Paper 80

Jazzing up Your Reports - Some Tricks with PROC REPORT

By David Trenery, Hoechst Marion Roussel, Denham

Abstract

Proc Report is an extremely useful and versatile procedure for producing tables and listings. There are several enhancements that I wanted in my reports, which I could not find in the manual. I produce many reports and I wanted to automatically break-up reports with solid lines, add customised page numbering and conditional footnotes. This paper outlines my solutions. It is targeted at those who have had some previous experience with Proc Report.

Introduction

PROC REPORT is my report generator of choice as it is more versatile than PROC PRINT and TABULATE, it’s ability to flow text is extremely useful and it is much easier to use than PUT statements in a data step. However the manual does not explain how to do some things that I must have in my reports.

These are how to:

- break up sections of the report with solid lines
- number the pages of report in the form ‘Page *x* of *y*’ immediately at the foot the report
- add footnotes conditional on the contents of the current page of the report

In this paper I will show you how put these in my reports.
All examples in this paper have been tried and tested in version 6.12 of the SAS® system.

Breaking up sections of the report with solid lines.

My boss likes a solid line spanning the entire report to break up sections and a double solid line to signify the end of the report. It is not to difficult to produce report breaks with lines going across the page every time a variable changes or conditionally changing the line type.

This can be done with a control variable which signifies the changing sections, and a compute block which draws a line of a specified length from a specified start (see the following SAS code).
I increment the control variable by one for each new section except for the last which I number 9999 and use a double underline.

```
%let startpos=12;      * Starting position of
line;
%let linelen =80;      * Length of line;

PROC REPORT ... headline ...;
      * Headline draws a line across the top;
     COLUMNS section var1 var2;
     DEFINE  section / order;
       * Draw a line across at the end of each
section;
     COMPUTE AFTER section;
              * Use a double underline for the
last section;
          IF section = 9999 then
               uline=repeat('”',&linelen);
          ELSE uline=repeat('ƒ',&linelen);
          LINE @&startpos uline $200.;
     ENDCOMP;
RUN;
```

The main difficulty with this method is that the start position and length of the line will vary with the data, report and current LINESIZE setting. To find these automatically you can either:

1) Produce the report twice. The first time send it to a file. Then process this file to find the start position and the length of the line produced by the HEADLINE option.
2) Find the characteristics of the variables in the report (length and formats applied) and the values of the options SPACING, WIDTH and NOPRINT in the PROC REPORT and DEFINE statements.

The second option is my preferred method and I will describe it in a little more detail.
I use PROC DATASETS to place the variable information into a SAS data set. To find the PROC REPORT details I read the stored version of the program I am running and process it to find the relevant PROC REPORT statements. Details on the WIDTH, SPACING and NOPRINT options are placed in a SAS data set. These two data sets provide me with all the information needed to calculate how long my lines should be.

Once I have my line length, I put this into the following equation to calculate my starting position.

```
startpos=round((linesize-linelen+1) /2);
        * LINESIZE is the current value of the
          LINESIZE option;
```

This is relatively easy to automate but the code is a little lengthy to go into any detail in this short paper.

Customised page numbering

My boss also likes the page numbering of each table and listing in the form ‘Page 1 of 6’ at the base of the table, while the bottom of the page is reserved for numbering the entire document. The easiest way to do this is to process the data first to create a page numbering variable which increases by one for each new page. This variable is used in the SAS code over the page to sort the data, cause page breaks, label the current page, calculate the last page, and to determine the end of the report.
The crucial thing to realise when using this method is that you cannot use a variable to sort by, <u>and</u> break-up the report, <u>and</u> use it inside a compute block. To get around this I create three identical variables to carry out each task (page, page2 and page3 in the accompanying SAS code).

Conditional footnotes

Often footnotes are wanted which are conditional on the contents of the current page of the report.
The easiest method I have found to do this is to create a footnote variable, which contains the text of the desired footnote. Its value is the same for an entire page of a report but may (or may not) change between pages. I then print the footnote at the bottom of the report page.

The SAS code

```
*--- Create data set with footnotes ---;
DATA footnote;
    SET demo;
     WHERE flag = ’*’;
     Footnote=’* Protocol violator age > 85
     years’;
     KEEP page footnote;
RUN;

*--- Merge in footnotes ---;
*--- Create additional page variables, 3 now needed ---;
DATA demo2;
     MERGE demo footnote;
     BY page;
     page2=page;
     page3=page;
RUN;

*--- Produce report ---;
PROC REPORT data=demo2 headline;
   COL page footnote page2 allocate
       subno flag age sex weight page3;
   DEFINE page     / order noprint;
               * 1st page variable to sort on;
   DEFINE footnote / order noprint;
               * Used in PAGE2 compute block
                 must occur before PAGE2 variable;
   DEFINE page2    / order noprint;
               * 2nd page variable to break on;
   DEFINE page3    / max   noprint;
               * 3rd page variable to find max on;
   DEFINE flag     / display ’ ’ spacing=0;

   COMPUTE before;
      lastpage=page3.max;
   ENDCOMP;

   BREAK  after page2/ skip;
   COMPUTE after page2;
      pagetext=compbl(’Page ’||put(page,3.)||
           ’ of ’ ||put(lastpage,3.));
      IF page=lastpage then
            uline=repeat('”',60);
      ELSE  uline=repeat('ƒ',60);
      LINE @2 uline $60.;
      LINE @2 footnote $45. @50 pagetext $15.;
   ENDCOMP;
RUN;
```

Below is the resulting customised report.

```
Table 1. Example of customised report.
____________________________________________________________
Treatment             Patient    Age  Sex        Weight (kg)
____________________________________________________________
Wonder drug             1-1113    52  Female              50
                        1-1116    72  Male                88
                        1-1291    56  Female              74
                        1-2232*   87  Male                99
____________________________________________________________
* Protocol violator age > 85 years             Page 1 of 6
```

```
Table 1. Example of customised report.
____________________________________________________________
Treatment             Patient    Age  Sex        Weight (kg)
____________________________________________________________
Old drug                1-1115    51  Female              89
                        1-1117    75  Male                80
                        1-1293    62  Male                75
                        1-1413    77  Male                90
                        1-1419    43  Male                86
                        1-2231    69  Female              72

============================================================
                                               Page 6 of 6
```

Discussion

I have described here three useful tricks that I use to jazz up my reports. I could not find these in the manual and discovered them through trial and error. I hope that these will improve both the look of your reports and make their production much easier. I find that Proc Report is an extremely useful procedure and the methods described here have enabled me to avoid creating reports with PUT statements in a DATA step, which is always good to avoid.

Contacts

David Trenery, Hoechst Marion Roussel, Broadwater Park,
Denham, Uxbridge, Middlx UB9 5HP, UK
(0044) 01895-837803

David.Trenery@hmrag.com