

Using ODS to Create User Defined Reports Without PROC TEMPLATE

Nicolle Mode, HealthPartners Research Foundation, Minneapolis, MN

ABSTRACT

The Output Delivery System (ODS) for SAS version 8.2 offers users the ability to create high quality reports in a variety of formats, from a variety of procedures. There are currently many well written and comprehensive tutorials for using ODS, especially from the SUGI conferences. These advanced level tutorials include many options for creating and customizing output including the array of style choices within PROC TEMPLATE. Unfortunately PROC TEMPLATE can be daunting to a novice user, and the majority of examples are written with HTML output in mind. In this presentation I outline several ways to customize reports without the use of PROC TEMPLATE with postscript, pdf, and rtf output. Specifically I review several proc report options, in-line formatting (for printer destinations), and user-defined column, row, and cell style attributes. The user-defined style attributes provide the ability to create reports with dynamic attributes that change based on the reported values. Once a user gains more experience, these styles can be saved within PROC TEMPLATE and applied to output from other procedures. I provide an example of an Adverse Event Report for internal medical review which automatically highlights triggering events, and potentially important medical information.

INTRODUCTION

Each new version of SAS offers greater tools and support for report creation through ODS. The power of ODS is the ability to change and modify almost every aspect of any output created in SAS. This is also what can make it overwhelming for novice users of SAS. This paper will give some easy ways to greatly improve SAS output while avoiding the more advanced aspects of ODS, specifically PROC TEMPLATE.

The paper begins with several non-ODS formatting options available in PROC REPORT. Next, information on getting, and saving ODS output are highlighted. Again, this focuses on printer destinations and Microsoft rich text format. Next, ODS formatting available without PROC TEMPLATE is introduced; specifically in-line formatting and user-defined formats. Finally, it is all put together in an example.

PROC REPORT OPTIONS

PROC REPORT can produce complex reports including grouping, calculations, and special formatting. In the context of this paper, several common formatting options available in the DEFINE statement are listed. These allow columns to be individually tailored by the user to each variable.

Below is a very simple PROC REPORT statement showing how the DEFINE statement is used. It establishes any column title, formatting, or calculation (not covered here) to be used in producing the report.

```
proc format data=new_rep;
  column ID test_dt comments;
  define test_dt / "Test Date";
  define comments / "Comments";
run;
```

When a define statement is not listed for a variable the default values, such as using the variable name as the column header, are used.

FORMAT=

This option applies either SAS defined, or user defined, formats to the variable.

```
define test_dt / format=mmddyy10. "Test
Date";
```

WIDTH=, FLOW=

Width controls the size of the defined column. Values can range from 1, being a single character width, to the number of characters in the page width (established in the SAS option LINESIZE=). Flow can be used with the width option for character variables. It wraps the variable onto the next line if the character string extends beyond the given column width. If not used, SAS will truncate the variable to fit within the specified width.

```
define comments / flow width=30 "Comments";
```

ORDER, ORDER=

This option establishes which variable to order by in the report. The ORDER= option allows users to define if the order is by the formatted or internal (unformatted) values of the variable. If the variable is formatted in the dataset it is best to explicitly state how the variable should be ordered.

```
define ID / order order=internal;
```

CENTER | LEFT | RIGHT

This option justifies the entire column including the header to the center of the column width, to the left edge of the column width, or to the right edge.

```
define test_dt / center format=mmddyy10.
"Test Date";
```

BASIC ODS: GETTING OUTPUT

ODS PRINTER and ODS RTF can produce printable output from any SAS procedure. The printer destination can produce postscript (PS), portable document format (PDF), and HP LaserJet printer format (PCL) from SAS on any platform. The RTF destination produces Microsoft rich text format (RTF). This output comes from the SAS procedure itself. ODS captures a data object from the procedure using a template to format the data. PROC TEMPLATE gives the user full access to define and adapt the template, but SAS can create the data object without the user bothering with PROC TEMPLATE by using the default template or one of the SAS defined templates.

EXAMPLE 1

Creating a postscript or pdf file using ODS. The first line opens the printer destination, defines the type of printer (ps or pdf) and points to the destination file. The SAS code that produces the output is then run, and can include several different procedures. The last line closes the destination and initiates writing to the file.

```
/* postscript */
ods printer ps file='C:\ods\report.ps';
/* SAS procedure creates output */
ods printer ps close;

/* pdf */
ods printer pdf file='C:\ods\report.pdf';
/* SAS procedure creates output */
ods printer pdf close;
```

EXAMPLE 2

Creating a rtf file using ODS. The first line opens the destination, and points to the destination file. The last line closes the destination and initiates writing to the file. The only difference from Example 1 is the destination.

```
ods rtf file='C:\ods\report.rtf';
/* SAS procedure creates output */
ods rtf close;
```

EXAMPLE 3

Although ODS will create output with the default template, there are several styles that are built into SAS and easy to use. A standard style I use for non-HTML output is *sasdocPrinter* due to

its lack of shading in column headers. Also, the default style can be changed by the user and the site administrator so it is better to define the 'default'.

```
ods rtf style=sasdocPrinter
    file='C:\ods\report.rtf';
/* SAS procedure creates output */
ods rtf close;
```

ODS FORMATTING WITHOUT PROC TEMPLATE

Two types of formatting are presented that can be used with ODS, but independent of PROC TEMPLATE. These are in-line formatting and user-defined formats.

IN-LINE FORMATTING

This formatting is embedded within the text of a title or column heading. It uses a 'turn-on' 'turn-off' type of command structure allowing multiple style attributes to be nested within each other. The in-line formatting is put within the double quotes of a text string, and like a macro variable, it uses a special symbol to identify the non-text characters. This symbol is defined by the user and is called an escape character (ods escapechar). This symbol should be chosen carefully, and should not be something used in the text strings as actual text. As when placing macro variables in a text string, double quotes must be used. The style attributes that can be changed are numerous, and interested readers can find a listing in Lauren Haworth's SUGI paper, as well as examples in Brian Schellenberger's 2000 paper. Both are available on the internet. The in-line format is started and ended with a style statement, e.g., \S={}. The 'turn-on' statement identifies the style to change, and the 'turn-off' can either define the default style, or can be a blank style statement which will cause the defaults to be used.

EXAMPLE 4

```
ods escapechar= '\';

/* Superscripting */
Title "Measured Response per unit of
    Treatment (cm\{super 2})";

/* Changing font weight within proc report */
proc report data=file;
    column variable;
    define variable / "This is
        \S={font_weight=bold}important\S={}";
run;

/* Changing size and weight */
proc report data=file;
    column variable;
    define variable / "This is
        \S={font_size=10} really
        \S={font_weight=bold}important\S={}";
run;
```

Notice in the final part of the example that the font size is 10 for both words ("really important") but only bold for the last word.

USER-DEFINED FORMATS

The styles used in the in-line formatting can also be applied to rows or cells. The use of color, bold, or size can be used to highlight important values, or to add information to the table. To make a style dependent on a single cell value, a format created using PROC FORMAT can be used. This format will define the colors or levels of bolding, and how they are associated with the variable values. The below example creates the format \$boldgrp. to make the 'Always' group print bold in the table. The style format is listed in the define statement for that variable. To create a style for an entire column the compute statement is used in PROC REPORT with a call define statement. In the below example, the variable which triggers the color for the row is not actually printed in the table, but is only used to create the format. The variable could be shown, but the use of color provides the

same information so the variable is not necessary in the table.

EXAMPLE 5

```
proc format; /* format used with style */
    value $boldgrp
        'Unk' = 'medium'
        'None' = 'medium'
        'Sometimes' = 'medium'
        'Always' = 'bold'
    ;
run;

proc report data=file;
    column id date react1 new_id;
    define id /order center "ID";
    define date / format=mmddyy10. "Date";
    define grp / "Groups"
        style={font_weight=$boldgrp.};
    define new_id / display noprint;
    compute new_id;
        if new_id eq "Y" then
            call define(_row_, "style",
                "style={foreground=blue}");
    endcomp;
run;
```

EXAMPLE REPORT

First, a point about this example. It is completely hypothetical, and does not represent the research or health care at any known facility. I created the data. The point is to illustrate the usefulness of the previous information.

The idea behind the example is that a researcher, or health care professional, may want timely reports which automatically highlight 'important' events. These important events may trigger the individual's appearance in the report like for an adverse event report, or may point out critical information in a weekly listing of new patients seen in a clinic.

To show how the formatting I presented can improve a report, I first present a 'basic' report. Example 6 gives SAS code for a basic table using PROC REPORT. There are nine variables displayed, each with a define statement listing the column title, and in one case a format. The options used in the procedure statement include the following. Headline draws a line under the column headers. Headskip includes a blank line under the column headers. Nowd stands for no windows and it suppresses the report window in a windowing environment and sends the report to the output window or file. Split identifies where the column headers should be split into multiple lines.

The output following the code is a picture of the report.

EXAMPLE 6 - BASIC PROC REPORT

```
title "Adverse Event Report for Fourth
    Quarter";
title2 "Protocol 1X";
proc report data=med_data headline headskip
    nowd split='*';
    column id test dt react1 sbp dbp test1
        test2 followup comments;
    define id /order 'ID';
    define test_dt / format=mmddyy10.
        'Test*Date';
    define react1 / 'Reactivity';
    define sbp / 'Systolic BP';
    define dbp / 'Diastolic BP';
    define test1 / 'Arm*Erythema*(mm^2)';
    define test2 / 'Secondary*Test';
    define followup / 'Wait*for*Followup';
    define comments / 'Comments';
run;
```

PICTURE OF THE REPORT

page 1

Adverse Event Report for Fourth Quarter Protocol 1X

ID	Test Date	Reactivity	Systolic BP	Diastolic BP	Arm Erythema (mm ²)	Secondary Test	Follow up
1001	15911	Severe	145	100	12	39.5	N
2001	15238	Mild	120	75	20.4	30.1	N
3001	15263	Moderate	160	90	10.6	32.1	Y
4001	15267	Mild	135	82	9.7	28.9	N

page 2

Adverse Event Report for Fourth Quarter Protocol 1X

Comments

Subject reported taking new medication starting last Wednesday
8. under stress at work, will follow up in a few days on high sbp

The basic report has several drawbacks. First, it is on two pages and it is hard to tell which ID corresponds to the comments. Second, the wait for followup variable has an unreadable column header and the information should probably be more prominent as it effects whether the reviewer should investigate any data or health problems for this participant. Finally, this is a short report, but for a large study a listing of participants or clinic patients could be substantial. It would be helpful if data which held significant clinical or research values was easier to pick out of the list.

Example 7 shows SAS code for an improved report. It includes only a few changes. First, the multiple page problem is solved by establishing cell widths, and using flow to wrap the comments. Next, the wait for followup variable is not explicitly shown in the table, but instead is used to change the color of the entire record. A footnote is included which explains the use of color in the table ("Notes in blue are awaiting followup"). Next, several style formats are created and used to highlight significant data values. In this case reactivity of severe is printed in bold, and a secondary test value over 35.0 is made both red and bold. Finally, the report is in pdf which allows for easy emailing of the report, or posting on a secure web page, while maintaining the formats. The notoc option in the PROC REPORT statement stands for 'no table of contents' and with the pdf destination it suppresses automatic bookmark creation.

EXAMPLE 7 - ODS PROC REPORT

```
/* Formats to be used in PROC REPORT */
proc format;
  value $boldcat /* bold mod/sev for react */
    'Unk' = 'medium'
    'NA' = 'medium'
    'None' = 'medium'
    'Mild' = 'medium'
    'Moderate' = 'bold'
    'Severe' = 'bold'
  ;
  value boldtst /* bold test > 35 */
    35.01-999 = 'bold'
    other = 'medium'
  ;
  value red
    35.01-999 = 'red'
    other = 'black'
  ;
run;

/* Start ODS*/
```

```
ods printer pdf notoc style=sasdocPrinter
  file='C:\ods\temp.pdf';
Ods escapechar='\'';
/* Titles and Footnotes */
title "Adverse Event Report for Fourth Quarter";
title2 "Protocol 1X";
footnote "\S={font_size=8pt}Notes in
\S={font_size=8pt
  foreground=blue}blue
\S={font_size=8pt foreground=black} are
  awaiting followup\S={}";
```

```
/* The Report */
proc report data=med_data headline headskip
  nowd split='*' missing
  style(header)=(font_face=Helvetica);
  column id test_dt react1 sbp dbp test1
  test2 followup comments;
  define id /order center 'ID';
  define test_dt / format=mmddyy10.
    'Test*Date';
  define react1 / right 'Reactivity'
    style={font_weight=$boldcat.};
  define sbp / center 'Systolic*BP';
  define dbp / center 'Diastolic*BP';
  define test1 / center
    "Arm*Erythema*(mm\{super 2})";
  define test2 / center 'Secondary*Test'
    style={font_weight=boldtst.
    foreground=red.};
  define followup / display noprint;
  define comments / flow
    style={cellwidth=2.0in
    font_style=italic} 'Comments';
/* Makes row bold if followup yes */
  compute followup;
    if followup eq 'Y' then
      call define(_row, "style",
        "style=[foreground=blue]");
    endcomp;
  run;

/* Closing ODS and sending off data obj */
ods printer pdf close;
```

PICTURE OF THE REPORT

page 1

Adverse Event Report for Fourth Quarter Protocol 1X

ID	Test Date	Reactivity	Systolic BP	Diastolic BP	Arm Erythema (mm ²)	Secondary Test	Comments
1001	12/02/2001	Severe	145	100	12	39.5	Subject reported taking new medication starting last Wednesday
2001	09/20/2001	Mild	120	75	20.4	30.1	
3001	10/15/2001	Moderate	160	90	10.6	32.1	8. under stress at work, will follow up in a few days on high sbp
4001	10/19/2001	Mild	135	82	9.7	28.9	

Footnote:

Notes in blue are awaiting followup

CONCLUSION

The purpose of this paper is to offer novice users of SAS ODS several ways to improve and customize existing reports. ODS offers a powerful tool for customizing reports and creating high quality output in a variety of formats. Once comfortable with the basics of ODS, users can learn to document and save these formats within a user style through PROC TEMPLATE. I encourage users to look up the references below which offer

several more advanced examples. The improvements listed in this paper are only a small sample of what can be done.

REFERENCES

Haworth, Lauren. ODS for PRINT, REPORT and TABULATE, Genentech, Inc., SUGI 26, 2001.
<http://www2.sas.com/proceedings/sugi26/p003-26.pdf>

McNeill, Sandy. Changes & Enhancements for ODS by Example (through Version 8.2), SAS Institute Inc., SUGI 26, 2001.
<http://www2.sas.com/proceedings/sugi26/p002-26.pdf>

Schellenberger, Brian. Presentation-Quality Tabular Output via ODS, SAS Institute Inc. 2000.
<http://www.sas.com/rnd/base/topics/odsprinter/qual.pdf>

Schellenberger, Brian. ODS PRINTER Family Statement Overview, SAS Institute Inc.
<http://www.sas.com/rnd/base/topics/odsprinter/stmt.html>

ACKNOWLEDGMENTS

This work was begun at SCHARP, Fred Hutchinson Cancer Research Center in Seattle, WA with the help of Geoff Snyder. This presentation was supported by my current employer, HealthPartners Research Foundation in Minneapolis, MN.

CONTACT INFORMATION

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

Nicolle Mode, M.S.
HealthPartners Research Foundation
8100 34th Ave S, Mail stop 23302G
Minneapolis, MN 55440-1524
Work Phone: 952-967-7721
Fax: 952-967-5022
Email: nicolle.a.mode@healthpartners.com
Web: www.hrfp.org

LEGAL DISCLAIMER

SAS is a registered trademark of the SAS Institute Inc. Other brand names, such as Microsoft, are registered trademarks of their respective companies. Neither HealthPartners, nor Fred Hutchinson Cancer Research Center is in any way responsible for the information or misinformation contained in this paper.