

Paper TU07

MOVING FORWARD with FORMATS

David Maddox, Regions Financial Corporation, Birmingham, AL

Abstract

FORMATS provide an instruction or template that SAS® uses to output data values. They are used primarily to control the written appearance of data values. But, FORMATS have other uses as well. This paper will feature some of those additional uses, including table lookups and traffic lighting.

Introduction

FORMATS are a basic part of the SAS programmer's toolkit. As best I can tell, they have been around since the early days of SAS. The need to control the appearance of output generally encourages novice programmers to gain some level of familiarity with FORMATS. Another basic use of the FORMAT facility is to classify values in lieu of writing a series of conditional statements. That capability is a well-documented and useful feature. The purpose of this paper is to move a little beyond the most basic use of FORMATS to present several less obvious uses of them. The references and Recommended Reading section will also provide what I hope will be a useful resource for those looking to add to their level of FORMAT expertise.

Table Look-Ups

FORMATS are very useful as a look-up facility. A simple example would be to check a data set against a list and, in effect select observations from the data set based on the list. This capability is extremely well documented, and there are a number of excellent user group papers on this topic. I mention it here as we begin to stretch a little out of the realm of the FORMAT as just a presentation device. An alternative to using a FORMAT as a look-up tool would be match-merging or PROC SQL joins. While hand-coding a PROC FORMAT value statement can be a tedious task, the CNTLIN option can be used to load an entire data set to create a FORMAT that includes many entries. The process of using a FORMAT as a look-up device can be easily demonstrated with just a few observations and a modest FORMAT. In this example, the "list" is a character FORMAT called \$metro. The "larger" data set is a listing of telephone subscribers and their monthly bills.

Obs	npa	nxx	line	revenue
1	404	423	AAAA	50.00
2	229	543	XXXX	75.00
3	912	333	BBBB	25.00
4	912	767	CCCC	31.75
5	678	221	DDDD	46.00
6	478	332	EEEE	34.50
7	770	681	FFFF	21.70
8	706	728	GGGG	45.80
9	229	476	HHHH	31.20

The following code produces the "list" which is used in the subsequent data step to subset the data based on the resolution of the PUT function to select the subscribers that had numbers in one of the metro area codes.

```

PROC FORMAT;
    value $metro '404','470','678','770'='me';
run;

data metro;
set subscribers;
if put(npa,$metro.)='me';      /* The format is used as the argument of a PUT
                                function to select the customers on the list
                                that had metro numbers. */
run;
proc print data=metro;
title 'Metro Atlanta';
run;

```

The execution of the code would yield the following:

Obs	npa	nxx	line	revenue
1	404	423	AAAA	50.0
2	678	221	DDDD	46.0
3	770	681	FFFF	21.7

Using FORMATS as look-up table is very efficient in comparison to match-merging even with indexing. It has been estimated through various bench-marking tests that this method has been proven to reduce CPU usage by 70%-80% (Eason, 2005). Several of the papers cited have excellent discussions of this method.

Traffic Lighting

Moving along just a little more, we discover another use for FORMATS in the area of traffic lighting. This type of exception reporting provides a dramatic improvement in the visual effect of making certain values stand out. This simple example illustrates the method of using FORMATS to distinguish between low, medium, and higher subscriber bills. A numeric user-defined FORMAT is first defined and then passed to the PROC PRINT via the style option on the REVENUE variable.

```

proc format;
value larg  low - 24.00 ='Green'
            23.00 - 49.00='Blue'
            50.00 - high='Red';
Run;

proc print data=subscribers;
var npa nxx line;
var revenue / style={foreground=larg.};
run;

```

This example uses PROC PRINT which primarily allows you to traffic light at the variable level. Both PROC TABULATE and PROC REPORT allow traffic lighting in a similar manner. The key elements are the user-defined FORMATS and the use of the Style option. The “foreground” specification controls the font color. See the excellent poster discussion of the various options that can be employed to traffic light using the ODS and FORMAT facilities (Hadden, 2006).

The results are as follows:

Obs	npa	nxx	line	revenue
1	404	423	AAAA	50.00
2	229	543	XXXX	75.00
3	912	333	BBBB	25.00
4	912	767	CCCC	31.75
5	678	221	DDDD	46.00
6	478	332	EEEE	34.50
7	770	681	FFFF	21.70
8	706	728	GGGG	45.80
9	229	476	HHHH	31.20

The specification of "background" in the VAR statement above results in the following:

Obs	npa	nxx	line	revenue
1	404	423	AAAA	50.00
2	229	543	XXXX	75.00
3	912	333	BBBB	25.00
4	912	767	CCCC	31.75
5	678	221	DDDD	46.00
6	478	332	EEEE	34.50
7	770	681	FFFF	21.70
8	706	728	GGGG	45.80
9	229	476	HHHH	31.20

It is important to note that in Version 8.2 that the presence of a permanent FORMAT for the traffic lighting variable prevented the traffic lighting effect. This conflict was restricted to PROC PRINT. The workaround strategy was to either remove the permanent FORMAT or use PROC REPORT. This problem was fixed in Version 9.1.

Conditional Code Styles

Various traffic lighting schemes easily lead us into an additional area of conditional code styles. Performing similar functions in PROC REPORT requires the use of a COMPUTE block with an IF statement. However, the style option can be used to accomplish the same result.

```
proc report data=subscribers nowd;
  columns npa nxx line revenue;
  define npa / 'NPA';
  define nxx / 'NXX';
  define line / 'LINE';
  define revenue / format=5.2 style=[foreground=larg.] 'REVENUE';
run;
```

The excellent discussion in (Levin, 2005) discusses some of these methods and presents a useful taxonomy of FORMAT uses.

Nested FORMATS

A sometimes overlooked feature of FORMATS is that they can be “nested” which means that one or more labels can be another FORMAT. That FORMAT can either be user-defined or supplied by SAS. The following user-defined FORMAT illustrates this capability.

```
PROC FORMAT;
  value $metro '404'='City'
              '470','678','770'='Metro'
              '229','912'='Outstate'
  other=[$inter.];
  value $inter '803'='SC'
              '423'='TN'
              '205'='AL'
  other='Rest of Country';
run;
```

run;

Using the \$metro FORMAT to print the account database produces the following output:

Obs	npa	NPA	nxx	line	revenue
1	City	404	423	AAAA	50.0
2	Outstate	229	543	XXXX	75.0
3	Outstate	912	333	BBBB	25.0
4	Metro	678	221	DDDD	46.0
5	Rest of Country	478	332	EEEE	34.5
6	Metro	770	681	FFFF	21.7
7	Rest of Country	706	728	GGGG	45.8
8	AL	205	768	xxxx	25.0
9	Rest of Country	202	321	xxxx	140.0
10	Rest of Country	505	421	xxxx	76.0

Conclusion

The purpose of this paper was to discuss just a few of the alternative uses of FORMATS beyond simple display. There are many more, and these are discussed in many of the papers cited below. As is often the case with SAS, facilities developed for one purpose can often be used for other purposes as well. FORMATS can really help SAS programmers to write some elegant and concise code that does a lot with just a few statements. They should definitely be a part of your SAS repertoire.

References

- Eason, Jenine. 2005 "Proc Format, a Speedy Alternative to Sort/Merge." *Proceedings of the Thirtieth Annual SAS Users Group International*, Philadelphia, PA.
- Hadden, Louise. 2006 "Stop! Wait! Go!: See What Traffic-Lighting Can Do For You." *Proceedings of the Thirty-First Annual SAS Users Group International*, San Francisco, CA.
- Levin, Lois. 2005 "PROC FORMAT – Not Just Another Pretty Face." *Proceedings of the Thirtieth Annual SAS Users Group International*, Philadelphia, PA.
- Lund, Pete. 2001 "More Than Just Value: A Look Into the Depths of PROC FORMAT." *Proceedings of the Twenty-sixth Annual SAS Users Group International*, Long Beach, CA.
- Maddox, David. 2006 "FORMAT FESTIVAL – An Introduction to SAS FORMATS and INFORMATS." *Proceedings of the SouthEast SAS Users Group 2006 Conference*, Atlanta, GA.

Recommended Reading

- Droogendyk, Harry. 2004 "(In)Formats (In)Decently Exposed." *Proceedings of the Twenty-ninth Annual SAS Users Group International*, Montréal, Québec, CANADA
- Hall, Diane. 2006 "Dynamic Table Numbering – A Straightforward Approach." *Proceedings of the Thirty-First Annual SAS Users Group International*, San Francisco, CA.
- Jungvig, Lara K. 2003 "The Use of Formats, Concatenate, and Sum for Reporting on "Check All That Apply" Variables." *Proceedings of the Twenty-Eighth Annual SAS Users Group International*, Seattle, WA.
- Karp, Andrew. 2005 "My Friend the SAS® Format." *Proceedings of the Thirtieth Annual SAS Users Group International*, Philadelphia, PA.
- Kupronis, Ben. 2006 "PROC FORMAT: An Analyst's Buddy." *Proceedings of the Thirty-First Annual SAS Users Group International*, San Francisco, CA.
- Scerbo, Marge. 2007 "Win the Pennant! Use PROC FORMAT." *Proceedings of SAS Global Forum 2007*, Orlando, FL.
- Shoemaker, Jack N. "PROC FORMAT in Action." *Proceedings of the Twenty-Seventh Annual SAS Users Group International*, Orlando, FL
- Williams, Christiana. 2007 "Using PROC FORMAT for Data Validation and Cleanup." *Proceedings of SAS Global Forum 2007*, Orlando, FL.

Contact Information

Your comments and questions are valued and encouraged. Contact the author at:

David Maddox
Regions Financial Corporation
RCS-1
2050 Parkway Office Circle
Birmingham, AL 35244
Work Phone: 205-560-6339
FAX: 205-560-3525
Email: Garland.MaddoxJr@Regions.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.