Paper PO–4

Getting Out of the PROC PRINT Comfort Zone to Start Using PROC REPORT

Imelda C. Go, Office of Assessment, South Carolina Department of Education
Abbas S. Tavakoli, College of Nursing, University of South Carolina–Columbia

ABSTRACT

PROC PRINT is one of the first things taught to a beginner SAS programmer because it provides an easy and simple way to view the records in a data set. The procedure is fast, simple, and straightforward. As one continues to learn about SAS, one finds out about other procedures, such as PROC REPORT. This paper is written for the PROC PRINT user who has not, for whatever reason, ventured into PROC REPORT territory. The paper provides examples of PROC PRINT code and the corresponding PROC REPORT code that produces the same results. Examples of what PROC REPORT can produce that PROC PRINT cannot are also provided.

INTRODUCTION

The paper will approach the transition from PROC PRINT to PROC REPORT by using a question-and-answer format. Each question will be answered using each PROC with the results presented side-by-side for easy comparison. The paper is written for SAS programmers already familiar with PROC PRINT.

The emphasis will be on using PROC REPORT to produce what is generated by PROC PRINT. There are indeed a variety of ways to get the same things accomplished, but the focus is to provide a simple example that will accomplish the task.

A few examples of PROC REPORT output that are not directly available through PROC PRINT are also provided. The simple examples will hopefully whet the PROC REPORT beginner’s appetite to learn more about PROC REPORT’s syntax and powerful capabilities.

The sample code in this paper was tested using SAS Version 9.2. The PROC REPORT examples were generated in a batch environment and the interactive/windowing environment is not discussed. Therefore, the NOWD option was used in all the PROC REPORT examples in order to suppress the windowing environment.

DATA SET FOR EXAMPLES

The following data set will be used for the examples:

<table>
<thead>
<tr>
<th>obsnum</th>
<th>points</th>
<th>grade</th>
<th>lname</th>
<th>fname</th>
<th>gender</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>3</td>
<td>Griffin</td>
<td>Angela</td>
<td>F</td>
</tr>
<tr>
<td>2</td>
<td>200</td>
<td>3</td>
<td>James</td>
<td>Ellen</td>
<td>F</td>
</tr>
<tr>
<td>3</td>
<td>400</td>
<td>4</td>
<td>Peeler</td>
<td>Lane</td>
<td>M</td>
</tr>
<tr>
<td>4</td>
<td>600</td>
<td>6</td>
<td>Whitlock</td>
<td>Nancy</td>
<td>F</td>
</tr>
</tbody>
</table>
PROC PRINT VS. PROC REPORT

Q1: How do I get a listing of all the variables in a data set?

You can use the simplest code (one statement) as shown below. All variables will be listed when no variable list is provided. The OBS column, which can be suppressed by the NOBS option, is generated automatically by PROC PRINT. For the PROC PRINT examples, the OBS column will appear in order to remind readers they are looking at PROC PRINT output on the left side of the side-by-side comparisons. PROC REPORT does not generate such a column automatically. Note that by default there is no space between the PROC REPORT variable names and the first record in the data set.

```
proc print;
```

<table>
<thead>
<tr>
<th>Obs</th>
<th>obsnum</th>
<th>points</th>
<th>grade</th>
<th>lname</th>
<th>fname</th>
<th>gender</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>100</td>
<td>3</td>
<td>Griffin</td>
<td>Angela</td>
<td>F</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>200</td>
<td>3</td>
<td>James</td>
<td>Ellen</td>
<td>F</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>400</td>
<td>4</td>
<td>Peeler</td>
<td>Lane</td>
<td>M</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>600</td>
<td>6</td>
<td>Whitlock</td>
<td>Nancy</td>
<td>F</td>
</tr>
</tbody>
</table>

```
proc report nowd;
```

<table>
<thead>
<tr>
<th>obsnum</th>
<th>points</th>
<th>grade</th>
<th>lname</th>
<th>fname</th>
<th>gender</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>3</td>
<td>Griffin</td>
<td>Angela</td>
<td>F</td>
</tr>
<tr>
<td>2</td>
<td>200</td>
<td>3</td>
<td>James</td>
<td>Ellen</td>
<td>F</td>
</tr>
<tr>
<td>3</td>
<td>400</td>
<td>4</td>
<td>Peeler</td>
<td>Lane</td>
<td>M</td>
</tr>
<tr>
<td>4</td>
<td>600</td>
<td>6</td>
<td>Whitlock</td>
<td>Nancy</td>
<td>F</td>
</tr>
</tbody>
</table>

Q2: How do I put a space after the variable name row in PROC REPORT so that the output will look more like PROC PRINT output, which by default has a blank line after the variable names?

Use the HEADSKIP option in PROC REPORT.

```
proc print;
```

<table>
<thead>
<tr>
<th>Obs</th>
<th>obsnum</th>
<th>points</th>
<th>grade</th>
<th>lname</th>
<th>fname</th>
<th>gender</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>100</td>
<td>3</td>
<td>Griffin</td>
<td>Angela</td>
<td>F</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>200</td>
<td>3</td>
<td>James</td>
<td>Ellen</td>
<td>F</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>400</td>
<td>4</td>
<td>Peeler</td>
<td>Lane</td>
<td>M</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>600</td>
<td>6</td>
<td>Whitlock</td>
<td>Nancy</td>
<td>F</td>
</tr>
</tbody>
</table>

```
proc report nowd headskip;
```

<table>
<thead>
<tr>
<th>obsnum</th>
<th>points</th>
<th>grade</th>
<th>lname</th>
<th>fname</th>
<th>gender</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>3</td>
<td>Griffin</td>
<td>Angela</td>
<td>F</td>
</tr>
<tr>
<td>2</td>
<td>200</td>
<td>3</td>
<td>James</td>
<td>Ellen</td>
<td>F</td>
</tr>
<tr>
<td>3</td>
<td>400</td>
<td>4</td>
<td>Peeler</td>
<td>Lane</td>
<td>M</td>
</tr>
<tr>
<td>4</td>
<td>600</td>
<td>6</td>
<td>Whitlock</td>
<td>Nancy</td>
<td>F</td>
</tr>
</tbody>
</table>

Q3: How do I get a listing of selected variables in a data set?

List the variables using the VAR statement with PROC PRINT and the COLUMN statement with PROC REPORT.

```
proc print;
```

```
var points lname fname;
```

<table>
<thead>
<tr>
<th>Obs</th>
<th>points</th>
<th>lname</th>
<th>fname</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>Griffin</td>
<td>Angela</td>
</tr>
<tr>
<td>2</td>
<td>200</td>
<td>James</td>
<td>Ellen</td>
</tr>
<tr>
<td>3</td>
<td>400</td>
<td>Peeler</td>
<td>Lane</td>
</tr>
<tr>
<td>4</td>
<td>600</td>
<td>Whitlock</td>
<td>Nancy</td>
</tr>
</tbody>
</table>

```
proc report nowd headskip;
```

```
column points lname fname;
```

<table>
<thead>
<tr>
<th>points</th>
<th>lname</th>
<th>fname</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Griffin</td>
<td>Angela</td>
</tr>
<tr>
<td>200</td>
<td>James</td>
<td>Ellen</td>
</tr>
<tr>
<td>400</td>
<td>Peeler</td>
<td>Lane</td>
</tr>
<tr>
<td>600</td>
<td>Whitlock</td>
<td>Nancy</td>
</tr>
</tbody>
</table>
Q4: How do I label a variable?

Use the LABEL option and statement in PROC PRINT. Use a DEFINE statement for each variable that needs a label in PROC REPORT.

```
proc print label;
var points lname fname;
label lname='Last Name'
  fname='First Name';
```

```
proc report nowd headskip;
column points lname fname;
define lname/'Last Name';
define fname/'First Name';
```

<table>
<thead>
<tr>
<th>Obs</th>
<th>points</th>
<th>Last Name</th>
<th>First Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>Griffin</td>
<td>Angela</td>
</tr>
<tr>
<td>2</td>
<td>200</td>
<td>James</td>
<td>Ellen</td>
</tr>
<tr>
<td>3</td>
<td>400</td>
<td>Peeler</td>
<td>Lane</td>
</tr>
<tr>
<td>4</td>
<td>600</td>
<td>Whitlock</td>
<td>Nancy</td>
</tr>
</tbody>
</table>

Q5: How do I control the splitting of variable labels?

Use the SPLIT option for each of the two PROCs.

```
proc print label split='*';
var points lname fname;
label lname='Sur-*name'
  fname='First Name';
```

```
proc report nowd headskip split='*';
column points lname fname;
define lname/'Sur*-name';
define fname/'First Name';
```

<table>
<thead>
<tr>
<th>Obs</th>
<th>points</th>
<th>Sur-name</th>
<th>First Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>Griffin</td>
<td>Angela</td>
</tr>
<tr>
<td>2</td>
<td>200</td>
<td>James</td>
<td>Ellen</td>
</tr>
<tr>
<td>3</td>
<td>400</td>
<td>Peeler</td>
<td>Lane</td>
</tr>
<tr>
<td>4</td>
<td>600</td>
<td>Whitlock</td>
<td>Nancy</td>
</tr>
</tbody>
</table>

Q6: How do I change the width of a column of data?

Use a format with PROC PRINT and the WIDTH option with PROC REPORT’s DEFINE statement. Note a variable name in PROC REPORT will be displayed up to the same width of the corresponding variable.

```
proc print;
var lname;
format lname $4.;
```

```
proc report nowd headskip;
column fname;
define lname/width=4;
```

<table>
<thead>
<tr>
<th>Obs</th>
<th>lname</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Grif</td>
</tr>
<tr>
<td>2</td>
<td>Jame</td>
</tr>
<tr>
<td>3</td>
<td>Peel</td>
</tr>
<tr>
<td>4</td>
<td>Whit</td>
</tr>
</tbody>
</table>
The reduction of the variable name display width in PROC REPORT is even more apparent in the next example where variable labels are used.

```
proc print data=demo label;
var lname;
format lname $4.;
label lname='Student Last Name';
```

```
proc report nowd headskip;
column lname;
define lname/width=4 'Student Last Name';
```

---

**Q7: How do I produce a list with sorted values?**

The PROC PRINT example requires sorting by `lname` and `fname` in order to get a list of names in alphabetical order. The data need to be sorted using PROC SORT prior to using PROC PRINT with a BY statement. The data do not need to be sorted using PROC SORT prior to PROC REPORT data processing. Each variable in the sorting criteria will be in a DEFINE statement with an ORDER option.

```
proc sort;
by lname fname;
```

```
proc report nowd headskip;
column grade lname fname;
define lname/order;
define fname/order;
```

---

The second example sorts the list alphabetically per `grade`. The PROC PRINT example uses the BY and ID statements. The SKIP option in the BREAK AFTER statement puts a line after each cluster of records per `grade` in PROC REPORT.

```
proc sort;
by grade;
```

```
proc report nowd headskip;
column grade points lname fname;
define grade/order;
break after grade/skip;
```

---

```
Q8: How do I get observation numbers in the listing?

The observation number is automatically displayed by PROC PRINT unless it is suppressed using the NOBS option. To create the observation number, use PROC REPORT’s COMPUTE block to calculate the OBS column. The ENDCOMP statement marks the end of the COMPUTE block.

```
proc print;
var points lname fname;

proc report nowd headskip;
column obs points lname fname;
compute obs;
  count+1;
  obs=count;
endcomp;
```

<table>
<thead>
<tr>
<th>Obs</th>
<th>points</th>
<th>lname</th>
<th>fname</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>Griffin</td>
<td>Angela</td>
</tr>
<tr>
<td>2</td>
<td>200</td>
<td>James</td>
<td>Ellen</td>
</tr>
<tr>
<td>3</td>
<td>400</td>
<td>Peeler</td>
<td>Lane</td>
</tr>
<tr>
<td>4</td>
<td>600</td>
<td>Whitlock</td>
<td>Nancy</td>
</tr>
</tbody>
</table>

Caution: If you use the WHERE statement in PROC PRINT and PROC REPORT, results will likely vary. The OBS number in PROC PRINT is automatically generated and the records will be numbered according to the order they appear in the data set. When a WHERE statement is applied, the original numbering with the complete data set appears in the OBS column in PROC PRINT. In PROC REPORT, the OBS column is created according to the number of rows that are processed. Since the WHERE condition resulted in just one observation, the OBS column value for the PROC REPORT example is 1. In contrast, the OBS column value is 2 in the PROC PRINT example.

```
proc print;
var points lname fname;
  where lname='James';

proc report nowd headskip;
column obs points lname fname;
compute obs points lname fname;
  count+1;
  obs=count;
endcomp;
  where lname='James';
```

<table>
<thead>
<tr>
<th>Obs</th>
<th>points</th>
<th>lname</th>
<th>fname</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>200</td>
<td>James</td>
<td>Ellen</td>
</tr>
</tbody>
</table>

In order to get the same results in PROC REPORT, the observation number should be determined in the data set prior to using PROC REPORT. The data set has the observation number in the obsnum variable.

```
proc print;
var obsnum points lname fname;
  where lname='Griffin';

proc report nowd headskip;
column obsnum points lname fname;
define obsnum/"Obs";
  where lname='James';
```

<table>
<thead>
<tr>
<th>Obs</th>
<th>points</th>
<th>lname</th>
<th>fname</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>200</td>
<td>James</td>
<td>Ellen</td>
</tr>
</tbody>
</table>
**Q9: How do I get the sum of variables in a listing?**

Use the SUM statement with PROC PRINT. Use the RBREAK AFTER statement with the SUMMARIZE option in PROC REPORT. The DOL option indicates a double line should be placed after all the records that are summarized.

```
proc print;
var points lname fname;
sum points;
```

```
proc report nowd;
column points lname fname;
rbreak after /summarize dol;
```

**Q10: How do I get the sum of variables according to values of other variables?**

Specify the variables of interest in the SUM statement with PROC PRINT. Use the BREAK AFTER statement with PROC REPORT to produce the sum of the points per grade value. The OL option means one line should be placed prior to the sum of the points per grade. The DOL option means a double line should be placed at the end of the report prior to the sum of the points over all records.

```
proc print;
by grade;
id grade;
var points lname fname;
sumby grade;
sum points;
```

```
proc report nowd;
column grade points lname fname;
define grade/order;
break after grade/skip summarize ol;
rbreak after/summarize dol;
```
PROC REPORT FEATURES WITH NO PROC PRINT COUNTERPART

In the following examples, PROC REPORT is used to calculate descriptive statistics, perform user-defined calculations, and display the results. PROC PRINT has a very limited capability for producing descriptive statistics. In order for PROC PRINT to produce output with other types of descriptive statistics, the statistics would be produced by another PROC and then rendered for output using PROC PRINT.

Q11: How do I generate columns for output display using PROC REPORT?

In this PROC REPORT example, two columns in the output appear as a result of using COMPUTE blocks in PROC REPORT to calculate the values. In contrast, PROC PRINT does not have a similar capability.

The
\[ \text{ptsXgrade} \]
variable is simply the product of the
\[ \text{points} \]
and
\[ \text{grade} \]
values. The
\[ \text{wholename} \]
variable is the concatenation of the
\[ \text{lname} \]
and
\[ \text{fname} \]
variables with a comma separating the two.

```
proc report data=demo nowd;
column points grade ptsXgrade lname fname wholename;
define points/display;
define grade/display;
define ptsXgrade/computed 'points X grade';
define wholename/computed;
compute ptsXgrade;
  ptsXgrade=points*grade;
endcomp;
compute wholename/character length=16;
  wholename=trim(lname)||', '||trim(fname);
endcomp;
```

<table>
<thead>
<tr>
<th>points</th>
<th>grade</th>
<th>ptsXgrade</th>
<th>lname</th>
<th>fname</th>
<th>wholename</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>3</td>
<td>300</td>
<td>Griffin</td>
<td>Angela</td>
<td>Griffin, Angela</td>
</tr>
<tr>
<td>200</td>
<td>3</td>
<td>600</td>
<td>James</td>
<td>Ellen</td>
<td>James, Ellen</td>
</tr>
<tr>
<td>400</td>
<td>4</td>
<td>1600</td>
<td>Peeler</td>
<td>Lane</td>
<td>Peeler, Lane</td>
</tr>
<tr>
<td>600</td>
<td>6</td>
<td>3600</td>
<td>Whitlock</td>
<td>Nancy</td>
<td>Whitlock, Nancy</td>
</tr>
</tbody>
</table>

Q12: How do I generate a simple frequency distribution using PROC REPORT?

The following example is a frequency distribution for the variable
\[ \text{grade} \]. The percentages are displayed in \text{percent7.1} format by specifying the \text{FORMAT} option with the \text{DEFINE} statement. The \text{BOX} option was used to display the table gridlines.

```
proc report data=demo nowd box;
column grade n pctn;
define grade/group;
define pctn/format=percent7.1;
```

<table>
<thead>
<tr>
<th>grade</th>
<th>n</th>
<th>pctn</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2</td>
<td>50.0%</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>25.0%</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>25.0%</td>
</tr>
</tbody>
</table>
Q13: How do I generate a frequency distribution with cumulative frequencies and percentages using PROC REPORT?

The following example is a cumulative frequency distribution for the variable `grade`.

```
proc report nowd box;
  column grade n pctn cumn cumpctn;
  define grade/group;
  define pctn/format=percent7.1;
  define cumn/computed format=7.;
  define cumpctn/computed format=percent7.1;
  compute cumn;
    tempn+n;
    cumn=tempn;
  endcomp;
  compute cumpctn;
    temppctn+pctn;
    cumpctn=temppctn;
  endcomp;
```

<table>
<thead>
<tr>
<th>grade</th>
<th>n</th>
<th>pctn</th>
<th>cumn</th>
<th>cumpctn</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2</td>
<td>50.0%</td>
<td>2</td>
<td>50.0%</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>25.0%</td>
<td>3</td>
<td>75.0%</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>25.0%</td>
<td>4</td>
<td>100%</td>
</tr>
</tbody>
</table>

Q14: How do I generate a cross-tabulation using PROC REPORT?

The following example produces the cross-tabulation between the `grade` and `points` variables. The GROUP and ACROSS options are used with the corresponding DEFINE statements.

```
proc report data=demo nowd box;
  column grade points;
  define grade/group;
  define points/across;
```

<table>
<thead>
<tr>
<th>grade</th>
<th>100</th>
<th>200</th>
<th>400</th>
<th>600</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>
Q15: How do I generate a table of summary statistics using PROC REPORT?

The following example illustrates that PROC REPORT is not limited to displaying record-level detail. It can start mimicking output generated by PROC TABULATE. In this example the sum of the points (totalpoints variable) is displayed by grade.

```
proc report data=demo nowd box;
  column grade points=totalpoints;
  define grade/group width=5;
  define points/n;
  define totalpoints/sum 'Total Points';
```

<table>
<thead>
<tr>
<th>grade</th>
<th>Total Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>300</td>
</tr>
<tr>
<td>4</td>
<td>400</td>
</tr>
<tr>
<td>5</td>
<td>600</td>
</tr>
</tbody>
</table>

In the next example, more descriptive statistics are computed for the points variable by grade.

```
proc report nowd box;
  column grade points, (n mean min max);
  define grade/group;
  define points/’Points’;
  define n/’N’ width=3;
  define mean/’Mean’;
  define std/’SD’;
  define min/’Min’;
  define max/’Max’;
```

<table>
<thead>
<tr>
<th>grade</th>
<th>N</th>
<th>Mean</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2</td>
<td>150</td>
<td>100</td>
<td>200</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>400</td>
<td>400</td>
<td>400</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>500</td>
<td>500</td>
<td>500</td>
</tr>
</tbody>
</table>

The next example produces the total points per combination of grade and gender values.

```
proc report data=demo nowd split='*' box;
  column grade gender, points;
  define grade/group;
  define gender/across center;
  define points/analysis sum 'Total*Points';
```

<table>
<thead>
<tr>
<th>grade</th>
<th>Total Points</th>
<th>N</th>
<th>Total Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>300</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>400</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>600</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
REFERENCES


CONTACT INFORMATION
Imelda C. Go, PhD
Office of Assessment
Department of Education
1429 Senate Street
Columbia, SC 29201

Abbas S. Tavakoli, DrPH, MPH, ME
College of Nursing
University of South Carolina
1601 Greene Street
Columbia, SC 29208-4001
Tel: (803) 777-2978
Fax: (803) 777-5561
E-mail: abbas.tavakoli@sc.edu

TRADEMARK NOTICE
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.