Chapter 37

The TABULATE Procedure

ABSTRACT

The TABULATE procedure displays descriptive statistics in tabular format. The value in each table cell is calculated from the variables and statistics that define the pages, rows, and columns of the table. The statistic associated with each cell is calculated on values from all observations in that category. PROC TABULATE computes many of the same statistics that are computed by other descriptive statistical procedures such as MEANS, FREQ, and SUMMARY. PROC TABULATE provides

- simple but powerful methods to create tabular reports
- flexibility in classifying the values of variables and establishing hierarchical relationships between the variables
- mechanisms for labeling and formatting variables and procedure-generated statistics.

This chapter briefly describes the TABULATE procedure. Detailed information and examples of how to use the TABULATE procedure are given in the SAS Guide to TABULATE Processing, 1987 Edition.
**SPECIFICATIONS**

The TABULATE procedure is controlled by the following statements:

- **PROC TABULATE** `<option-list>`;
- **CLASS** class-variable-list;
- **VAR** analysis-variable-list;
- **TABLE** `<page-expression,>` row-expression, `>` column-expression `</ table-option-list>`;
- **BY** `<NOTSORTED>` `<DESCENDING>` variable-`1` `<... `<DESCENDING>` variable-n>`;
- **FORMAT** variable-list-`1` format-`1` `<...variable-list-n format-n>`;
- **FREQ** variable;
- **KEYLABEL** keyword-`1` = 'description-`1`' `<...keyword-n` = 'description-n'>;
- **LABEL** variable-`1` = 'label-`1`' `<...variable-n` = 'label-n'>;
- **WEIGHT** variable;

Table 37.1 summarizes which statements and options to use to perform specific functions with the TABULATE procedure. Use the options listed in the last column of the table with the statements listed in the middle column.

<table>
<thead>
<tr>
<th>Function</th>
<th>Statements</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Print frequencies for combinations of variable values</td>
<td>CLASS, TABLE</td>
<td></td>
</tr>
<tr>
<td>Calculate statistics other than frequency on values of a numeric variable</td>
<td>VAR, TABLE</td>
<td></td>
</tr>
<tr>
<td>Calculate statistics other than frequency on values of a numeric variable within classes of other variable values</td>
<td>CLASS, VAR, TABLE</td>
<td></td>
</tr>
<tr>
<td>Format values in table cells</td>
<td>PROC TABULATE, TABLE (F= ) FORMAT=</td>
<td></td>
</tr>
<tr>
<td>Format values of class variables in row and column headings</td>
<td>FORMAT</td>
<td></td>
</tr>
<tr>
<td>Change appearance of table</td>
<td>PROC TABULATE, TABLE</td>
<td>NOSEPS, ORDER= BOX, CONDENSE, ROW=, RTSPACE=</td>
</tr>
<tr>
<td>Change treatment of missing values</td>
<td>PROC TABULATE, TABLE</td>
<td>MISSING, MISSTEXT=, PRINTMISS</td>
</tr>
<tr>
<td>Change row and column headings</td>
<td>KEYLABEL, LABEL</td>
<td></td>
</tr>
<tr>
<td>Change default computations</td>
<td>PROC TABULATE, TABLE, FREQ, WEIGHT VARDEF= FUZZ=</td>
<td></td>
</tr>
</tbody>
</table>
The PROC TABULATE statement is always accompanied by one or more TABLE statements specifying the tables to be produced. In addition, you must use either a VAR statement or a CLASS statement or both. All variables used in the TABLE statement must be specified in either the VAR statement or the CLASS statement, but not both. The WEIGHT, FREQ, and BY statements are optional; each can be specified once for the entire TABULATE procedure step. The FORMAT, LABEL, and KEYLABEL statements are also optional; if you repeat one of these statements, the value in the last statement applies to the entire step.

**PROC TABULATE Statement**

PROC TABULATE <option-list>;

You can specify the following options in the PROC TABULATE statement:

- **DATA=SAS-data-set**
  specifies the SAS data set used by PROC TABULATE. If you omit the DATA= option, PROC TABULATE uses the SAS data set created most recently in the current job or session.

- **DEPTH=number**
  specifies the maximum depth of any dimension's crossing. (Refer to **TABLE Statement** later in this chapter for an explanation of crossings.) The default depth is 10. You may need to increase the value for the DEPTH= option, but there is no benefit to decreasing the value. The depth of a crossing refers to the number of elements, including the default statistic, that are crossed with each other within any single dimension of the TABLE statement. For example, the depth of the following TABLE statement is 3 because the default statistic must be included in the crossing:

  table a*b;

  Note: Format modifiers are also counted in the crossings. Therefore, the maximum depth of the following TABLE statement is 5:

  table a*b,x*y*z*sum*10.0;

- **FORMAT=format-name**
  specifies a default format for formatting the value in each table cell. You can use any valid SAS or user-defined format. If you omit the FORMAT= option, PROC TABULATE uses BEST12.2 as the default format. The default format is overridden by any formats specified in a TABLE statement. This option is especially useful for decreasing the number of print positions used to print a table. Refer to Chapter 5, "Controlling the Table's Appearance," in the SAS Guide to TABULATE Processing for more information on formatting output.

- **FORMCHAR<(index-list)>='string'**
- **FC<(index-list)>='string'**
  defines the characters used for constructing the table outlines and dividers. The value is a string 11 characters long defining the two bar characters, vertical and horizontal, and the 9 corner characters: upper left, upper middle, upper right, middle left, middle middle (cross), middle right, lower left, lower middle, and lower right. The default value is FORMCHAR = ' | --- | + --- '. You can substitute any character or hexadecimal string to customize the table's appearance. You can replace the entire default string by specifying a full 11-character replacement string, or you can replace selected characters by including an index list that indicates which characters are to be replaced. For example, change the four corners to asterisks by using FORMCHAR(3 5 9 11)= '****'
Specifying 11 blanks produces tables with blank outlines and dividers:

        FONNAME='1'

Refer to Chapter 5 in the SAS Guide to TABULATE Processing for more information on formatting output.

MISSING

considers missing values as valid levels for the class variables. Special missing values are considered as different level values.* A heading for each missing value is shown in the table. Unless the MISSING option is specified, PROC TABULATE does not include observations with a

missing value for one or more class variables in the analysis.

NOSEPS

eliminates horizontal separator lines from the row titles and body of the

printed table. Horizontal separator lines remain in the column title section of the table. Note that the NOSEPS option completely removes the separator lines instead of substituting blank characters, as illustrated in the FORMCHAR= option discussed earlier in this chapter.

ORDER=order

specifies the order in which headings for class variable values are displayed in each table. The possible values for order are as follows:

DATA keeps values of class variables in the order they were encountered when the input was read. Note that the order remains the same for the entire data set or BY group if a BY statement is specified.

FORMATTED orders the class values by the formatted (external)

representation of the value.

FREQ orders the headings for class variables by descending frequency count so that class values occurring in the greatest number of observations come first.

INTERNAL orders the headings in the same sequence as they would be ordered by the SORT procedure.

If you omit the ORDER= option, PROC TABULATE defaults to ORDER=INTERNAL.

VARDEF=DF

| N
| WDF
| WEIGHT | WGT

specifies the divisor to be used in the calculation of the variances. The possible values for divisor are as follows:

DF requests that the degrees of freedom (N−1) be used as the divisor.

N requests that the number of observations (N) be used.

WDF requests that the sum of the weights minus one be used.

WEIGHT | WGT requests that the sum of the weights be used.

The default is VARDEF=DF.

* Special missing values are the uppercase letters A through Z and the underscore (_), which are used to represent missing numeric values. Refer to Missing Values with Special Meanings in Chapter 2, "The DATA Step," in SAS Language: Reference, Version 6, First Edition for more information.
BY Statement

BY <NOTSORTED> <DESCENDING> variable-1
  <...<DESCENDING> variable-n>;

Use a BY statement with PROC TABULATE to obtain separate analyses on observations in groups defined by the BY variables.

Note that the page-dimension expression of a TABLE statement can have an effect similar to using a BY statement. Your input data set need not be sorted or indexed when the page-dimension expression is used. The page dimension should be used in most cases where a new page is desired for a given level of a class variable or combination of variables. Refer to TABLE Statement later in this chapter for more information on the page dimension and Comparison of BY-Group Processing to Using the Page Dimension in Chapter 3, “Details of TABULATE Processing,” in the SAS Guide to TABULATE Processing.

When a BY statement appears, the TABULATE procedure expects the input data set to be sorted in order of the BY variables or to have an appropriate index. If your input data set is not sorted in ascending order, you can do one of the following:

- Remove the BY statement and use the page dimension to produce the same effect as the BY statement.
- Use the SORT procedure with a similar BY statement to sort the data.
- If appropriate, use the BY statement options NOTSORTED or DESCENDING.
- Create an index on the BY variables you want to use. For more information on creating indexes and using the BY statement with indexed data sets, see Chapter 17, “The DATASETS Procedure.”

The following options can be specified in the BY statement:

DESCENDING
  specifies that the data set is 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. This option can appear anywhere in the BY statement.

CLASS Statement

CLASS class-variable-list;
CLASSES class-variable-list;

Use the CLASS statement to identify variables in the input data set as class variables. Class variables may have either numeric or character values. Normally each class variable has a small number of discrete values or unique levels. Continuous values for a numeric variable can be grouped into discrete levels by using the FORMAT procedure and then including a FORMAT statement in the PROC TABULATE step. Refer to Chapter 3 in the SAS Guide to TABULATE Processing for more information on creating classes.

If an observation contains missing values for any variable listed in the CLASS statement, the observation is not included in the table unless you specify the MISSING option in the PROC TABULATE statement. Note that the variables listed in the CLASS statement affect observations regardless of whether the class variable appears in a TABLE statement because the CLASS statement is in effect for the entire PROC TABULATE step.
FORMAT Statement

```
FORMAT variable-list-1 format-1 <...variable-list-n format-n>;
```

where

- `variable-list` names one or more variables to format.
- `format` specifies the format for the preceding variables.

In the TABULATE procedure, the FORMAT statement formats the values of class variables used as headings in the page, row, and column dimensions. The FORMAT statement has no effect on either analysis variables (variables specified in the VAR statement) or the content of table cells.

You can use the FORMAT statement in combination with the FORMAT procedure to group values of class variables. Keep in mind that when you use PROC FORMAT to define temporary user-written formats, you must also use the FORMAT statement in the PROC TABULATE step to associate the format with the variable.* Refer to Setting Up Useful Classes in Chapter 3 in the SAS Guide to TABULATE Processing for more information on how to use the FORMAT statement and PROC FORMAT.

FREQ Statement

```
FREQ variable;
```

The FREQ statement specifies a numeric variable in the input SAS data set whose value represents the frequency of the observation.

If you use the FREQ statement, each observation in the input data set is assumed to represent \( n \) observations, where \( n \) is the value of the FREQ variable. If the value is not an integer, the value is truncated to the integer portion. If the FREQ variable has a value less than 1, PROC TABULATE skips the observation. You can use only one variable in a FREQ statement. The FREQ statement can be used in combination with the WEIGHT statement.

KEYLABEL Statement

```
KEYLABEL keyword-1='label-1' <...keyword-n='label-n'>;
```

where

- `keyword` is one of the valid keywords for statistics discussed in STATISTICS AVAILABLE WITH PROC TABULATE later in this chapter, or the universal class variable ALL (discussed in TABLE Statement later in this chapter).
- `label` is up to 40 characters of labeling information. The label must be enclosed in single or double quotes.

PROC TABULATE uses the replacement text in the label anywhere the specified keyword is used, unless another label is assigned in the TABLE statement. The KEYLABEL statement is useful for relabeling a keyword once in a PROC TABULATE step rather than each time it occurs in a TABLE statement. Each keyword can have only one label in a particular PROC TABULATE step; if you request multiple labels for the same keyword, PROC TABULATE uses the last one specified in the step. An example of a KEYLABEL statement is

```
keylabel all='Total $'
mean='Average'
pctsum='Percent of Sum';
```

* You can also create permanent formats by assigning the format in a DATA step. In this case, you do not need the FORMAT statement in the PROC TABULATE step.
**LABEL Statement**

```
LABEL variable-1='label-1' <...variable-n='label-n'>;
```

where

- `variable` names a class or analysis variable used in a TABLE statement.
- `label` specifies a label of up to 40 characters, including blanks, for the variable. The `label` must be enclosed in single or double quotes.

The label specified for the variable replaces the name of the variable in the page, row, or column heading where the variable appears. Any number of pairs of variable names and labels can be specified in a LABEL statement.

**TABLE Statement**

```
TABLE < <page-expression, > row-expression, > column-expression
  < / table-option-list>>;
```

The TABLE statement describes the table to be printed. Every PROC TABULATE step requires at least one TABLE statement. All variables used in the TABLE statement must be specified in either the VAR statement or the CLASS statement but not both.

A TABLE statement consists of one to three dimension expressions separated by commas that can be followed by an option list. If all three dimensions are specified, the leftmost dimension defines pages, the middle dimension defines rows, and the rightmost dimension defines columns. If two dimensions are specified, the left defines rows, and the right defines columns. If a single dimension is specified, it defines columns.

The `page-expression`, `row-expression`, and `column-expression` are constructed in the same way and are referred to collectively as dimension expressions. A dimension expression is composed of elements and operators.

The elements you can use in a dimension expression are

- analysis variables. Refer to **VAR Statement** later in this chapter for more information.
- class variables. Refer to **CLASS Statement** earlier in this chapter for more information.
- the universal class variable ALL, which summarizes all of the categories for class variables in the same parenthetical group or dimension (if the variable ALL is not contained in a parenthetical group).
- keywords for statistics. Refer to **STATISTICS AVAILABLE WITH PROC TABULATE** later in this chapter for more information.
- format modifiers, which define how to format values in cells. These have the form `f=format` and must be crossed with the elements that produce the cells you want to format. See Chapter 5 in the SAS Guide to TABULATE Processing for more information.
- labels, which temporarily replace the names of variables and statistics with a label. These have the form `='label'` and affect only the variable or statistic that immediately precedes the label.
- expressions formed by combining any of these elements.
A dimension expression can have any of the following forms:

\[
\begin{align*}
\text{element*element} & \quad \text{(crossing)} \\
\text{element element} & \quad \text{(concatenation)} \\
(\text{element element}) & \quad \text{(grouping)}
\end{align*}
\]

When you cross class variables in an expression, PROC TABULATE creates categories from the combination of values of the variables. If one of the elements in the crossing is an analysis variable, the statistics for the analysis variable are calculated for the categories created by the class variables.

Concatenating elements joins information for the elements by placing the output for the second element immediately after the output for the first element.

Grouping elements causes the operator adjacent to the parenthesis to be applied to each concatenated element inside the parentheses.

Table 37.2 lists the operators and the effects they produce.

**Table 37.2 Operators Used in the TABLE Statement**

<table>
<thead>
<tr>
<th>Operator</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>, (comma)</td>
<td>separates dimensions of a table and crosses elements across dimensions</td>
</tr>
<tr>
<td>* (asterisk)</td>
<td>crosses elements within a dimension</td>
</tr>
<tr>
<td>(blank space)</td>
<td>concatenates elements in a dimension</td>
</tr>
<tr>
<td>() (parentheses)</td>
<td>group elements and associate an operator with an entire group</td>
</tr>
<tr>
<td>&lt;&gt; (brackets)</td>
<td>specify denominator definitions</td>
</tr>
<tr>
<td>= (equal sign)</td>
<td>assigns a label to a variable or statistic, or completes a format modifier</td>
</tr>
</tbody>
</table>

A TABLE statement can define only one table. Multiple TABLE statements can appear in one PROC TABULATE step, each defining a separate table. Refer to Chapter 4, “Learning to Use PROC TABULATE,” in the SAS Guide to TABULATE Processing for more information on dimension expressions and how to construct TABLE statements.

You can use the following options in the TABLE statement:

**BOX=**<value>

specifies the text to be placed in the empty box above the row titles.

The possible values are as follows:

- **_PAGE_** causes the page-dimension text to appear in the box. If the page-dimension text does not fit, it is placed in its default position, and the box is left empty.

- **'string'** causes the quoted string to appear in the box. Any name, label, or quoted string that does not fit is truncated.

- **variable** causes the name or label of a variable to appear in the box.

**CONDENSE**

prints multiple logical pages on a single physical page. PROC TABULATE prints as many complete logical pages as fit on a single printed page. This option can be used to condense multiple pages generated by the page dimension of the TABLE statement, or multiple pages caused by
tables that are too wide to fit on a single page. The CONDENSE option has no effect on the pages generated by the BY statement.

FUZZ=number
supplies a numeric value against which analysis variable values and table cell values other than frequency counts are compared to eliminate trivial values (absolute values less than the FUZZ= value) from computation and printing. A number whose absolute value is less than the FUZZ= value is treated as zero in computations and printing. The default value is the smallest representable floating-point number on the computer you are using.

MISSTEXT='text'
supplies up to 20 characters of text to print in table cells containing missing values.

PRINTMISS
specifies that row and column headings are the same for all logical pages of the table. The PRINTMISS option indicates that you want to print all values that occur for a class variable each time headings for that variable are printed. For example, consider a data set with the three observations below:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

The following TABLE statement does not produce a column for A=1 and B=3 because this combination of values does not exist in the data set:

table a*b;

If you specify the PRINTMISS option, the table includes a column for A=1 and B=3 with missing values for all table cells in the column.

If an entire logical page contains only missing values, that page does not print regardless of the PRINTMISS option. Note: By default PROC TABULATE does not suppress a row or column with all missing values when the missing values are the result of computations on analysis variables. The PRINTMISS option affects only missing rows and columns that result from combinations of class variable values that do not exist.

ROW=spacing
specifies whether all title elements in a row crossing are allotted space even when they are blank. The possible values for spacing are as follows:

- CONSTANT | CONST
  causes all row title elements to have space allotted to them, even if the title has been blanked out (for example, N=' ' in the row dimension). CONSTANT is the default.

- FLOAT
  causes the row title space to be divided equally among the nonblank title elements in the crossing.

RTSPACE=number
RTS=number
supplies an integer value that specifies the number of print positions allotted to the headings in the row dimension. Note that this space is divided equally among all levels of row headings and includes spaces used to print outlining characters for the row headings. The default value is one-fourth of the LINESIZE= value. Refer to Chapter 5 in the
SAS Guide to TABULATE Processing for more information on controlling the row title space.

**VAR Statement**

**VAR** analysis-variable-list;

**VARIABLES** analysis-variable-list;

Use the VAR statement to identify analysis variables in the input data set. Analysis variables must be numeric and can contain continuous values.

If an observation contains missing values for a variable listed in the VAR statement, the value is omitted from calculations of all statistics except N (the number of observations with nonmissing variable values) and NMISS (the number of observations with missing variable values). For example, the missing value does not increase the SUM, and it is not counted when calculating statistics such as the MEAN.

**WEIGHT Statement**

**WEIGHT** variable;

**WGT** variable;

The WEIGHT statement specifies a numeric variable in the input data set whose value is used to weight each analysis variable. Note that the WEIGHT variable value need not be an integer and does not affect the degrees of freedom.

If you specify a WEIGHT statement, PROC TABULATE uses the value of the WEIGHT variable to calculate weighted statistics. Refer to $w_i$ in the formulas in **STATISTICS AVAILABLE WITH PROC TABULATE** for information on how the WEIGHT value affects statistic calculations.

**STATISTICS AVAILABLE WITH PROC TABULATE**

A standardized set of keywords is used to refer to the descriptive statistics for PROC TABULATE. Use these keywords to request statistics in the TABLE statement.

**Keywords and Formulas**

The following notations are used where summation is over all nonmissing values:

- $x_i$: the $i$th nonmissing observation of the variable
- $w_i$: the weight associated with $x_i$ if a WEIGHT statement is specified, otherwise 1
- $n$: the number of nonmissing observations
- $\bar{x} = \Sigma w(x_i)/\Sigma w_i$
- $d = \Sigma w_i$ (if the option VARDEF=N is specified)  
  $= n - 1$ (if VARDEF=DF)  
  $= \Sigma w_i$ (if VARDEF=WEIGHT or WGT)  
  $= \Sigma w_i - 1$ (if VARDEF=WDF)
- $s^2 = \Sigma w_i(x_i - \bar{x})^2/d$
- $s = \sqrt{s^2}$
- $z_i = (x_i - \bar{x})/s$, standardized variables.