

What's Your `_TYPE_`? Finding the `CLASS` You Want In `PROC SUMMARY`

Frank Ferriola, Pacioli Companies, Highlands Ranch, CO

ABSTRACT

When using the `CLASS` statement in your `PROC SUMMARY`, SAS will calculate every possible combination of the `CLASS` variables. By understanding the way SAS summarizes, you can find a simple way to determine which summaries you really want to keep and eliminating the others.

CAVEATS

For simplicity, this paper discusses the use of `CLASS` and `_TYPE_` in the context of `PROC SUMMARY`. The rules apply for any SAS procedure, which uses `CLASS`, such as `PROC MEANS`. SAS also includes a different way of doing the same thing in V8 using a `CHARTYPE` option in the `PROC`, which this paper does not address.

INTRODUCTION

I have always been fascinated by number patterns. When I was about 8 years old, my father showed me a "Magic Square", a 3 x 3 or odd numbered square in which you could place the numbers 1 through 9 and have all rows, across, down and diagonally equal the same number. It's easy to memorize the answer:

8	1	6
3	5	7
4	9	2

But how do you memorize a 5x5, 7x7 or even a 99x99 square? There are many people who can spout complicated algorithms and formulas to compute the answer, however, my father showed me a pattern with simple rules to fill out the square under any circumstances. This makes the solution easy if not impressive. I'd tell you the pattern but then you'd probably stop reading the paper and start trying larger squares.

In the same way, SAS uses a simple binary pattern to summarize the `CLASS` statement in `PROC SUMMARY` and other `PROCS`. By knowing this pattern, you can simplify your output to just the summaries you need cutting down on paper and processing time.

If the user is looking only for the summary with all the `CLASS` variables this can be done using the `NWAY` option in the `PROC SUMMARY` statement:

```
PROC SUMMARY NWAY;  
  CLASS A B C D;  
  VAR TOTAL;  
  OUTPUT OUT =NWAYSUMM  
  SUM=SUMTOTAL;  
RUN;
```

You will also get the same result with the `BY` statement in a summary preceded by a `PROC SORT`;

```
PROC SORT;  
  BY A B C D;  
RUN;  
  
PROC SUMMARY;  
  BY A B C D;  
  VAR TOTAL;  
  OUTPUT OUT =BYSUMM  
  SUM=SUMTOTAL;  
RUN;
```

Many times you need more than one summary. On these occasions, you can easily do one `PROC SUMMARY` and retain only the summaries you need from that one procedure.

UNDERSTANDING `_TYPE_`

To understand how `_TYPE_` works in SAS let's use a simple `PROC SUMMARY`

```
PROC SUMMARY;  
  CLASS A;  
  VAR TOTAL;  
  OUTPUT OUT = ONEVAR  
  SUM=SUMTOTAL;  
RUN;
```

This summary will produce 2 summaries and assign `_TYPE_` as follows:

```
_TYPE_ = 0 All records  
_TYPE_ = 1 Records summarized by A.
```

Now we can expand it by one variable.

```
PROC SUMMARY;  
  CLASS A B;  
  VAR TOTAL;  
  OUTPUT OUT = TWOVAR  
  SUM=SUMTOTAL;  
RUN;
```

In this way we double the number of `_TYPE_'s`:

```
_TYPE_ = 0 All records  
_TYPE_ = 1 Records summarized by B only.  
_TYPE_ = 2 Records summarized by A only.  
_TYPE_ = 3 Records summarized by both A and B.
```

Now add a 3rd Variable:

```
PROC SUMMARY;  
  CLASS A B C;  
  VAR TOTAL;  
  OUTPUT OUT = THREEVAR  
  SUM=SUMTOTAL;  
RUN;
```

Once again we double the number of `_TYPE_'s`:

```
_TYPE_ = 0 All records  
_TYPE_ = 1 Records summarized by C only.  
_TYPE_ = 2 Records summarized by B only.  
_TYPE_ = 3 Records summarized by both B and C.  
_TYPE_ = 4 Records summarized by A only.  
_TYPE_ = 5 Records summarized by A and C only.  
_TYPE_ = 6 Records summarized by A and B only.  
_TYPE_ = 7 Records summarized by A and B and C.
```

If you're following along you should start seeing the pattern. Let's add one more variable:

```
PROC SUMMARY;  
  CLASS A B C D;  
  VAR TOTAL;  
  OUTPUT OUT = FOURVAR  
  SUM=SUMTOTAL;
```

RUN;

Once again we double the number of `_TYPE_`'s:

`_TYPE_ = 0` All records
`_TYPE_ = 1` Records summarized by D only.
`_TYPE_ = 2` Records summarized by C only.
`_TYPE_ = 3` Records summarized by both C and D.
`_TYPE_ = 4` Records summarized by B only.
`_TYPE_ = 5` Records summarized by B and D only.
`_TYPE_ = 6` Records summarized by B and C only.
`_TYPE_ = 7` Records summarized by B and C and D.
`_TYPE_ = 8` Records summarized by A only.
`_TYPE_ = 9` Records summarized by A and D.
`_TYPE_ = 10` Records summarized by A and C.
`_TYPE_ = 11` Records summarized by A, C, and D.
`_TYPE_ = 12` Records summarized by A and B.
`_TYPE_ = 13` Records summarized by A, B, and D.
`_TYPE_ = 14` Records summarized by A, B and C.
`_TYPE_ = 15` Records summarized by A, B, C and D.

From this pattern we can determine a formula:

1 Variable = 2 Summaries = 2^1
2 Variables = 4 Summaries = 2^2
3 Variables = 8 Summaries = 2^3
4 Variables = 16 Summaries = 2^4
5 Variables = 32 Summaries = 2^5
6 Variables = 64 Summaries = 2^6

or $S=2^n$

where S= # of Summaries and n= number of variables in the CLASS Statement.

We also find that PROC SUMMARY generates the summary variables from right to left in the CLASS statement.

So, how do we tell which summaries we want?
Let's look at the following example:

```
PROC SUMMARY;  
  CLASS A B C D E F;  
  VAR TOTAL;  
  OUTPUT OUT = WHICHSUM  
  SUM=SUMTOTAL;  
RUN;
```

Now we want to find the summary `_TYPE_` number for variables B, D & F. We can use the formula above to insert a comment line to help us:

```
PROC SUMMARY;  
* _TYPE_ 32 16 8 4 2 1;  
  CLASS A B C D E F;  
  VAR TOTAL;  
  OUTPUT OUT = WHICHSUM  
  SUM=SUMTOTAL;  
RUN;
```

Remember PROC SUMMARY will summarize CLASS from right to left, so we assign the value of the variable by itself F=1, E=2, D=4, C=8, B=16 A=32. Also remember that using no variables (total of data) is always 0. Using all variables will be 2 times the left most variable - 1. In this case 63.

So how do we determine the summary for B D & F. From the comment line we take
(B=16) + (D=4) + (F=1)

or $16 + 4 + 1 = 21$

So `_TYPE_ = 21` would be summarized by B,D&F.

Similarly, A,C,E,F would be equal $32 + 16 + 2 + 1 = 51$.

Once you decide which `_TYPE_`'s you want you can print the shortened list with a where clause.

```
PROC PRINT DATA=WHICHSUM;  
  Where _TYPE_ in (0,21,51,63);  
RUN;
```

AN EXAMPLE OF PROC SUMMARY

Let's look at some real data. Table 1 on the next page shows PROC SUMMARY output from fantasy football league data created with the following code.

```
PROC SUMMARY DATA=CFFL;  
  CLASS WEEK CFFL_TEAM POSITION;  
  VAR Points;  
  OUTPUT OUT=BASESUMM SUM=TOTPTS;  
RUN;
```

First look at the `_TYPE_` (5th) column. The first line shows when `_TYPE_` equals 0 we have a total of 1,404 records and Totpts=14,814. `_TYPE_ = 1` shows 6 lines by position. The total of all 6 of these observations would equal 1,404 records and 14,814 Totpts. All of `_TYPE_ = 2` would also add up to the same amount. This would have 10 observations by CFFL_TEAM.

The total summaries for this example would be 2^n or 2^3 or 8.

In Table 2 we have added two more variables CFFL_OPP and LOCATION. This now gives us 2^5 or 32 summaries. If we want to find the summaries we can use the following code:

```
PROC SUMMARY DATA=CFFL;  
* _TYPE_ 16 8 4 2 1;  
  CLASS WEEK CFFL_TEAM CFFL_OPP LOCATION POSITION;  
  VAR Points;  
  OUTPUT OUT=BASESUMM SUM=TOTPTS;  
RUN;
```

Thus, in this summary, we can associate Week with the value (16), CFFL_TEAM (8) CFFL_OPP (4) Location (2) Position (1).

- 1) The summary by Week (16) and Location (2) would be 18.
- 2) The summary by CFFL_TEAM (8) CFFL_OPP (4) and Position (1) would be 13.
- 3) The summary by Week (16) CFFL_TEAM (8) and Position (1) would be 25.
- 4) The summary by all variables Week (16) + CFFL_TEAM (8) CFFL_OPP (4) Location (2) Position (1) would be 31.

CONCLUSION

Understanding PROC SUMMARY and how it works can make it easier when summarizing large data sets. By knowing the number of summaries your PROC SUMMARY creates you can determine which summaries you want to keep and which ones will be of no use to you.

The formula for total summaries is $S=2^n$ where S= number of summaries and n=number of class variables.

This method will also work with PROC MEANS and other SAS Procedures which use the CLASS Statement.

Table 2

SAS

File Edit View Tools Solutions Window Help

Output - (Untitled)

The SAS System 23:11 Thursday, January 13, 2000 27

Obs	Week	CFFL_Team	cffl_opp	location	Position	_TYPE_	_FREQ_	TOTPTS
2192	1			Away		18	38	294
2193	1			Home		18	52	591
2194	2			Away		18	47	383
2195	2			Home		18	43	467
2196	3			Away		18	45	412
2197	3			Home		18	45	464
2198	4			Away		18	48	478
2199	4			Home		18	42	456
2200	5			Away		18	45	440
2201	5			Home		18	45	514
2202	6			Away		18	48	475
2203	6			Home		18	42	350
2204	7			Away		18	40	400
2205	7			Home		18	50	553
2206	8			Away		18	46	526
2207	8			Home		18	44	338
2208	9			Away		18	40	465
2209	9			Home		18	50	535
2210	10			Away		18	47	552
2211	10			Home		18	43	466
2212	11			Away		18	50	555
2213	11			Home		18	40	394
2214	12			Away		18	39	414
2215	12			Home		18	51	543
2216	13			Away		18	44	448
2217	13			Home		18	46	544
2218	14			Away		18	45	454
2219	14			Home		18	45	483
2220	15			Away		18	34	308
2221	15			Home		18	38	625
2222	16			Away		18	47	470
2223	16			Home		18	25	357

Output - (Untitled) Log - (Untitled) Editor - Untitled1 * cffl1.sas *

C:\My Documents\My SAS Files\V8

Table 1

SAS

File Edit View Tools Solutions Window Help

Output - (Untitled)

The SAS System 23:11 Thursday, January 13, 2000 1

Obs	Week	CFFL_Team	Position	_TYPE_	_FREQ_	TOTPTS
1	.			0	1404	14814
2	.		DEF	1	156	1497
3	.		K	1	156	1158
4	.		QB	1	156	2604
5	.		RB	1	312	3848
6	.		TE	1	156	858
7	.		WR	1	468	4849
8	.	'85 Bears		2	144	1663
9	.	Buncha Munchers		2	144	1915
10	.	Cajun Voo Doo		2	144	1194
11	.	Frozen Tundra		2	144	1476
12	.	Homer's Heroes		2	144	1541
13	.	Hurricane Grizzlies		2	126	1097
14	.	Meadow Muffins		2	144	1643
15	.	Pencil Necked Geek		2	144	1721
16	.	Scream'n' Eagles		2	126	1174
17	.	The Skirts		2	144	1390
18	.	'85 Bears	DEF	3	16	151
19	.	'85 Bears	K	3	16	122
20	.	'85 Bears	QB	3	16	240
21	.	'85 Bears	RB	3	32	485
22	.	'85 Bears	TE	3	16	96
23	.	'85 Bears	WR	3	48	569
24	.	Buncha Munchers	DEF	3	16	172
25	.	Buncha Munchers	K	3	16	152
26	.	Buncha Munchers	QB	3	16	258
27	.	Buncha Munchers	RB	3	32	605
28	.	Buncha Munchers	TE	3	16	163
29	.	Buncha Munchers	WR	3	48	565
30	.	Cajun Voo Doo	DEF	3	16	168
31	.	Cajun Voo Doo	K	3	16	112
32	.	Cajun Voo Doo	QB	3	16	257
33	.	Cajun Voo Doo	RB	3	32	263
34	.	Cajun Voo Doo	TE	3	16	61
35	.	Cajun Voo Doo	WR	3	48	333
36	.	Frozen Tundra	DEF	3	16	142
37	.	Frozen Tundra	K	3	16	105

Output - (Untitled) Log - (Untitled) Editor - Untitled1 * cffl1.sas *

C:\My Documents\My SAS Files\V8

ACKNOWLEDGEMENTS

I would like to thank the following people:

Doug Felton for encouraging me to write this paper;

Nikki Carroll and Denver SAS Users Group for allowing me to present it;

Joe Leoni of Blue Cross Blue Shield of Delaware who led me to SAS in 1984;

My father, Frank Ferriola, Sr. who encouraged me as a child in my fascination with numbers and logic.

REFERENCES

SAS is a registered trademark of SAS Institute, Inc in the USA and other countries.

SAS Language and Procedures, Version 6, SAS Institute, Inc.

CONTACT INFORMATION

Frank Ferriola
Pacioli Companies
3110 W. Deer Creek Dr.
Highlands Ranch, CO 80129
Email: faferriola@yahoo.com

MAGIC SQUARE SOLUTION

For a copy of the Magic Square Solution see
<http://www.elkcrossing.com/presentations/magicsquare.html>