PROC REPORT: How To Get Started

Malachy J. Foley

University of North Carolina at Chapel Hill, NC

ABSTRACT

PROC REPORT started as a soupped-up version of PROC PRINT. Now this unique product combines features from PROC PRINT, SORT, FREQ, MEANS, and TABULATE. Because of its special blend of possibilities, PROC REPORT is often the easiest way to do an elegant data listing or a report with descriptive statistics.

This tutorial shows how to use the batch versions of PROC REPORT. Via examples, it thoroughly explores the use of PROC REPORT in creating specialized data listings.

INTRODUCTION

Many people shy away from PROC REPORT because of it's mysterious defaults. Or put another way, PROC REPORT acts differently than other SAS® PROC's. However, once you see how PROC REPORT works, you may never want to go back to other PROC's.

PROC REPORT does the yeoman's work of PROC PRINT, SORT, FREQ, MEANS, and TABULATE, and PUT-Statement Formatting (DATA _NULL_) all in one procedure.

The purpose of this paper is to examine, in detail, the how to produce listings with batch-mode PROC REPORT. That is, this paper will look at the PRINT, SORT and DATA _NULL_ aspects of PROC REPORT. The FREQ, MEANS and TABULATE aspects will be examined in a subsequent article. All of the examples in this paper are designed to function using SAS version 6.12 or higher.

This tutorial is intended for SAS programmers with knowledge of the SAS dataset structure, and exposure to the SAS PRINT and CONTENTS procedures. After completing this article, the reader should be able to do data listing using PROC REPORT and have the foundation for going on to learn how to do descriptive statistics with the procedure.

SAMPLE INPUT DATA SET

The following two Exhibits are a PROC PRINT and a partial PROC CONTENTS of an example data set. This data set is going to be used as input to all the examples of PROC REPORT presented in this paper.

```
Exhibit 1. Listing of Input Data Set
------------------------------------
PROC PRINT DATA=ORG NOOBS UNIFORM;
RUN;
------------------------------------
      OUTPUT... file=ORG
------------------------------------
 ID    SITE  DUM  NAME  SEX  AGE
 A01    RU    1   SUE    F    58
 A02    LA    1    X     M    58
 A04    RU    1   TOM    M    21
 A07    LA    1   LEE    F    47
 A08    LA    1   KAY    F    29
 A10    RU    1          M    36
------------------------------------
```

```
Exhibit 2.  Partial CONTENTS of Data Set
----------------------------------------
   Variables Ordered by Position
----------------------------------------
 # Variable  Type  Len  Pos  Label
 1   ID      Char    4    0
 2   SITE    Char    2    4
 3   DUM     Num     8    6
 4   NAME    Char    3   14
 5   SEX     Char    1   17
 6   AGE     Num     8   18  Age in years
----------------------------------------
```

FEATURES & DEFAULTS OF PROC REPORT

Observe Exhibit 3. It shows a very simple PROC REPORT. This example illustrates the procedure's defaults.

```
Exhibit 3. REPORT's defaults in Batch Mode
------------------------------------------
     PROC REPORT DATA=ORG NOWINDOWS;
     RUN;
------------------------------------------
                         S
      SI            NAM  E     Age in
  ID  TE        DUM  E   X      years
  A01 RU          1  SUE F         58
  A02 LA          1   X  M         58
  A04 RU          1  TOM M         21
  A07 LA          1  LEE F         47
  A08 LA          1  KAY F         29
  A10 RU          1      M         36
------------------------------------------
```

The only option used in Exhibit 3 is the NOWINDOWS option. This option is required in batch mode and whenever you want your output sent to a file rather than a window. Since this paper is dedicated to batch mode processing, all examples will use the NOWINDOWS option. Everything else in the Exhibit is a default.

As you can see from Exhibit 3, PROC REPORT's defaults are different than PROC PRINT's defaults. In fact, some of REPORT's defaults are different than the defaults used in the rest of SAS. Below is a list of REPORT's defaults. "Same" and "Dif" indicate if the default is the same or different than PROC PRINT.

- Recs/rows ordered as they appear in data set- Same
- Variables/Columns in position order (Proc Contents) -Same
- UNIFORM is default. -Dif
- No Record Numbers (NOOBS) is default - Dif
- Labels (not var names) used as headers - Dif
- REPORT needs NOWINDOWS option. - Dif
- Default spacing not as nice as Proc PRINT. – Dif

VARIABLE NAMES AS COLUMN HEADINGS

As a default, PROC REPORT uses the variable labels as column headings. This is different than PROC PRINT which uses variable names as column headings. If you want to create a report that uses variable names as column headings, the NOLABELS system option will do the trick. This is illustrated in the next exhibit.

```
Exhibit 4. Variable Names as Col Headings
-----------------------------------------
     OPTIONS NOLABEL;
     PROC REPORT DATA=ORG NOWINDOWS;
     RUN;
-----------------------------------------
                           S
       SI              NAM E
  ID   TE         DUM  E   X        AGE
  A01  RU           1  SUE F         58
  A02  LA           1   X  M         58
  A04  RU           1  TOM M         21
  A07  LA           1  LEE F         47
  A08  LA           1  KAY F         29
  A10  RU           1      M         36
-----------------------------------------
```

MYSTERIOUS RESULTS

One of the things people don't like about PROC REPORT is that sometimes it gives some unexpected results.

Look at Exhibit 5. This is a simple PROC REPORT which is exactly the same as Exhibit 3, except that the input file is subsetted to two variables. But something strange happens. Rather than listing the values in the input data set from Exhibit 1, it sums the values! This is because the two variables (DUM and AGE) are numeric. When all the variables in the input file are numeric, PROC REPORT does a sum as a default.

```
Exhibit 5. Unexpected Results
------------------------------
PROC REPORT
      NOWINOWS
      DATA=ORG (KEEP=DUM AGE)
    ;
RUN;
------------------------------
       OUTPUT
------------------------------
           Age in
     DUM    years
      6       249
------------------------------
```

THE DEFINE STATEMENT

To avoid having the sum of numeric variables, one or more of the input variables must be defined as DISPLAY. The DISPLAY option forces each observation to Print. Furthermore, it is good programming practice to always define each variable that is in the output report. The following is an example of how to use the DEFINE statement.

This example also shows that the NOWINDOW option can be abbreviated as NOWD.

```
Exhibit 6. The DEFINE statement and NOWD
------------------------------------------
PROC REPORT DATA=ORG (Keep=DUM AGE) NOWD ;
       DEFINE  DUM  /DISPLAY;
       DEFINE  AGE  /DISPLAY;
RUN;
------------------------------------------
               Age in
       DUM      years
         1         58
         1         58
         1         21
         1         47
         1         29
         1         36
------------------------------------------
```

ORDERING/SUBSETTING COLUMNS (VARS)

In the previous example, the KEEP data set option was used to subset the variables to be displayed in the output. Another way to control which variables are outputted, is to use the COLUMN statement. This statement is similar to the VAR statement in PROC PRINT. It determines which variables are going to be in the report and in what order the variables are displayed.

```
Exhibit 7. Ordering/Subsetting Cols/Vars
----------------------------------------
   PROC REPORT   NOWD   DATA=ORG  ;
       COLUMN  AGE  DUM;
       DEFINE  AGE  /DISPLAY;
       DEFINE  DUM  /DISPLAY;
   RUN;
----------------------------------------
         Age in
          Years     DUM
             58      1
             58      1
             21      1
             47      1
             29      1
             36      1
----------------------------------------
```

SYNTAX SO FAR

As seen, in the batch mode you must always use the NOWINDOWS=NOWD option in PROC REPORT. Also, iIt is good programming practice to always use the COLUMN and DEFINE statements. Furthermore, the NOLABEL option and the DISPLAY manipulation type have been introduced. As such, the syntax of PROC REPORT thus far is.

```
Exhibit 8. PROC REPORT’s Syntax So Far
------------------------------------------
OPTIONS NOLABEL;
PROC REPORT
          NOWD
          DATA=file-name
            (OBS= WHERE=() DROP= KEEP=)
        ;
    COLUMN list-of-variables;
    DEFINE  var-name /DISPLAY;
    DEFINE  var-name /DISPLAY;
RUN;
------------------------------------------
```

WHY PEOPLE DO/DO NOT USE PROC REPORT

By now, it is becoming apparent why people avoid PROC REPORT. As seen in the previous Exhibit, PROC REPORT almost always requires more statements than PRINT. Also, PROC REPORT’s defaults are to different than PROC PRINT’s and generally unfamiliar. Sometime, like in Exhibit 5, PROC REPORT seems to act strangely. Moreover, REPORT defaults do not space fields nicely.

What may not be so apparent is why people DO use PROC REPORT. The short answer is that PROC REPORT is much more flexible than PRINT and allows you to create more professional looking reports. The significant capabilities of REPORT will become noticeable as this paper proceeds.

To appreciate the flexibility of PROC REPORT, horizontal spacing is examined in the next section.

HORIZONTAL SPACING AND MORE

The following Exhibit shows a PROC-PRINT type of report of the data using, but PROC REPORT. Notice that the output is not as nice as PROC PRINT’s defaults provide (see Exhibit 1). In the output of the next Exhibit, the column titles are crazy and spacing between the columns is not uniform.

```
Ehibit 9. A Proc-PRINT Type of Listing
------------------------------------------
PROC REPORT NOWD
             DATA=ORG (WHERE=(ID<"A08")) ;
    COL ID SITE NAME AGE ;
    DEFINE ID/DISPLAY;
    DEFINE SITE/DISPLAY;
    DEFINE NAME/DISPLAY;
    DEFINE AGE/DISPLAY;
RUN;
------------------------------------------
             SI  NAM     Age in
       ID    TE  E        years
       A01   RU  SUE         58
       A02   LA   X          58
       A04   RU  TOM         21
       A07   LA  LEE         47
------------------------------------------
```

To figure out how to make the PROC REPORT listing look nicer, one must understand how REPORT creates horizontal spacing. Actually, horizontal spacing is a combination of spacing between fields, width of the fields, format of the fields and justification of the data within the fields. All of this can be summarized as follows.

- Default Spacing between fields = 2 blanks

- Default Justification (=Alignment)
  - RIGHT for Numeric Fields
  - LEFT for Character Fields

- If no format specified, then Proc REPORT uses
  - Best9. for Numeric Fields
  - $w. for Character Fields (w=width)

- Default Width= Format width

If you examine Exhibit 9, it becomes apparent that the crazy titles and spacing is caused by the default widths. It is quite easy to upgrade the spacing with by adding a width specification to the define statements as follows.

```
Exhibit 10.Cleaning Up Exhibit 9’s Listing
------------------------------------------
PROC REPORT HEADLINE HEADSKIP NOWD
             DATA=ORG(WHERE=(ID<"A07"));
    COL ID SITE NAME AGE ;
    DEFINE ID   /DISPLAY;
    DEFINE SITE /DISPLAY WIDTH=4 RIGHT;
    DEFINE NAME /DISPLAY WIDTH=4 RIGHT;
    DEFINE AGE  /DISPLAY "AGE" WIDTH=3;
RUN;
------------------------------------------

      ID    SITE  NAME  AGE
      ---------------------

      A01     RU   SUE   58
      A02     LA     X   58
      A04     RU   TOM   21
------------------------------------------
```

You will see in Exhibit 10 that aside from adding the WIDTH option on the DEFINE statement several other things were done.

To start with, the HEADLINE and HEADSKIP options were add to the PROC REPORT statement. HEADLINE creates the line below the column titles and HEADSKIP creates the blank line below the headline.

Also, the “Age” column title was added to the DEFINE statement. This title overrides the label in the descriptor part of the input data set. Column labels or titles can come from a variety of sources. How PROC REPORT chooses which label to use is described in the next exhibit.

```
Exhibit 11. Label Hierarchy
------------------------------------------
REPORT uses the 1st label it finds in this
list.

  •  col-heading”  (DEFINE option)
  •  OPTIONS NOLABEL;  (implies Var Names)
  •  LABEL Statement in Proc REPORT
  •  LABEL in the Data Descriptor
  •  Variable Names
------------------------------------------
```

You may have noticed in Exhibit 10 that several options were used in the DEFINE statement. Actually the DEFINE statement is one of the places were REPORT derives a lot of its power. The DEFINE statement lets you do just about anything you want to the column of data it is defining. The next Exhibit outlines some of the options available in the DEFINE statement.

```
Exhibit 12. Some DEFINE Statement Options
-----------------------------------------
  •  DISPLAY
  •  Col-Heading in quotes
  •  SPACING= (Overrides all Spacings)
  •  FORMAT= (Overrides all other Formats)
  •  WIDTH=  (Default=Format Width)
  •  Justification Specifications
        o  RIGHT
        o  LEFT
        o  CENTER
-----------------------------------------
```

Notice that the DEFINE statement gives you almost total horizontal control. Also, notice that the DEFINE statement options override almost all other options in your program.

As such, the author suggest that you use the DEFINE statement to define all of your horizontal spacing rather than use other methods. There are several reasons for the suggestion. First, you don’t have to memorize the various hierarchies, like the label hierarchy of Exhibit 11. You know that whatever is in the DEFINE statement is what is going to happen. Second, you don’t have to keep track of what has already been defined elsewhere in your program. For example, you don’t have to remember if you have labels defined in the descriptor part of your program or if you have a subsequent label statement, etc.

While you don’t have to remember hierarchy rules to use REPORT, it probably is a good idea to remember the justification rules. These rules follow.

```
Exhibit 13. Justification Rules
   (Right, Left, Center)
------------------------------------------
Justification applies to
  •  col-headings
  •  data values.

Default Justification (=Alignment)
  •  RIGHT for Numeric Fields
  •  LEFT for Character Fields

Numerical values:
  •  always remain right justified within
     FORMATs
```

```
      •   FORMATs are justified within the
          WIDTH.

     For col-headings & character values:
      •    values are justified within the
          WIDTH.  (without regard for the $w.
          FORMAT).
      •   leading blanks are retained.
      •   trailing blanks are eliminated.
     ------------------------------------------
```

Just how these Justification rules work are demonstrated in the next two Exhibits. Exhibit 14 shows the case of right justification, and Exhibit 15 shows the case of left justification.

```
Exhibit 14. RIGHT Justification
------------------------------------------

DEFINE AGE / ‘AGE∆’ SPACING=3 WIDTH=6
              FORMAT=4. RIGHT;
------------------------------------------
COL=  123456789       s=Spacing location
TYPE= ssswwwwww       w=Width location
TYPE=      ffff       f=Format location
            AGE       ∆=Blank space
           ∆∆58
------------------------------------------
```

```
Exhibit 15. LEFT Justification
------------------------------------------

DEFINE AGE / ‘∆AGE’” SPACING=3 WIDTH=6
              FORMAT=4. LEFT;
------------------------------------------
COL=  123456789       s=Spacing location
TYPE= ssswwwwww       w=Width location
TYPE=     ffff        f=Format location
         ∆AGE         ∆=Blank space
         ∆∆58
------------------------------------------
```

You will note from the previous two examples that REPORT gives you almost complete horizontal control of where you are putting your data. One thing it does not give you is trailing blanks.

FORCING A TRAILING BLANK

Notice the last line of Exhibit 13. It says that for column headings and character data trailing blanks are always eliminated by REPORT. Sometimes you want trailing blanks so that your data is justified correctly. One way of creating trailing blanks is given in the next example.

This trick works on most ASCII-based computers and PC’s. The example shown in Exhibit 16 is identical to Exhibit 14 except that a &blk is placed behind the AGE column label in the DEFINE statement. The &blk character is created in the DATA _NULL_ step at the beginning of the Exhibit.

You do not need to understand how the CALL SYMPUT works to use this trick. Merely put the DATA _NULL_ somewhere at the beginning of your program and then use the &blk character whenever you need it.

One note of CAUTION… you must use double quotes (rather than single quotes) around the &blk for this trick to function.

```
Exhibit 16. Forcing Trailing Blanks
------------------------------------------
DATA _NULL_;
   CALL SYMPUT('blk','FF'X);
RUN;

DEFINE AGE / ” AGE&blk” SPACING=3 WIDTH=6
              FORMAT=4. RIGHT;
------------------------------------------

COL=  123456789       s=Spacing location
TYPE= ssswwwwww       w=Width location
TYPE=      ffff       f=Format location
           AGE∆
           ∆∆58
------------------------------------------
```

THE FLOW OPTION

Exhibits 14 to 16 discussed how to place your data almost anywhere you want to within a line. The next example reveals how to make a very long character value, like a note or a comment, to span several lines of output. This is yet another feature of PROC REPORT which makes it so powerful. The feature is called FLOW. Sometimes people will use REPORT rather than PRINT just because of the FLOW option. Observe how the FLOW option in the DEFINE statement for the note variable works.

```
Exhibit 17. The Flow Option
------------------------------------------
PROC REPORT Headline NOWD DATA=ORG;
    COL ID NAME AGE NOTE;
    DEFINE NAME /DISPLAY WIDTH=4 RIGHT;
    DEFINE AGE  /DISPLAY "AGE" WIDTH=3;
    DEFINE NOTE /DISPLAY WIDTH=13 FLOW;
------------------------------------------
```

```
     ID    NAME  AGE  NOTE
     A01    SUE   58  This is an ex
                      of FLOW.
     A02      X   58  No flow here.
     A04    TOM   21  Adverse
                      Event.
------------------------------------------
```

COLUMN HEADINGS

The previous sections demonstrate how REPORT gives you almost complete control of how to place your data horizontally. This section shows how REPORT lets you to play with the column heading.

The following exhibit shows how the dash is automatically expanded around a column heading (NOTE) and how a slash may be used to split column labels onto two lines.

```
Exhibit 18. Expanding dash.
            Splitting slash.
------------------------------------------
PROC REPORT Headline NOWD DATA=ORG;
  COL ID NAME AGE NOTE;
  DEFINE NAME /DISPLAY WIDTH=7 RIGHT
             'First/Name"';
  DEFINE AGE  /DISPLAY "AGE" WIDTH=3;
  DEFINE NOTE /DISPLAY WIDTH=13 FLOW
            '- Note- '; RUN;
------------------------------------------
            First
     ID       Name  AGE  ---- NOTE ---
     A01       SUE   58  This is an ex
                         of FLOW.
------------------------------------------
```

The next exhibit shows how the STAR is automatically expanded around a column heading (NOTE) and how a SPLIT= option works. The SPLIT option in PROC REPORT functions much as it does in PROC PRINT.

```
Exhibit 19. Expanding star.
            Split= option.
------------------------------------------
PROC REPORT Headline NOWD DATA=ORG
        SPLIT='*';
    COL ID NAME AGE NOTE;
    DEFINE NAME /DISPLAY WIDTH=7 RIGHT
             'First*Name'
    DEFINE AGE  /DISPLAY "AGE" WIDTH=3;
    DEFINE NOTE /DISPLAY WIDTH=13 FLOW
            '* Note *'; RUN;
------------------------------------------
            First
     ID       Name  AGE  **** NOTE ***
     A01       SUE   58  This is an ex
                         of FLOW.
------------------------------------------
```

Exhibit 20 illustrates how to create a column heading that spans several columns. Observe how the parenthesis indicate which columns the special title is to span. Specifically, the parentheses embrace the ID and NAME variables.

```
Exhibit 20. Special Column Heading
------------------------------------------
PROC REPORT Headline NOWD DATA=ORG;
    COL ('-ID INFO-' ID NAME) AGE NOTE;
    DEFINE ID   /DISPLAY 'ID#';
    DEFINE NAME /DISPLAY WIDTH=6 RIGHT;
    DEFINE AGE  /DISPLAY  WIDTH=3;
    DEFINE NOTE /DISPLAY WIDTH=13 FLOW
            '* Note *';    RUN;
------------------------------------------
     --ID INFO---
     ID#      NAME  AGE  **** NOTE ***
     A01       SUE   58  This is an ex
                         of FLOW.
------------------------------------------
```

The next exhibit details how to add a line above the column headings. This line could be called an "overline".

Note how the underline symbol in quotes is automatically expanded out to cover all the column headings to form the overline.

Here, like in the previous example, the parentheses indicate which columns are to be spanned with the special heading. The difference here is that there are two sets of parentheses. The outermost set is for the overline (to cover all the columns). The innermost set is to cover or span only the variable ID and NAME.

Exhibit 21 also demonstrates the automatic expansion of the greater-than and less-than symbols around the Note column title.

```
Exhibit 21. Overlining the column headings
------------------------------------------
PROC REPORT Headline NOWD DATA=ORG;
   COL ('_ _'('-ID INFO-' ID NAME) Age
                              Note);
    DEFINE ID / 'ID#';
    DEFINE NAME /DISPLAY WIDTH=6 RIGHT;
    DEFINE AGE  /DISPLAY  WIDTH=3;
    DEFINE NOTE /DISPLAY WIDTH=13 FLOW
            '< Note >';    RUN;
------------------------------------------
```

```
          ________________________________
          --ID INFO---
          ID#     NAME  AGE  <<<< NOTE >>>
          A01      SUE   58  This is an ex
                             of FLOW.
     ------------------------------------------
```

In this section, many different column heading or labeling techniques have been illustrated. These techniques are summarized in the following table.

```
Exhibit 22.Summary of Column Headings
------------------------------------------
•   Controlling the Breaks (stacking text)
    •   'xxx' 'yyy' (multiple pairs of
        quotes)
    •   'xxx/yyy' (slash is default split
        char.)
    •   'xxx*yyy' (with SPLIT='*' Report
        Option)

•   Underlining
     HEADLINE Proc Report Statement Option

•   Overlining (with text or characters)
      COL ('text' var var) var var;

•   Extending Headers to Column Width
•   'char header-text char'
      (where char pair is **  ==  ++
                -- _ _  ..  < >   > <   )

•   Justification (LEFT, CENTER, RIGHT)
------------------------------------------
```

WHY PEOPLE DO USE PROC REPORT

Earlier in the article, several reasons shy people might not use PROC REPORT were given. Now that some of the features of RPEORT have been explored, it is only fair to state some of the reasons why people DO use REPORT. Basically all the reasons are REPORT features. With REPORT you have:

- Complete Horizontal Control
- Complete Col-heading Control
- Justification Control
- The FLOW option
- More Professional Looking Reports

And this is just the beginning of the list of features. The list goes on and on. In the next section, another feature of REPORT is examined. Namely, how you can do sorts within PROC REPORT.

ORDERING LINES IN OUTPUT - SORT

Until now, only the DISPLAY manipulation option of the DEFINE Statement has been used. There are data manipulation options available. One of them is the ORDER option. This option sort the rows before the output is printed. Here is an example.

```
Exhibit 23. The ORDER Option.
----------------------------------------
  Proc REPORT nowd headline headskip
              Data=ORG;
    COL SITE NAME AGE ;
    DEFINE SITE /ORDER WIDTH=4;
    DEFINE NAME /DISPLAY WIDTH=4;
    DEFINE AGE  /DISPLAY "AGE" WIDTH=3;

----------------------------------------
          SITE  NAME  AGE

           LA     X     58
                LEE     47
                KAY     29
           RU   SUE     58
                TOM     21
                        36
----------------------------------------
```

Notice how in Exhibit 23 the records from the input data set described in Exhibit 1 are now sorted according to SITE. This was done with the ORDER option.

Also notice that in Exhibit 23 the SITE value is not repeated on each line, but rather given only once on its first occurrence. This overhang feature is automatic when you use the ORDER option.

The next exhibit is the same as the previous exhibit, except that now two variables are declared as ORDER type variables. Thus, you will get the output sorted on two variables. What's more, you should get two overhanging variables. However, in this example, each value of the second variable is distinct, so the overhanging effect is not evident.

```
Exhibit 24. Sort on two variables
------------------------------------------
  Proc REPORT nowd headline headskip
              Data=ORG;
    COL SITE NAME AGE ;
    DEFINE SITE /ORDER WIDTH=4;
    DEFINE NAME /ORDER WIDTH=4;
    DEFINE AGE  /DISPLAY "AGE" WIDTH=3;
------------------------------------------
```

```
          SITE  NAME  AGE

           LA     X     58
                 KAY    29
                 LEE    47
           RU    SUE    58
                 TOM    21
------------------------------------------
```

You might see that something is wrong with the output in the preceding exhibit. Namely, the record or row for the person with a missing Name is not printed. This is the way REPORT handles missing values on ORDER type variables. It does not print them. To get around this feature, you must add the MISSING option. The following exhibit is the same as Exhibit 24, except it includes the MISSING option .

```
Exhibit 25. MISSING Option
------------------------------------------
  Proc REPORT nowd headline headskip
              Data=ORG MISSING;
    COL SITE NAME AGE ;
    DEFINE SITE /ORDER WIDTH=4;
    DEFINE NAME /ORDER WIDTH=4;
    DEFINE AGE  /DISPLAY "AGE" WIDTH=3;
------------------------------------------
          SITE  NAME  AGE

           LA     X     58
                 KAY    29
                 LEE    47
           RU           36
                 SUE    58
                 TOM    21
------------------------------------------
```

TYPES OF STATEMENTS

REPORT like other PROC's allows you do use several general SAS statements, In addition to all the PROC REPORT statements like COL and DEFINE. Here is a list of which general statements are supported by RPEORT.

```
Exhibit 26. General SAS Statements.
----------------------------------------
Supported by PROC REPORT.
 •   WHERE
 •   LABEL - not recommended
 •   FORMAT- not recommended
 •   TITLE
 •   FOOTNOTE
 •   BY
NOT supported by PROC REPORT
 •   KEEP
 •   DROP
----------------------------------------
```

This article assumes the reader is familiar with these general SAS statements and knows how to use them.

FORMAT HIERARCHY

Exhibit 26 says that the LABEL and FORMAT statements are not recommended. This again is the suggestion to set your label and format values in the DEFINE statement for each variable. This suggestion is made so that you do not have to remember what the different hierarchies are nor remember what has been previously defined in your program. However, if you do wish to use something other than the DEFINE statement to specify your labels and formats, you should be aware of the hierarchies. The label hierarchy was presented in Exhibit 11 and the format hierarchy follows.

```
Exhibit 26. Format Hierarchy
----------------------------------------
REPORT accepts the 1st Format from the
following list that fits in WIDTH.
(Thus, WIDTH can affect FORMAT/Values.)

 •   FORMAT=  (DEFINE option)
 •   FORMAT Statement in Proc REPORT
 •   FORMAT in the Data Descriptor
 •   Default Formats as follows
 •   Best9.  for Numeric Fields
 •   $w. for Character Fields (w=width)
 •   If Value does not fit in Format
 •   Numeric Fields are filled with *
 •   Character Fields are truncated
------------------------------------------
```

Notice that WIDTH's do affect the FORMAT and values, and that WIDTH size defaults to the FORMAT size. As such, the author makes two suggestions in choosing FORMAT's and WIDTH's for REPORT

- Choose FORMATs with more columns than your maximum expected value.

- Choose WIDTHs greater than or equal to the FORMATs.

ORDER= OPTION

Another programming suggestion is to always use the ORDER= option when you use the ORDER option in a DEFINE statement. Exhibit 26 is an example of how to use this the ORDER= option. The reason for the suggestion is that in every other PROC the order is different than in PROC REPORT. So, it is easy to expect the wrong sort. Exhibit 27 explains the different ORDER= Options.

```
Exhibit 26. Order= Option
-----------------------------------------
  Proc REPORT nowd missing headline
          headskip Data=ORG;
    COL  SITE NAME AGE ;
    WHERE AGE>30. ;
    DEFINE SITE /ORDER WIDTH=14
          FORMAT=$SITE. ORDER=INTERNAL;
    DEFINE NAME /ORDER WIDTH=4;
    DEFINE AGE  /DISPLAY "AGE" WIDTH=3;
-----------------------------------------
      SITE            NAME  AGE
      ___________________________

      LOS ANGELES      X     58
                      LEE    47
      DURHAM-RALEIGH         36
                      SUE    58
-----------------------------------------
```

```
Exhibit 27. ORDER= Option  Values
-----------------------------------------
ORDER= (one of the following)
    DATA or FORMATTED or FREQ or INTERNAL

Meaning of Options
•   DATA - Order recs. as in the data set.
•   FORMATTED - Sort values after
    formatting.
•   INTERNAL - Sort values before
    formatting them.
•   FREQ - Sort values by frequency of
    occurrence in the data set.

Defaults
•   DATA when manipulation-type is
    DISPLAY.
•   FORMATTED when manip.-type is ORDER.
•   (in every other PROC, default is
    INTERNAL.)
-----------------------------------------
```

THE BREAK STATEMENT

Finally, the BREAK statement is introduced in the next exhibit. BREAKS can be use in a variety of ways. In this example, it is used to put a blank line (a SKIP) after each different SITE.

```
Exhibit 28. BREAK  Statement
------------------------------------------
Proc REPORT nowd missing headline DATA=ORG
    COL  SITE NAME AGE ;
    WHERE AGE>30. ;
    DEFINE SITE /ORDER WIDTH=4;
    DEFINE NAME /ORDER WIDTH=4;
    DEFINE AGE  /DISPLAY "AGE" WIDTH=3;
    BREAK AFTER SITE /SKIP;
RUN;
------------------------------------------
          SITE  NAME  AGE
          _______________
           LA     X     58
                 LEE    47

           RU           36
                 SUE    58
------------------------------------------
```

SYNTAX FOR LISTINGS

This article is an introduction to how to use PROC REPORT to create listings. Exhibit 29 shows all of the syntax that has been covered in this article and serves as an example of what can be done.

```
Exhibit 29. Syntax for Listings
------------------------------------------
OPTIONS NOLABEL LS= PS=;
PROC REPORT   NOWD  MISSING SPLIT=’char’
      HEADLINE HEADSKIP SPACING=
      DATA=file-nam
        (OBS= Where=() Drop= Keep=);
  WHERE expression ;
  TITLE ‘your-message‘ ;
  FOOTNOTE ‘your-message’;
  COLUMN (‘Δheader&blk’ list-of-vars)
             more-vars;
  DEFINE  var-name /<DISPLAY or ORDER>
         ‘Δcol-head&blk”
          SPACING=  WIDTH=  FORMAT=  FLOW
          RIGHT  LEFT  CENTER
          ORDER= DESCENDING;
  BREAK AFTER order-var /SKIP;
RUN;
```

PROGRAMMING RECOMMENDATIONS

Throughout this paper, the author has made several suggestions on how one might program a PROC REPORT. The following bullets summarize these suggestions/tips.

- Always use NOWD
- Use MISSING most of the time.
- Use DEFINE for every variable.
- Use a Manipulation type (Display/Order) for every DEFINE.
- Use ORDER= with ORDER manipulation type.
- Use DEFINE for horizontal control (spacing, width, format)
- Don't use FORMAT and LABEL statements.
- Indent and align code.
- Always use RUN statement.

CONCLUSION

They say "there is no such thing as a free lunch". So it is with creating listings of data sets. When you need a quick listing, PROC PRINT is your best choice. However, when you need a more formal listing, PROC REPORT is probably your best choice. REPORT provides listing features that are difficult or impossible to obtain with PROC PRINT. For example,

- Fancy headings.
- FLOW of character value onto multiple lines.
- Total horizontal control.
- Break lines

REPORT almost gives you all of the features of using PUT Statement Formatting (DATA _NULL_), and without as much work.

Nonetheless, you don't get something for nothing. REPRORT, unlike PRINT, pretty much requires you use a define statement for every variable you output. On the other hand, this is a small price to pay for near-PUT-statement capabilities.

Furthermore, PROC REPORT goes way beyond great looking listings. It offers the ability to create descriptive statistics that are usually found in the FREQ, MEANS, and TABULATE procedures. Once you are familiar with all the listing features of REPORT, it is fairly easy to go and create great-looking reports that include descriptive statistics. How to use PROC RPEORT to create statistical reports is the subject of a future paper.

REFERENCE

SAS Technical Report P-258, *Using the REPORT Procecure in a Nonwindowing Environment*, Release 6.07, Cary, NC: SAS Institute Inc., 1993.

TRADEMARKS

AUTHOR CONTACT

The author welcomes comments, questions, corrections and suggestions.

Malachy J. Foley
2502 Foxwood Dr.
Chapel Hill, NC 27514

Email: FOLEY@unc.edu