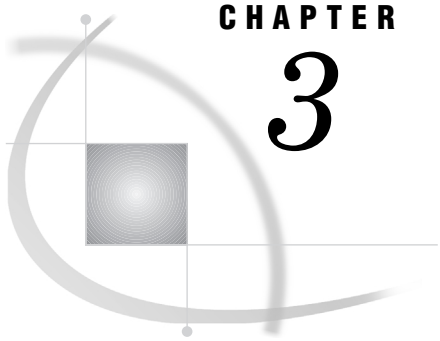

CHAPTER

# 3

# Statements with the Same Function in Multiple Procedures



## Overview

Several statements are available and have the same function in a number of base SAS procedures. Some of the statements are fully documented in *SAS Language Reference: Dictionary*, and others are documented in this section. The following list shows you where to find more information about each statement:

ATTRIB
affects the procedure output and the output data set. The ATTRIB statement does not permanently alter the variables in the input data set. The LENGTH= option has no effect. See *SAS Language Reference: Dictionary* for complete documentation.

BY
orders the output according to the BY groups. See “BY” on page 68.

FORMAT
affects the procedure output and the output data set. The FORMAT statement does not permanently alter the variables in the input data set. The DEFAULT= option is not valid. See *SAS Language Reference: Dictionary* for complete documentation.

FREQ
treats observations as if they appear multiple times in the input data set. See “FREQ” on page 70.

LABEL
affects the procedure output and the output data set. The LABEL statement does not permanently alter the variables in the input data set except when it is used with the MODIFY statement in PROC DATASETS. See *SAS Language Reference: Dictionary* for complete documentation.

QUIT
executes any statements that have not executed and ends the procedure. See “QUIT” on page 72.

WEIGHT
specifies weights for analysis variables in the statistical calculations. See "WEIGHT" on page 73.

WHERE
subsets the input data set by specifying certain conditions that each observation must meet before it is available for processing. See "WHERE" on page 77.

# Statements

## BY

**Orders the output according to the BY groups.**

**See also:** "Creating Titles That Contain BY-Group Information" on page 54

**BY** <DESCENDING> *variable-1*
<… <DESCENDING> *variable-n*>
<NOTSORTED>;

### Required Arguments

***variable***
specifies the variable that the procedure uses to form BY groups. You can specify more than one variable. If you do not use the NOTSORTED option in the BY statement, the observations in the data set must either be sorted by all the variables that you specify, or they must be indexed appropriately. Variables in a BY statement are called *BY variables*.

### Options

**DESCENDING**
specifies that the observations are sorted in descending order by the variable that immediately follows the word DESCENDING in the BY statement.

**NOTSORTED**
specifies that observations are not necessarily sorted in alphabetic or numeric order. The observations are grouped in another way, for example, chronological order.

The requirement for ordering or indexing observations according to the values of BY variables is suspended for BY-group processing when you use the NOTSORTED option. In fact, the procedure does not use an index if you specify NOTSORTED. The procedure defines a BY group as a set of contiguous observations that have the same values for all BY variables. If observations with the same values for the BY variables are not contiguous, the procedure treats each contiguous set as a separate BY group.

*Note:* You cannot use the NOTSORTED option in a PROC SORT step. △

*Note:* You cannot use the GROUPFORMAT option, which is available in the BY statement in a DATA step, in a BY statement in any PROC step. △

## BY-Group Processing

Procedures create output for each BY group. For example, the elementary statistics procedures and the scoring procedures perform separate analyses for each BY group. The reporting procedures produce a report for each BY group.

*Note:* All base procedures except PROC PRINT process BY groups completely independently. PROC PRINT can report the number of observations in each BY group as well as the number of observations in all BY groups. Similarly, PROC PRINT can sum numeric variables in each BY group and across all BY groups. △

You can use only one BY statement in each PROC step. When you use a BY statement, the procedure expects an input data set that is sorted by the order of the BY variables or one that has an appropriate index. If your input data set does not meet these criteria, an error occurs. Either sort it with the SORT procedure or create an appropriate index on the BY variables.

Depending on the order of your data, you may need to use the NOTSORTED or DESCENDING option in the BY statement in the PROC step.

For more information on

- the BY statement, see *SAS Language Reference: Dictionary*.
- PROC SORT, see Chapter 33, "The SORT Procedure," on page 1005.
- creating indexes, see "INDEX CREATE Statement" on page 363.

## Procedures That Support the BY Statement

| | |
|---|---|
| CALENDAR | RANK |
| CHART | REPORT (nonwindowing environment only) |
| COMPARE | SORT (required) |
| CORR | STANDARD |
| FORMS | SUMMARY |
| FREQ | TABULATE |
| MEANS | TIMEPLOT |
| PLOT | TRANSPOSE |
| PRINT | UNIVARIATE |

*Note:* In the SORT procedure, the BY statement specifies how to sort the data. With the other procedures, the BY statement specifies how the data are currently sorted. △

## Example

This example uses a BY statement in a PROC PRINT step. There is output for each value of the BY variable, Year. The DEBATE data set is created in "Example: Temporarily Dissociating a Format from a Variable" on page 63.

```
options nodate pageno=1 linesize=64
        pagesize=40;
```

```
proc print data=debate noobs;
   by year;
   title 'Printing of Team Members';
   title2 'by Year';
run;
```

```
                    Printing of Team Members                    1
                            by Year

------------------------ Year=Freshman -------------------------

                     Name      Gender     GPA

                   Capiccio      m       3.598
                   Tucker        m       3.901


------------------------ Year=Sophomore ------------------------

                     Name      Gender     GPA

                    Bagwell      f       3.722
                    Berry        m       3.198
                    Metcalf      m       3.342


------------------------- Year=Junior --------------------------

                     Name    Gender     GPA

                     Gold      f       3.609
                     Gray      f       3.177
                     Syme      f       3.883


------------------------- Year=Senior --------------------------

                   Name        Gender     GPA

                   Baglione      f       4.000
                   Carr          m       3.750
                   Hall          m       3.574
                   Lewis         m       3.421
```

# FREQ

**Treats observations as if they appear multiple times in the input data set.**

**Tip:** You can use a WEIGHT statement and a FREQ statement in the same step of any procedure that supports both statements.

**FREQ** *variable*;

## Required Arguments

***variable***
specifies a numeric variable whose value represents the frequency of the observation. If you use the FREQ statement, the procedure assumes that each observation represents *n* observations, where *n* is the value of *variable*. If *variable* is not an integer, the SAS System truncates it. If *variable* is less than 1 or is missing, the procedure does not use that observation to calculate statistics. If a FREQ statement does not appear, each observation has a default frequency of 1.

The sum of the frequency variable represents the total number of observations.

## Procedures That Support the FREQ Statement

- CORR
- FORMS
- MEANS/SUMMARY
- REPORT
- STANDARD
- TABULATE
- UNIVARIATE

*Note:* PROC FORMS does not calculate statistics. In PROC FORMS, the value of the frequency variable affects the number of form units that are printed for each observation. △

## Example

The data in this example represent a ship's course and the speed (in knots), recorded every hour. The frequency variable, Hours, represents the number of hours that the ship maintained the same course and speed. Each of the following PROC MEANS steps calculates average course and speed. The different results demonstrate the effect of using Hours as a frequency variable.

The following PROC MEANS step does not use a frequency variable:

```
options nodate pageno=1 linesize=64 pagesize=40;

data track;
   input Course Speed Hours @@;
   datalines;
30  4  8 50 7 20
75 10 30 30 8 10
80  9 22 20 8 25
83 11  6 20 6 20
;


proc means data=track maxdec=2 n mean;
   var course speed;
   title 'Average Course and Speed';
run;
```

Without a frequency variable, each observation has a frequency of 1, and the total number of observations is 8.

```
                  Average Course and Speed                    1

                    The MEANS Procedure

                Variable    N            Mean
                ------------------------------
                Course      8           48.50
                Speed       8            7.88
                ------------------------------
```

The second PROC MEANS step uses Hours as a frequency variable:

```
proc means data=track maxdec=2 n mean;
   var course speed;
   freq hours;
   title 'Average Course and Speed';
run;
```

When you use Hours as a frequency variable, the frequency of each observation is the value of Hours, and the total number of observations is 141 (the sum of the values of the frequency variable).

```
                  Average Course and Speed                    1

                    The MEANS Procedure

             Variable               N            Mean
             ----------------------------------------
             Course               141           49.28
             Speed                141            8.06
             ----------------------------------------
```

# QUIT

**Executes any statements that have not executed and ends the procedure.**

**QUIT**;

## Procedures That Support the QUIT Statement

- CATALOG
- DATASETS
- PLOT
- PMENU
- SQL

# WEIGHT

**Specifies weights for analysis variables in the statistical calculations.**

**Tip:** You can use a WEIGHT statement and a FREQ statement in the same step of any procedure that supports both statements.

**WEIGHT** *variable*;

## Required Arguments

***variable***
specifies a numeric variable whose values weight the values of the analysis variables. The values of the variable do not have to be integers. The behavior of the procedure when it encounters a nonpositive weight variable value is as follows:

| Weight value … | The procedure … |
|---|---|
| 0 | counts the observation in the total number of observations |
| less than 0 | converts the weight value to zero and counts the observation in the total number of observations |
| missing | excludes the observation from the analysis |

Different behavior for nonpositive values is discussed in the WEIGHT statement syntax under the individual procedure.

Prior to Version 7 of the SAS System, no base procedure excluded the observations with missing weights from the analysis. Most SAS/STAT procedures, such as PROC GLM, have always excluded not only missing weights but also negative and zero weights from the analysis. You can achieve this same behavior in a base procedure that support the WEIGHT statement by using EXCLNPWGT in the PROC statement.

The procedure substitutes the value of the WEIGHT variable for $w_i$, which appears in "Keywords and Formulas" on page 1458.

## Procedures That Support the WEIGHT Statement

- CORR
- FREQ
- MEANS/SUMMARY
- REPORT
- STANDARD
- TABULATE
- UNIVARIATE

*Note:* In PROC FREQ, the value of the variable in the WEIGHT statement represents the frequency of occurrence for each observation. See "WEIGHT Statement" on page 524 for more information. △

## Calculating Weighted Statistics

The procedures that support the WEIGHT statement also support the VARDEF= option, which lets you specify a divisor to use in the calculation of the variance and standard deviation.

By using a WEIGHT statement to compute moments, you assume that the $i$th observation has a variance that is equal to $\sigma^2/w_i$. When you specify VARDEF=DF (the default), the computed variance is a weighted least squares estimate of $\sigma^2$. Similarly, the computed standard deviation is an estimate of $\sigma$. Note that the computed variance is not an estimate of the variance of the $i$th observation, because this variance involves the observation's weight which varies from observation to observation.

If the values of your variable are counts that represent the number of occurrences of each observation, use this variable in the FREQ statement rather than in the WEIGHT statement. In this case, because the values are counts, they should be integers. (The FREQ statement truncates any noninteger values.) The variance that is computed with a FREQ variable is an estimate of the common variance, $\sigma^2$, of the observations.

*Note:* If your data come from a stratified sample where the weights $w_i$ represent the strata weights, neither the WEIGHT statement nor the FREQ statement provides appropriate stratified estimates of the mean, variance, or variance of the mean. To perform the appropriate analysis, consider using PROC SURVEYMEANS which is a SAS/STAT procedure that is documented in the *SAS/STAT User's Guide*. △

## Example

As an example of the WEIGHT statement, suppose 20 people are asked to estimate the size of a 12–inch-wide object. Each person is placed at a different distance from the object. As the distance from the object increases, the estimates should become less precise.

The SAS data set SIZE contains the estimate (ObjectSize) at each distance (Distance), and the precision (Precision) for each estimate. Notice that the largest deviation (an overestimate by 8 inches) came at the largest distance (25 feet). As a measure of precision, 1/Distance gives more weight to estimates that were made closer to the object and less weight to estimates that were made at greater distances.

The following statements create the data set SIZE:

```
options nodate pageno=1 linesize=64 pagesize=60;

data size;
   input Distance ObjectSize @@;
   Precision=1/distance;
   datalines;
 5 12   5  8   5 12   5 10
10 17  10 13  10 10  10 12
15 10  15 14  15 19  15 13
20 17  20 14  20  9  20 19
25 12  25 10  25 20  25 15
;
```

The following PROC MEANS step computes the average estimate of the object size while ignoring the weights. Without a WEIGHT variable, PROC MEANS uses the default weight of 1 for every observation. Thus, the estimates of object size at all distances are given equal weight. The average estimate of the object size is overestimated by 1.3 inches.

```
proc means data=size maxdec=3 n mean var stddev;
   var objectsize;
```

```
   title1 'Unweighted Analysis of the SIZE Data Set';
run;
```

```
           Unweighted Analysis of the SIZE Data Set          1

                     The MEANS Procedure

                Analysis Variable : ObjectSize

        N            Mean        Variance         Std Dev
      ----------------------------------------------------
      20          13.300          12.537           3.541
      ----------------------------------------------------
```

The next two PROC MEANS steps use the precision measure (Precision) in the WEIGHT statement and show the effect of using different values of the VARDEF= option. The first PROC step creates an output data set that contains the variance and standard deviation. By down weighting the estimates made at greater distances, the weighted average estimate of the object size is closer to the actual size.

```
proc means data=size maxdec=3 n mean var stddev;
   weight precision;
   var objectsize;
   output out=wtstats var=Est_SigmaSq std=Est_Sigma;
   title1 'Weighted Analysis Using Default VARDEF=DF';
run;

proc means data=size maxdec=3 n mean var std
                      vardef=weight;
   weight precision;
   var objectsize;
   title1 'Weighted Analysis Using VARDEF=WEIGHT';
run;
```

In the first PROC MEANS step, the variance is an estimate of $\sigma^2$, where the variance of the *i*th observation is assumed to be $var\left(x_i\right) = \sigma^2/w_i$ and $w_i$ is the weight for the *i*th observation. In the second PROC MEANS step, the computed variance is an estimate of $\left(n - 1/n\right)\ \sigma^2/\overline{w}$, where $\overline{w}$ is the average weight. For large n, this is an approximate estimate of the variance of an observation with average weight.

```
         Weighted Analysis Using Default VARDEF=DF          1

                     The MEANS Procedure

                Analysis Variable : ObjectSize

        N            Mean        Variance         Std Dev
      ----------------------------------------------------
      20          12.352           0.951           0.975
      ----------------------------------------------------
```

```
            Weighted Analysis Using VARDEF=WEIGHT              2

                      The MEANS Procedure

                  Analysis Variable : ObjectSize

        N            Mean        Variance         Std Dev
      ----------------------------------------------------
      20          12.352           9.892           3.145
      ----------------------------------------------------
```

The following statements create and print a data set with the weighted variance and weighted standard deviation of each observation. The DATA step combines the output data set that contains the variance and the standard deviation from the weighted analysis with the original data set. The variance of each observation is computed by dividing Est_SigmaSq, the estimate of $\sigma^2$ from the weighted analysis when VARDEF=DF, by each observation's weight (Precision). The standard deviation of each observation is computed by dividing Est_Sigma, the estimate of $\sigma$ from the weighted analysis when VARDEF=DF, by the square root of each observation's weight (Precision).

```
data wtsize(drop=_freq_ _type_);
   set size;
   if _n_=1 then set wtstats;
   Est_VarObs=est_sigmasq/precision;
   Est_StdObs=est_sigma/sqrt(precision);

proc print data=wtsize noobs;
   title 'Weighted Statistics';
   by distance;
   format est_varobs est_stdobs
          est_sigmasq est_sigma precision 6.3;
```

```
run;
```

```
                    Weighted Statistics                    4

------------------------- Distance=5 -------------------------

  Object                 Est_       Est_     Est_      Est_
   Size     Precision    SigmaSq     Sigma    VarObs    StdObs

     12        0.200       0.951     0.975     4.755     2.181
      8        0.200       0.951     0.975     4.755     2.181
     12        0.200       0.951     0.975     4.755     2.181
     10        0.200       0.951     0.975     4.755     2.181


------------------------- Distance=10 ------------------------

  Object                 Est_       Est_     Est_      Est_
   Size     Precision    SigmaSq     Sigma    VarObs    StdObs

     17        0.100       0.951     0.975     9.511     3.084
     13        0.100       0.951     0.975     9.511     3.084
     10        0.100       0.951     0.975     9.511     3.084
     12        0.100       0.951     0.975     9.511     3.084


------------------------- Distance=15 ------------------------

  Object                 Est_       Est_     Est_      Est_
   Size     Precision    SigmaSq     Sigma    VarObs    StdObs

     10        0.067       0.951     0.975    14.266     3.777
     14        0.067       0.951     0.975    14.266     3.777
     19        0.067       0.951     0.975    14.266     3.777
     13        0.067       0.951     0.975    14.266     3.777


------------------------- Distance=20 ------------------------

  Object                 Est_       Est_     Est_      Est_
   Size     Precision    SigmaSq     Sigma    VarObs    StdObs

     17        0.050       0.951     0.975    19.021     4.361
     14        0.050       0.951     0.975    19.021     4.361
      9        0.050       0.951     0.975    19.021     4.361
     19        0.050       0.951     0.975    19.021     4.361


------------------------- Distance=25 ------------------------

  Object                 Est_       Est_     Est_      Est_
   Size     Precision    SigmaSq     Sigma    VarObs    StdObs

     12        0.040       0.951     0.975    23.776     4.876
     10        0.040       0.951     0.975    23.776     4.876
     20        0.040       0.951     0.975    23.776     4.876
     15        0.040       0.951     0.975    23.776     4.876
```

# WHERE

**Subsets the input data set by specifying certain conditions that each observation must meet before it is available for processing.**

**WHERE** *where-expression*;

## Required Arguments

***where-expression***
is a valid arithmetic or logical expression that generally consists of a sequence of operands and operators. See *SAS Language Reference: Dictionary* for more information on where processing.

## Procedures That Support the WHERE Statement

You can use the WHERE statement with any of the following base SAS procedures that read a SAS data set:

CALENDAR
CHART
COMPARE
CORR
DATASETS (APPEND statement)
FORMS
FREQ
MEANS/SUMMARY
PLOT
PRINT
RANK
REPORT
SORT
SQL
STANDARD
TABULATE
TIMEPLOT
TRANSPOSE
UNIVARIATE

## Details

- The CALENDAR and COMPARE procedures and the APPEND statement in PROC DATASETS accept more than one input data set. See the documentation for the specific procedure for more information.
- To subset the output data set, use the WHERE= data set option:

```
proc report data=debate nowd
            out=onlyfr(where=(year='1'));
run;
```

For more information on WHERE=, see *SAS Language Reference: Dictionary*.

## Example

In this example, PROC PRINT prints only those observations that meet the condition of the WHERE expression. The DEBATE data set is created in "Example: Temporarily Dissociating a Format from a Variable" on page 63.

```
options nodate pageno=1 linesize=64
        pagesize=40;


proc print data=debate noobs;
   where gpa>3.5;
   title 'Team Members with a GPA';
   title2 'Greater than 3.5';
```

```
run;
```

```
                  Team Members with a GPA                    1
                      Greater than 3.5

          Name        Gender    Year          GPA

          Capiccio      m       Freshman     3.598
          Tucker        m       Freshman     3.901
          Bagwell       f       Sophomore    3.722
          Gold          f       Junior       3.609
          Syme          f       Junior       3.883
          Baglione      f       Senior       4.000
          Carr          m       Senior       3.750
          Hall          m       Senior       3.574
```