDEC option. A number of descriptive statistics are requested using keywords. They replace the default statsitics.

The statistics keywords that can be used in PROC MEANS are as follows (some should be obvious, others are not - you'll need the documentation to determine what some of the following are)...

Descriptive statistics: **CSS CV KURTOSIS|KURT LCLM MAX MEAN MIN N NMISS RANGE SKEWNESS|SKEW STDDEV|STD STDERR SUM SUMWGT UCLM USS VAR**

Quantile statistics: **P1 P5 P10 Q1|P25 MEDIAN|P50 Q3|P75 P90 P95 P99 QRANGE**

Hypothesis testing: **PROBT T**

Analysis can be done within groups of the values of variables within the data set by using a CLASS statement. If we were interested in the statistics computed separately for males and females, we can use gender as a class variable.

...Example 6.16...
```
proc means data=x.cancer99 maxdec=1 mean min q1 median q3 max;
class gender;                                                                   1
format gender $gnd.;                                                            2
run;
```

Analysis Variable : age

gender	N Obs	Mean	Minimum	Lower Quartile	Median	Upper Quartile	Maximum
MALES	18542	70.1	0.0	62.0	72.0	79.0	112.0
FEMALES	19041	71.0	0.0	63.0	73.0	81.0	108.0

1 The CLASS statement in PROC MEANS produces statistics for every value of the variable GENDER.
2 A format (created in example 6.8) is used to label values of the variable GENDER.

More than one class variable can be specified. We can limit the observations used with a WHERE statement and use the cause of death as another class variable.

...Example 6.17...

```
proc means data=x.cancer99 maxdec=1 mean min q1 median q3 max;
class cause gender;                                                    1
where cause in: ('C18' 'C34' 'C50' 'C56' 'C61');                       2
format cause $3.;                                                      3
run;
```

Analysis Variable : age

cause	gender	N Obs	Mean	Minimum	Lower Quartile	Median	Upper Quartile	Maximum
C18	1	1705	72.5	24.0	65.0	74.0	81.0	103.0
	2	1940	75.2	21.0	68.0	77.0	85.0	108.0
C34	1	5317	69.4	17.0	62.0	71.0	77.0	112.0
	2	4319	70.5	6.0	63.0	72.0	79.0	101.0
C50	1	41	70.7	38.0	60.0	74.0	80.0	98.0
	2	3113	68.4	26.0	57.0	70.0	81.0	102.0
C56	2	979	68.7	18.0	59.0	70.0	79.0	102.0
C61	1	2050	78.1	42.0	72.0	79.0	85.0	102.0

1 Two CLASS variables are specified
2 A WHERE statement is used to limit analysis to specific cancers: C18, colon; C34, lung; C50, breast; C51, ovary; C61, prostate. The IN followed by a colon (:) limits the comparison to the lengths of the character strings within quotes.
3 A format statement is used to aggregate the causes using the first three characters of the cause of death.

The advantage of using a CLASS statement for producing analyses within groups is that the data set does not have to be sorted according to the variable(s) in the CLASS statement. Another way to produce group-specific analyses is to use a BY statement...

```
by cause gender;
```

However, by-group processing requires that a data set be sorted in ascending order according to all variables used with the BY statement. PROC SORT can be used prior to PROC MEANS to sort the data set in ascending order by cause and gender.

...PROC UNIVARIATE/DESCRIPTIVE STATISTICS

Though PROC MEANS can produce a number of descriptive statistics, PROC UNIVARIATE is also easy to use and can produce a more comprehensive description of the numeric variables in a data set. Unless you specify otherwise, PROC UNIVARIATE will analyze all numeric variables in a data set.

...Example 6.18...

```
proc univariate data=x.cancer99;
run;
```

```
Variable:  age
                          Moments

N                    37578    Sum Weights             37578
Mean              70.5725691    Sum Observations      2651976
Std Deviation     13.8227372    Variance             191.068063
Skewness          -0.8225036    Kurtosis             1.22025703
Uncorrected SS     194336524    Corrected SS          7179764.6
Coeff Variation    19.586558    Std Error Mean       0.07130619
```

```
             Basic Statistical Measures

    Location                    Variability

Mean     70.57257     Std Deviation           13.82274
Median   73.00000     Variance               191.06806
Mode     73.00000     Range                  112.00000
                      Interquartile Range     18.00000


           Tests for Location: Mu0=0

Test           -Statistic-    -----p Value------

Student's t    t  989.7117    Pr > |t|    <.0001
Sign           M   18785.5    Pr >= |M|   <.0001
Signed Rank    S   3.529E8    Pr >= |S|   <.0001


Quantiles (Definition 5)

Quantile      Estimate

100% Max           112
99%                 95
95%                 90
90%                 86
75% Q3              80
50% Median          73
25% Q1              62
10%                 52
5%                  46
1%                  32
0% Min               0


Variable:  age

        Extreme Observations

----Lowest----        ----Highest---

Value      Obs        Value      Obs

    0    36660          103    32187
    0    36313          103    33501
    0    34867          105    27425
    0    33633          108     9723
    0    28582          112    27533


               Missing Values

                      -----Percent Of-----
Missing                           Missing
  Value       Count     All Obs         Obs

      .           5        0.01      100.00

               Missing Values
                      -----Percent Of-----
Missing                           Missing
  Value       Count     All Obs         Obs

      .           4        0.01      100.00
```

As you can see, PROC UNIVARIATE produces a very comprehensive list of descriptive statistics. The data labeled EXTREME OBSERVATIONS shows the lowest and highest values of the analysis variable and the observation number. You can specify one or mare variables in an ID statement and the value of the variable(s) will be listed with the extremes...

```
proc univariate data=x.cancer99;
id gender cause;
run;
```

```
Variable:  age
                                 Extreme Observations

-------------Lowest-----------        ------------Highest-----------

Value   gender   cause       Obs        Value   gender   cause       Obs

    0   2        C719      36660          103   1        C169      32187
    0   1        C80       36313          103   2        C189      33501
    0   2        C80       34867          105   2        C189      27425
    0   2        C719      33633          108   2        C189       9723
    0   1        C439      28582          112   1        C349      27533
```

There is no MAXDEC option within PROC UNIVARIATE, in fact there is no way to control the number of decimal places produced. There is a ROUND option, but it does not perform the same function in UNIVARIATE as MAXDEC does in MEANS.

Prior to version 8, you could not use a CLASS statement in PROC UNIVARIATE to produce group-specific analyses. BY-GROUP processing was required. There is now CLASS statement available in PROC UNIVARIATE that permits up to two CLASS variables. Rather than sort the data (as is required using a by-variable), a CLASS statement is used to compute statistics within gender groups.

...Example 6.19...

```
proc univariate data=x.cancer99 plot;                                        1
class gender;                                                                2
run;
```

1 A PLOT option is used to add graphics to the output.
2 The CLASS statement produces gender-specific output.

Rather than show all the statistical output (it would be similar to the output in example 6.18, but with separate analyses for males and females), the graphical output looks as follows...

```
Variable:  age
gender = 1

                          Histogram                           #  Boxplot                         Normal Probability Plot
  112.5+*                                                     1    0        112.5+                                                  *
       .                                                                         |
       .*                                                    18    |             |                                                ++*
       .***                                                 155    |             |                                            ++++ **
       .*********                                           551    |             |                                        +++******
       .************************                           1494    |             |                                     *******
       .**************************************             2398    |             |                                ******
       .************************************************   3052 +-----+          |                           ******
       .************************************************   3050 *--+--*          |                       *****+
       .*************************************              2350 |     |          |                    ****+
       .*****************************                      1828 +-----+          |                ******+
   57.5+*********************                              1313    |         57.5+              ***+
       .****************                                    982    |             |          *****
       .**********                                          618    |             |       ****
       .******                                              329    |             |   ++***
       .***                                                 144    0             |+++***
       .**                                                   81    0             |  **
       .*                                                    59    0             |***
       .*                                                    32    0             |*
       .*                                                    36    0             |*
       .*                                                    16    0             |*
       .*                                                    15    *             |*
    2.5+*                                                    19    *          2.5+*
        ----+----+----+----+----+----+----+----+----+---                         +----+----+----+----+----+----+----+----+----+----+
        * may represent up to 64 counts                                               -2        -1         0        +1        +2
```

```
Variable:  age
gender = 2

                          Histogram                          #  Boxplot                        Normal Probability Plot
  107.5+*                                                    2    |        107.5+                                                 +*
       .*                                                   41    |             |                                              +++ *
       .*****                                              276    |             |                                          ++++*****
       .****************                                   920    |             |                                      +********
       .********************************                  1913    |             |                                   ******
       .********************************************      2640 +-----+          |                              ******
       .************************************************  2906 |     |          |                          *****+
   72.5+***********************************************   2818 *--+--*      72.5+                      *****+
       .**********************************                2065 |     |          |                   ****++
       .***************************                       1614 +-----+          |                 ***+
       .*********************                             1269    |             |              ****
       .******************                                1047    |             |           ****
       .***********                                        648    |             |       +****
       .********                                           437    |             |    +****
   37.5+****                                               203    0         37.5++++***
       .**                                                 101    0             | ***
       .*                                                   49    0             |*
       .*                                                   19    0             |*
       .*                                                   18    0             |*
       .*                                                   18    0             |*
       .*                                                   16    0             |*
    2.5+*                                                   17    *          2.5+*
        ----+----+----+----+----+----+----+----+----+---                         +----+----+----+----+----+----+----+----+----+----+
        * may represent up to 61 counts                                              -2        -1         0       +1        +2
```

If you use a BY statement (requiring sorted observations) instead of a CLASS statement, you get one additional plot.

...Example 6.20...

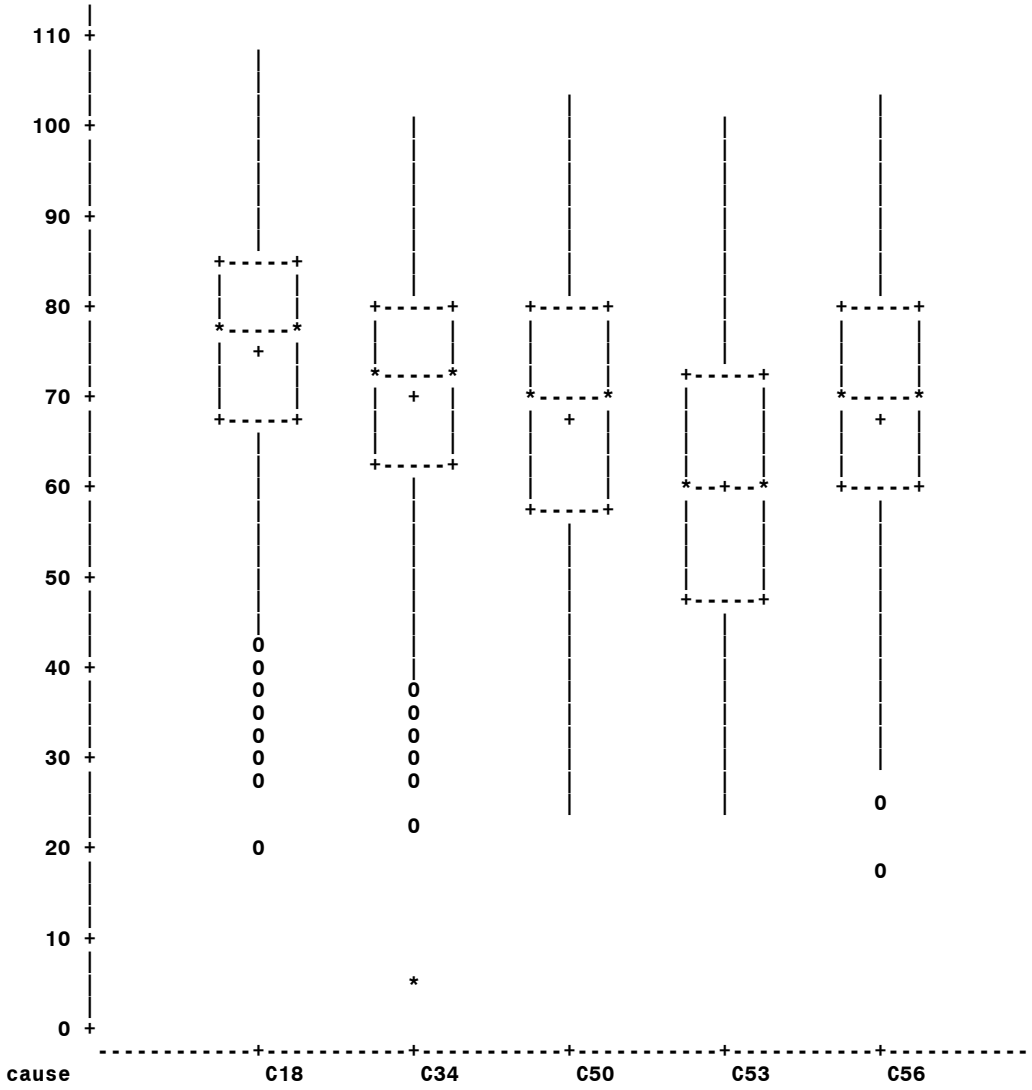
```
proc sort data=x.cancer99 out=f_temp;                                          1
by cause;
where cause in: ('C18' 'C34' 'C50' 'C53' 'C56') and gender eq '2';             2
run;

proc univariate data=f_temp plot;                                              3
by cause;                                                                      4
format cause $3.;                                                              5
run;
```

1 PROC SORT is used to create a new data set named F_TEMP. The observations will be ordered by CAUSE.
2 The new data set is limited to five causes of death (C18, colon; C34, lung; C50, breast; C51, ovary; C56, cervical) and to females.
3 F_TEMP is used in PROC UNIVARIATE and plots are requested.
4 A BY statement is used to produce cause-specific analyses.
5 The data are grouped by the first three characters in the cause of death.

The output would be similar to that in the last two examples, with one additional plot that is useful in comparing the age distribution among levels of the variable used in the BY statement, i.e. cause of death.

```
Variable:  age
Schematic Plots
```



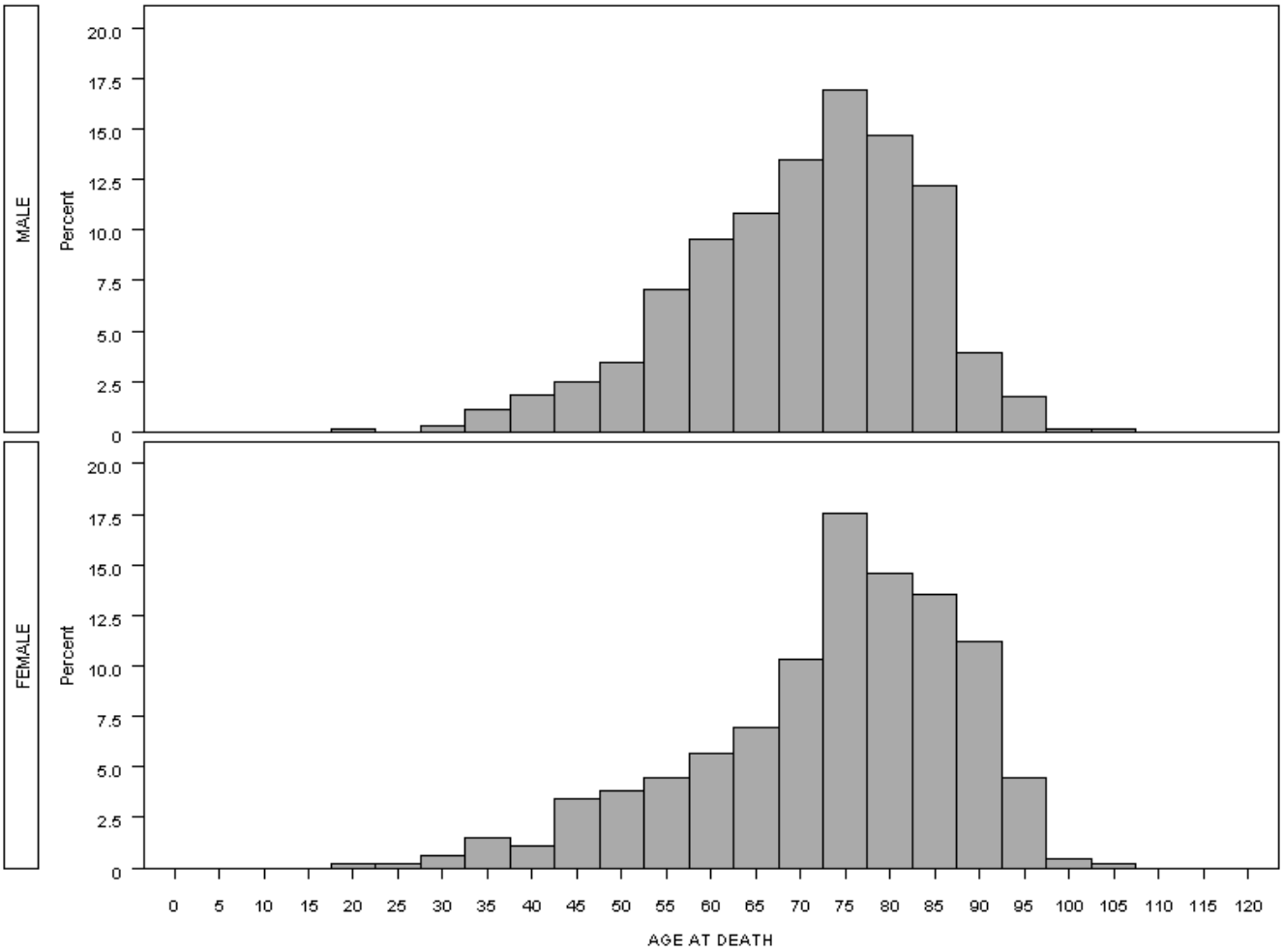
The boxes span the 25th-to-75th percentiles, the line in the middle shows the median, and the + indicates the location of the mean. The vertical dashed lines span 1.5 times the inter-quartile range. Symbols are used to locate extreme values (possible outliers).

If SAS/GRAPH is available, you can produce a very nice looking histogram with PROC UNIVARIATE.

...Example 6.21...

```
proc univariate data=x.cancer99 noprint;                                1
class gender;                                                           2
histogram / midpoints=0 to 120 by 5;                                    3
where cause eq : 'C16';                                                 4
format gender $gnd.;                                                    5
label age = 'AGE AT DEATH';                                             6
run;
```

1 All the default printed output is suppressed using the NOPRINT option.
2 The analyses will be dine for each level of the variable gender;
3 The HISTOGRAM statement will produce a histogram for males and females. The MIDPOINTS option allows control over the X-axis.
4 The analysis is limited to stomach cancer (C16).
5 Labels for levels of the variable gender will be place in the output.
6 A label will be used in the output rather than the name of the varaible.



...OUTPUT DELIVERY SYSTEM (ODS)

One complaint about SAS PROCs in the past is that they produced output that was not as (for want of a better word) 'pretty' as that produced by other software. That all changed with the addition of the output delivery system (ODS) to base SAS software. The output from any SAS procedure can be directed to a variety of different formats, including but not limited to: HTML; PDF; RTF (rich-text-format for inclusion of output in word processing documents). A few additions can be made to the SAS code used in example 6.17 and a the output can be placed in an HTML file.

...Example 6.22...

```
ods listing close;                                                    1
ods html file='z:\means.htm';                                         2

title;                                                                3
proc means data=x.cancer99 maxdec=1 mean min q1 median q3 max;        4
class cause gender;
where cause in: ('C18' 'C34' 'C50' 'C56' 'C61');
format cause $3.;
run;

ods html close;                                                       5
ods listing;                                                          6
```

1 The normal output (i.e. written to the output window) is suppressed.
2 Another ODS statement is used to direct the output of any subsequent PROC to the file MEANS.HTM in HTML format.
3 A blank title statement eliminates the default title of *The SAS System*.
4 PROC MEANS is run just as was done in example 6.17.
5 The HTML destination is closed. You cannot use any files produced using ODS until the file is closed.
6 Normal output to the output window in enabled.

The HTML output was inserted in the document and looks as follows...

Analysis Variable : age								
cause	gender	N Obs	Mean	Minimum	Lower Quartile	Median	Upper Quartile	Maximum
C18	1	1705	72.5	24.0	65.0	74.0	81.0	103.0
	2	1940	75.2	21.0	68.0	77.0	85.0	108.0
C34	1	5317	69.4	17.0	62.0	71.0	77.0	112.0
	2	4319	70.5	6.0	63.0	72.0	79.0	101.0
C50	1	41	70.7	38.0	60.0	74.0	80.0	98.0
	2	3113	68.4	26.0	57.0	70.0	81.0	102.0
C56	2	979	68.7	18.0	59.0	70.0	79.0	102.0
C61	1	2050	78.1	42.0	72.0	79.0	85.0	102.0

The output delivery system has its own manual. In addition to the manual, there are also several books about ODS that have been written by SAS users. This example is just intended to show you that you are not limited to writing PROC output to the output window. If you want to learn more about how to create various types of output with ODS, I suggest that you buy a book!

In addition to being able to create different types of 'pretty' output from PROCs, ODS also has a few other uses. The first is that it allows you to pick-and-choose among the various sections of output from procedures. Each portion of SAS procedure output has a name. You can find the names of the various portions using the ODS TRACE statement.

...Example 6.23...

```
*** write TRACE results to LOG;
ods trace on;                                                         1
proc univariate data=x.cancer99;
run;
```

```
ods trace off;                                                          2

*** write TRACE results to OUTPUT;
ods trace on/listing;                                                   3
proc univariate data=x.cancer99;
run;
ods trace off;
```

1 TRACING is turned on with ODS TRACE ON — trace results are written to the SAS LOG.
2 You can turn off TRACING with ODS TRACE OFF.
3 TRACING is turned on again, but trace results are added to the OUTPUT.

The names of the portions of the PROC UNIVARIATE output are: Moments; BasicMeasures; TestsForLocation; Quantiles; ExtremeObs; MissingValues. You can run the above examples to see what output is associated with each of the above names.

If you only wanted to the BasicMeasures, you could run PROC UNIVARIATE and request that only that portion of the output.

...Example 6.24...

```
proc univariate data=x.cancer99;
ods select basicmeasures;                                               1
run;
```

1 An ODS SELECT statement is used within PROC UNIVARIATE to specify that only the BASICMEASURES are required.

Once you know the names of the various sections of PROC output, you can use another ODS feature. Any portion of PROC output can be written to a SAS data set.

...Example 6.25...

```
ods output basicmeasures=bame_uni;                                      1
ods listing close;                                                      2

proc univariate data=x.cancer99;                                        3
run;

ods listing;                                                            4

proc print data=bame_uni;                                               5
run;
```

1 An ODS statement is used to tell SAS to write the BASICMEASURES from PROC UNIVARIATE to a SAS sata set named BAME_UNI.
2 The LISTING destination is closed - no printed output will be created from PROCs until the LISTING destination is reopened.
3 PROC UNIVARIATE is run.
4 The LISTING destination is reopened.
5 The data set BAME_UNI is printed.

The data in BAME_UNI looks as follows...

Obs	Var Name	Loc Measure	LocValue	VarMeasure	VarValue
1	age	Mean	70.57257	Std Deviation	13.82274
2	age	Median	73.00000	Variance	191.06806
3	age	Mode	73.00000	Range	112.00000
4	age	_		Interquartile Range	18.00000

(7) LABELING AND GROUPING (PROC FORMAT)

The output from SAS PROCs contains variable names and/or variable labels. Variable labels expand on the information about your data that is contained in a variable name. Formats can be used to associate more information to the values of variables. Formats can also be used to group observations based on the values of variables. You have already seen some SAS-supplied formats, e.g. MMDDYY10. used when printing the value of a variable that contains a date. PROC FORMAT allows you to create your own formats (or 'rules') to control the appearance of the values of variables.

...LABELING VALUES/GROUPING OBSERVATIONS

There are two methods to add more information to variable values. One of them is to create new variables based on variable values. The other is to create formats to label variable values. Creating new variables is done within a data step.

...Example 7.1...

```
data c99;
infile "f:\sasclass\data\cancer99.dat";
input
county $  1-2
gender $  3
age       4-6
cause  $  7-10
place  $ 11
;
run;

if gender eq '1' then newgender = 'MALE  ';                                    1
else                 newgender = 'FEMALE';

if cause eq : 'C34' then newcause = 'BRONCHUS & LUNG';
else
if cause eq : 'C18' then newcause = 'COLON';
else
if cause eq : 'C5O' then newcause = 'BREAST';
else
if cause eq : 'C25' then newcause = 'PANCREAS';
else
if cause eq : 'C61' then newcause = 'PROSTATE';
else                     newcause = 'OTHER';
;
run;

proc print data=c99 (obs=5);
run;
```

```
Obs    county    gender    age    cause    place    newgender    newcause
  1      O1        1        70    C349       D       MALE        BRONCHUS & LUNG
  2      O1        2        62    C189       E       FEMALE      COLON
  3      O1        2        36    C56        F       FEMALE      OTHER
  4      O1        1        61    C911       D       MALE        OTHER
  5      O1        2        45    C8O        D       FEMALE      OTHER
```

1 An IF-THEN-ELSE statement is used to create new variables: NEWGENDER, based on the values of the variable GENDER; NEWCAUSE, based on the values of the varaible CAUSE.

This "new variable" method works, but it is a 'long way' to the solution of labeling variable values. Another method is to use a format. We have already seen some SAS-supplied formats, e.g. DATE9. in example 1.1 to display a date, 4.1 in example 6.9 to round a mean to one decimal place, $3. in example 6.13 to group causes of death by the first three characters in a four character cause. You can create your own formats using PROC FORMAT as was shown in example 6.9. As stated in the SAS online help...

The FORMAT procedure provides a simple method for creating your own formats and informats. The SAS System uses informats and formats to read, display, and print data stored in a SAS data set. With PROC