

Basic Proc Tabulate Processing

Prepared by



International SAS® Training and Consulting

Destiny Corporation – 100 Great Meadow Rd Suite 601 - Wethersfield, CT 06109-2355

Phone: (860) 721-1684 - 1-800-7TRAINING - Fax: (860) 721-9784

Web: www.destinycorp.com Email: info@destinycorp.com

Copyright 1999

Proc Tabulate

Proc Tabulate is one of the most powerful of SAS procedures. Its syntax and specification allow you to produce a wide variety of simple and complex tables. Once you have mastered the essentials of the procedure, it becomes very easy to create some very complex tables.

Raw data such as these demographic values can be displayed in tables for a better understanding of the data.

OUTPUT

OBS	AGE	GENDER	SALARY	STATUS	CHILDREN	CARS
1	29	F	8000	D	2	1
2	26	F	5600	M	1	1
3	28	F	15000	M	3	1
4	34	F	18000	M	2	1
5	34	F	0	M	2	1
6	38	F	10000	M	2	1
7	65	F	10000	M	2	1
8	32	F	0	M	2	2
9	23	F	0	M	3	2
10	56	F	30000	M	3	2
11	54	F	0	M	3	2
12	52	F	15000	M	5	2
13	60	F	13000	M	3	2
14	56	F	15000	M	0	2
15	12	F	0	S	0	0
16	16	F	0	S	0	0
17	6	F	0	S	0	0
18	22	F	13000	S	0	1
19	30	F	30000	S	0	1

One of the advantages of Proc Tabulate is that you can request statistics such as the mean, total, minimum, maximum, etc., through the Table statement. This eliminates the need for any pre-processing of the data by other statistical procedures such as Proc Means or Proc Univariate.

Demographic Data Displayed in Tabular Form

Data for January 1990	Female	Male	Total Salary
	Average Salary	Average Salary	
Age of Employee			
Child	\$0.00	\$0.00	\$0.00
Teenager	\$0.00	\$0.00	\$0.00
Young Adult	\$10,760.00	\$17,133.33	\$210,400.00
Adult	\$13,750.00	\$13,060.00	\$120,300.00
Senior Adult	\$17,000.00	No Data	\$68,000.00
Overall Average and Salary Total	\$10,980.95	\$12,007.14	\$398,700.00

All Figures are Rounded

Steps to Produce a Table

1. Select data set
2. Define row and column variables
3. Define numeric for statistical calculation
4. Define physical form of table

Types of Variables

Pre-Defined Variables

As well as the data set name, Proc Tabulate requires that you pre-define variables as either:
CLASS or **VAR**

...on the appropriate statement.

Either a CLASS statement or a VAR statement or both is required.

CLASS

	1	2	3	4
female	\$100	\$150	\$200	\$250
male	\$400	\$350	\$200	\$200

VAR

Example

PROGRAM EDITOR

```
title; footnote;  
proc tabulate data=saved.demograf;  
  class gender cars;  
  var salary;  
  table gender,cars*salary;  
run;
```

OUTPUT

	CARS			
	0	1	2	3
	SALARY	SALARY	SALARY	SALARY
	SUM	SUM	SUM	SUM
GENDER				
F	0.00	157600.00	73000.00	.
M	12000.00	85100.00	31000.00	40000.00

Which Variables to Choose?

Which variables could be CLASS variables?

OUTPUT

OBS	AGE	GENDER	SALARY	STATUS	CHILDREN	CARS
1	11	M	0	S	0	0
2	2	M	0	S	0	0
3	12	F	0	S	0	0
4	16	F	0	S	0	0
5	6	F	0	S	0	0
6	14	M	0	S	0	0
7	23	M	10000	S	0	1
8	22	F	13000	S	0	1
9	26	F	5600	M	1	1
10	28	M	12000	M	2	0
11	29	F	8000	D	2	1
12	28	F	15000	M	3	1
ETC....						
34	46	F	30000	W	3	1
35	65	F	10000	M	2	1

Would this help for AGE?

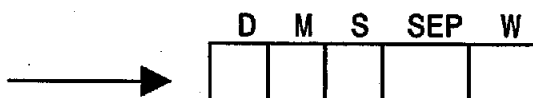
PROGRAM EDITOR

```
proc format;  
  value agefmt  
    0-12='Child'  
    13-19='Teenager'  
    20-35='Young Adult'  
    36-55='Adult'  
    55-65='Senior Adult'  
    66-high='Elderly';  
run;
```

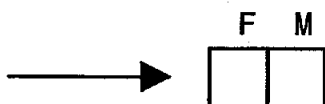
AGE could be used as a CLASS variable if a FORMAT AGE AGEFMT.; statement was included.

Examples

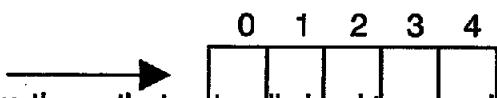
CLASS status;



CLASS gender;



CLASS cars;



Class variables are those that naturally lead to grouping or classifying in some way.

Var statement

Analysis variables

These are variables that are used in calculations, in other words, they are the numbers in the boxes of a table. The Var statement is used to specify the Analysis variables.

PROGRAM EDITOR

```
proc tabulate data=rwdata.demograf; <<< INFORMATION
  class gender cars; <<< INFORMATION
  var salary; <<< INFORMATION
  table gender,cars*salary; <<<<<<< ACTION!
run;
```

	1	2	3	4
female	\$100	\$150	\$200	\$250
male	\$400	\$350	\$200	\$200

VAR

This is the variable that is used in calculations, in other words, they are the numbers in the boxes of a table.

The analysis variable is always numeric.

Using Age as Analysis

PROGRAM EDITOR

```
proc tabulate data=saved.demograf;  
  class status gender;  
  var age;  
  table status,gender*age;  
run;
```

OUTPUT

	GENDER	
	F	M
	AGE	AGE
	SUM	SUM
STATUS		
D	29.00	69.00
M	558.00	220.00
S	86.00	83.00
SEP	23.00	55.00
W	46.00	.

In this table, Age is being used as an Analysis variable. We are calculating the average age for females and males.

Types of Tables

The Table Statement

The Table statement defines several aspects:

- the variables that are to be tabled
- the physical form of the table
- where the variables are placed in the table
- Enhancement options such as formats, labels, extra text, etc.
- Any totals required

The table statement can run over several physical lines of code, simply because of the vast amount of information, options and variable names that you can request.

Operators

The Table statement contains variables, options and operators. Operators are the characters that lie between the variable names:

- **Asterisk** * Nest, Cross or Subgroup
- **Comma** , Add a New Dimension
- **Space** Join together, concatenate
- **Parentheses** () Group or Order
- **Brackets** <> Denominator Definition
- **Equals** = Request Label for Variable

By placing an operator between the variable names, you control the physical appearance of the table.

Syntax of the Table Statement

```
TABLE      VARIABLE = 'variable label'
           * statistic = 'statistic label'
           * format = formatname.
           optional-all-specification

           OPERATOR

           VARIABLE = 'variable label'
           * statistic = 'statistic label'
           * format = formatname.
           optional-all-specification

           OPERATOR

           VARIABLE = 'variable label'
           * statistic = 'statistic label'
           * format = formatname.
           optional-all-specification

           /
           general table options
           ;
```

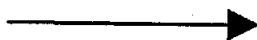
Examples

1. TABLE gender;
2. TABLE gender,status*salary;
3. TABLE age='Students Age' , gender='Gender' ;
4. TABLE gender all , salary * f=pound. ;

Table Examples

CLASS statement used

VAR statement left out



number of observations
shown in cells

TABLE statement specifies one variable
no operators
no totals
no formats
no labels

PROGRAM EDITOR

```
proc tabulate data=saved.demograf;  
  class gender;  
  table gender;  
run;
```

OUTPUT

GENDER	
F	M
N	N
21.00	14.00

Defaults:

variable names

format of values

statistics

columns sizes

Other Variables

CLASS statement used

VAR statement left out → number of observations shown in cells

TABLE statement specifies two variables
one operator → the blank operator
no totals
no formats
no labels

PROGRAM EDITOR

```
title; footnote;  
proc tabulate data=saved.demograf;  
  class status cars;  
  table status cars;  
run;
```

STATUS				
D	M	S	SEP	W
N	N	N	N	N
3.00	19.00	10.00	2.00	1.00

CARS			
0	1	2	3
N	N	N	N
7.00	18.00	9.00	1.00

The Table statement has created a table of both variables and concatenated them together because of the space between the variable names.

Different Operators

Comma

CLASS statement used

VAR statement left out

number of observations
shown in cells

TABLE statement specifies two variables

one operator

no totals

no formats

the comma operator

The Comma Operator gives us a new dimension:

PROGRAM EDITOR

```
proc tabulate data=saved.demograf;
  class gender status;
  table gender, status;
run;
```

OUTPUT

	STATUS			
	D	M	S	SEP
	N	N	N	N
GENDER				
F	1.00	13.00	5.00	1.00
M	2.00	6.00	5.00	1.00

	STATUS
	W
	N
GENDER	
F	1.00
M	.

Asterisk

CLASS statement used

VAR statement left out

number of observations
shown in cells

TABLE statement specifies two variables
one operator
no totals
no formats

the asterisk operator

The Asterisk Operator gives us a nesting:

PROGRAM EDITOR

```
proc tabulate data=saved.demograf;  
  class gender status;  
  table gender*status;  
run;
```

GENDER				
F				
STATUS				
D	M	S	SEP	W
N	N	N	N	N
1.00	13.00	5.00	1.00	1.00

GENDER			
M			
STATUS			
D	M	S	SEP
N	N	N	N
2.00	6.00	5.00	1.00

Blank

CLASS statement used

VAR statement left out

number of observations
shown in cells

TABLE statement specifies two variables

one operator
no totals
no formats

the blank operator

The Blank Operator gives us concatenation:

```
proc tabulate data=saved.demograf;  
  class gender status;  
  table gender status;  
run;
```

GENDER		STATUS		
F	M	D	M	S
N	N	N	N	N
21.00	14.00	3.00	19.00	10.00

STATUS	
SEP	W
N	N
2.00	1.00

3 Dimensional Tables

The addition of another COMMA operator gives 3 dimensions:

CLASS statement used

VAR statement left out → number of observations
shown in cells

TABLE statement specifies two variables
two operators → the comma operators
no totals
no formats

PROGRAM EDITOR

```
proc tabulate data=saved.demograf;  
  class gender status cars;  
  table gender,status,cars;  
run;
```

OUTPUT

GENDER F

	CARS		
	0	1	2
	N	N	N
STATUS			
D	.	1.00	.
M	.	6.00	7.00
S	3.00	2.00	.
SEP	.	1.00	.
W	.	1.00	.

GENDER M

	CARS			
	0	1	2	3
	N	N	N	N
STATUS				
D	.	2.00	.	.
M	1.00	2.00	2.00	1.00
S	3.00	2.00	.	.
SEP	.	1.00	.	.

The third dimension has been added by taking the first variable in the list: Gender, and controlling for it. In other words, we take the first value of Gender, 'F', and produce a table of Status by Cars. Then take Gender='M' and produce Status by Cars again.

Grouping Variables

Parentheses can produce radically different tables depending on which variables you group together.

CLASS statement used

VAR statement left out → number of observations shown in cells

TABLE statement specifies three variables
three operators → parenthesis
blank
asterisk

no totals
no formats

format=4.
used to reduce
column width

PROGRAM EDITOR

```
proc tabulate data=saved.demograf format=4.;
  class gender status cars;
  table (gender status)*cars;
  where cars > 0;
run;
```

OUTPUT

GENDER					STATUS						
F		M			D	M			S	SEP	W
CARS		CARS			CARS	CARS			CARS	CARS	CARS
1	2	1	2	3	1	1	2	3	1	1	1
N	N	N	N	N	N	N	N	N	N	N	N
11	7	7	2	1	3	8	9	1	4	2	1

Var Statement

The previous examples all contained Class statements and Tables statements. As we stated before, any variable that is specified on the Table statement must have been specified on the Class statement. What then is our Analysis variable?

Proc Tabulate gives the default statistic, N, or the number of observations in the group, if no Var statement is specified. Let's add in a Var statement and examine the effect:

CLASS statement used

VAR statement used value of salary
 shown in cells

TABLE statement specifies one variable
 no operators
 no totals
 no formats

PROGRAM EDITOR

```
proc tabulate data=saved.demograf;  
  class gender;  
  var salary;  
  table gender;  
run;
```

OUTPUT

GENDER	
F	M
N	N
21.00	14.00

Why is there no effect?

Use the analysis variable on the TABLE statement

The Var statement specifies a variable whose values will be placed in the data boxes in the table. This is the analysis variable, and must be a numeric.

CLASS statement used

VAR statement used

value of salary
shown in cells

TABLE statement specifies two variables - gender and salary

no operators

no totals

no formats

PROGRAM EDITOR

```
proc tabulate data=saved.demograf;  
  class gender;  
  var salary;  
  table gender*salary;  
run;
```

OUTPUT

GENDER	
F	M
SALARY	SALARY
SUM	SUM
230600.00	168100.00

Default Statistic - SUM

Instead of the number of observations in the particular categories, we now have the total (sum) of Salary.

Totals - the ALL option

As well as the physical form of the table and the variable names, the Table statement defines how totals are displayed. This is requested by the ALL option. It is sometimes easier to regard the All option as a separate "variable" on the Table statement.

This option allows totals to be requested at various places throughout the table, and behaves like another "variable" name in that it is affected by the operators of blank, comma, asterisk and parentheses.

CLASS statement used

VAR statement used

value of salary
shown in cells

TABLE statement specifies two variables - gender and salary
* and blank operators
ALL specified (but for which variable?)
no formats

PROGRAM EDITOR

```
proc tabulate data=saved.demograf;  
  class gender;  
  var salary;  
  table gender*salary all;  
run;
```

OUTPUT

GENDER		ALL
F	M	
SALARY	SALARY	
SUM	SUM	N
230600.00	168100.00	35.00

The total has been added to the table, but perhaps not as we expected.

Crossing ALL

The All option acts as if it were another variable to be concatenated to the side of the table. We require the total salary for Gender, in which case we have to nest All under Salary:

CLASS statement used

VAR statement used

value of salary
shown in cells

TABLE statement specifies two variables - gender and salary

* and blank operators

ALL specified (for SALARY)

no formats

PROGRAM EDITOR

```
proc tabulate data=saved.demograf;  
  class gender;  
  var salary;  
  table gender*salary salary*all;  
run;
```

OUTPUT

GENDER		SALARY
F	M	
SALARY	SALARY	SUM
SUM	SUM	ALL
230600.00	168100.00	398700.00

Row and Column Totals

Row totals and overall totals can be generated by the All option:

CLASS statement used

VAR statement not used  numbers of observations shown in cells

TABLE statement specifies two variables - status and gender
blank operators
ALL specified (for status and gender)
no formats

PROGRAM EDITOR

```
proc tabulate data=saved.demograf;  
  class status gender;  
  table status all,gender all;  
run;
```

OUTPUT

	GENDER		ALL
	F	M	
	N	N	N
STATUS			
D	1.00	2.00	3.00
M	13.00	6.00	19.00
S	5.00	5.00	10.00
SEP	1.00	1.00	2.00
W	1.00	.	1.00
ALL	21.00	14.00	35.00