

Proc Print to Be Proud Of

Lynn Pope, Amadeus Software Limited



ABSTRACT

We all use proc print, it is one of the simplest procedures in the book. So, if this is the case why don't we use to its full capabilities. The aim of this paper is to show you how much more you can do with proc print. All those options that we know are there but have never got around to using, or have long since been forgotten. Also, we will be looking at how we can significantly enhance our reports using ODS. Proc print is more than just Proc Print;Run; so explore the procedure and I think you will be pleasantly surprised.

INTRODUCTION

Proc print is a simple reporting procedure, which provides lots of interesting options for enhancing our reports. However, what is often the case is it is learnt and then its capabilities are quickly forgotten and overlooked. This paper highlights some of those more useful capabilities, with the aim of putting proc print back on the reporting agenda.

OVERVIEW

The Print procedure prints observations of a dataset to the listing window, and displays all or some of the variables. You have the ability to create a very simple report or use some of the print features to produce a more complex enhanced report.

PROC PRINT SYNTAX

Most people are used to using proc print, however are you limiting yourself to just the basic 2 lines of code:

```
Proc print data=work.demog;
Run;
```

Consider the complete syntax of Proc Print as shown below:

```
PROC PRINT DATA= SAS-table
      CONTENTS='link-text'
      DOUBLE
      NOOBS
      UNIFORM
      LABEL
      SPLIT= 'split-
character'
      N
      ROUND
      HEADING= direction
      ROWS= page-format
      STYLE
      (location(s))=style-element-name
      WIDTH= column-width;
      VAR variable-list;
      ID variable-list;
      BY variable-list;
      PAGEBY BY-variable;
      SUMBY BY-variable;
      SUM variable-list;
RUN;
```

You can also use Format, Where, and Label statements within Proc Print.

PROC PRINT STATEMENTS

VAR

Selects variables that appear in the report and determines their order, regardless of the order they are stored on the dataset.

Syntax : **Var** variable-n;

```
proc print.sas
Proc Print data=work.demog;
Var staffno age salary gender;
run;
```

Obs	staffno	age	salary	gender
1	001	22	13592.45	M
2	002	26	8870.23	F
3	003	29	12672.26	M
4	004	28	23760.82	M
5	005	23	10512.63	F
6	006	30	7520.61	M
7	007	34	28512.66	F
8	008	38	14840.27	F
9	009	56	47520.38	F
10	0010	52	23760.44	F

Hint: You can list variables if they have the same prefix:

```
proc print.sas
Proc Print data=work.demog ;
Var Temp1-Temp10;
run;
```

BY

Produces a separate section of the report for each BY group.

Syntax: **BY** <DESCENDING> variable-n <NOTSORTED>;

```
proc print.sas
Proc sort data=work.demog;
by gender;
run;

Proc Print data=work.demog;
Var staffno age salary;
By gender;
run;
```

We are producing a report for each by group gender. You can use more than one by group. If the notsorted option is not specified the dataset must be sorted by all the variables specified on the by statement.

----- gender=F -----			
Obs	staffno	age	salary
51	0051	61	25410.14
52	0052	21	25410.18
53	0053	45	15410.32
----- gender=M -----			
Obs	staffno	age	salary
54	0054	33	17610.44
55	0055	59	23023.33
56	0056	23	14060.82
57	0057	28	19050.35

ID

Identifies observations by using the formatted values of the variables that you list instead of by using observation numbers.

```
proc print.sas
  Proc Print data=work.demog ;
  Var staffno age salary;
  Id gender;
```

gender	staffno	age	salary
F	0051	61	25410.14
F	0052	21	25410.18
F	0053	45	15410.32
M	0054	33	17610.44
M	0055	59	23023.33
M	0056	23	14060.82
M	0057	28	19050.35
F	0058	25	13085.33
F	0059	28	12050.38
M	0060	38	26050.45
M	0061	49	21080.40
M	0062	43	22065.47
M	0063	19	10000.47

If the ID variable exists on the Var statement as well, it will print the variable twice.

ID AND BY TOGETHER

If the Id and the by statement are used together then proc print uses a special layout for the report.

```
proc print.sas
  Proc Print data=work.demog;
  Var staffno age salary;
  Id gender;
  by Gender;
  run;
```

gender	staffno	age	salary
F	002	26	8870.23
	005	23	10512.63
	007	34	28512.66
	008	38	14840.27
	009	56	47520.38
	0010	52	23760.44
	0011	60	20592.16
	0012	56	13760.27
	0013	46	47520.92

As can be seen by the above output the byline is suppressed, and the by variable is displayed as a separate column on the left of the report.

NOBYLINE

When you run a proc print with a by statement your normally get a by line splitting the screen, unless you use ID. You can remove this byline by using the global option nobyline.

You can then show the by variable and its value in a title statement. To do this use the #byvarn, #byvaln, as shown below.

```
proc print.sas
  Options nobyline;
  Proc Print data=work.demog;
  Title "List of #byvar1 - #byval1";
  Var staffno age salary;
  by Gender;
  run;
```

List for GENDER- M			
Obs	STAFFNO	AGE	SALARY
1	44	23	1000.00
List for GENDER - F			
Obs	STAFFNO	AGE	SALARY
2	54	29	5000.00

SUM

Totals values of numeric variables.

If this statement is used in combination with the By statement you will get totals for all the by groups as well as a total across all by groups.

```
proc print.sas
  Proc Print data=work.demog;
  Var staffno age salary;
  by Gender;
  Id Gender;
  Sum Salary;
  run;
```

gender	staffno	age	salary
M	0099	33	21500.34
	0100	46	26600.24
	0101	47	29200.18
	0102	52	35200.15
	0103	23	18600.95
	0104	64	41900.54

M			173002.40
=====			
			2202418.37

WHERE

The where statement or where options can be applied as a filter.

```
proc print.sas
  Proc print data=work.demog (where={gender='F'});
  var staffno age gender;
  Run;

  Proc print data=work.demog;
  var staffno age gender;
  where gender='F';
  Run;
```

Obs	staffno	age	gender
2	002	26	F
5	005	23	F
7	007	34	F
8	008	38	F
9	009	56	F
10	0010	52	F
11	0011	60	F
12	0012	56	F
13	0013	46	F
14	0014	65	F
15	0015	23	F

LABEL

Use the label statement if you want to assign an alternative heading for your columns, it can be used either in the data step of the proc step. To see your label with proc print, you must specify the label option on the proc print statement.

One useful feature of the label statement is created a blank label. To do this you have to suppress the label from being printed using '00'x.

```
proc print.sas *
  Proc print data=work.demog label;
  var salary gender status grade;
  label gender='Gender'
  status='Marital Status'
  grade='Job Grade'
  salary='00'x;
  run;
```

Obs		Gender	Marital Status	Job Grade
50	14840.14	M	D	low
51	25410.14	F	M	high
52	25410.10	F	M	high
53	15410.32	F	M	high
54	17610.44	M	M	high
55	23023.33	M	W	high
56	14060.82	M	S	low
57	19050.35	M	M	high
58	13085.33	F	S	low
59	12050.38	F	S	low

PROC PRINT OPTIONS

There are a number of Proc print options that can be used to enhance your report; some of the key ones are discussed below.

NOOBS

To suppress the automatically generated Obs variable from being printed out use the option NOOBS on the proc print statement. Use in combination with the N option to get the number of observations in the report printed as well.

```
proc print.sas
  Proc print data=work.demog noobs n;
  var staffno age gender status;
  where status='P';
  Run;
```

staffno	age	gender	status
005	23	F	P
0024	55	M	P
0030	25	F	P
0049	25	M	P
0071	46	M	P
0080	30	F	P
0091	50	M	P
N = 7			

OBS

The obs option allows you to assign a label to the automatically generated obs column shown in your output, thus differing from the global option obs, which allows you to state a certain number of observations. It is a simple case of using Obs='String', on the proc print statement.

```
Proc print data=work.names
  obs='Number of Records';
Run;
```

Number of Records	staffno	age
49	0049	25
50	0050	41
51	0051	61
52	0052	21
53	0053	45

N=

Print the number of observations in the data set, in BY groups, or both, and specify explanatory text to print with the number.

```
proc print data=work.demog
  obs='Number of Records'
  p='Total Number of Records in Report= ';
  var staffno age;
run;
```

Number of Records	staffno	age
97	0097	47
98	0098	22
99	0099	33
100	0100	46
101	0101	47
102	0102	52
103	0103	23
104	0104	64
Total Number of Records in Report= 104		

ROWS

The PAGE value can reduce the number of pages in the output if the data set contains large numbers of variables and observations. However, be aware that if the data set contains a large number of variables but few observations, the PAGE value can increase the number of pages in the output.

```
Proc print data=work.names
  Rows=page;
Run;
```

WIDTH

Determines the column width for each variable. To use specify:

width=(Full, Minimum, Uniform, or UniformBy).

Using Width=Full can reduce execution time. If your data set is large, and you want a uniform report, you can save computer resources by using formats. When you specify WIDTH = UNIFORM(BY), Proc Print reads the data set twice. If each variable has a format applied Proc Print reads the data set only once.

DIRECT PATHING

Direct pathing allows you to print a SAS table to the output window without having to use a libname. The syntax will differ depending on your operating system. When using the path requires quotes, double quotes or single quotes can be used. When referencing your table you may use the file extension if you wish, but it is not required.

```
proc print data="C:\temp\client";
run;
```

JUST PROC PRINT;RUN;

If you are not aware you can submit the code:

```
Proc Print;
Run;
```

This will print the last dataset that was created to the output window.

ODS

Ods allows you to use many style options in various places within your proc print. This allows you to enhance your report using colours, fonts, and styles. A table showing this is included below. Also, an example using some of these options is included, on the next page.

Location Values	Statements
DATA	PROC PRINT, VAR, ID, SUM
HEADER	PROC PRINT, VAR, ID, SUM
TOTAL	PROC PRINT, SUM
GRANDTOTAL	PROC PRINT
BYLABEL	PROC PRINT
OBS	PROC PRINT
OBSHEADER	PROC PRINT
N	PROC PRINT
TABLE	PROC PRINT

```

F9.2 Using the STYLE option.sas *
options nobyline;
title "List for #byvar1 - #byval1";

ods listing close;
ods html body='c:\temp\demo.html';

proc print data=work.demo
      style(header)={background=white foreground=blue}
      label;
  var gender grade / style(data)={foreground=blue};
  var salary / style(data)={foreground=red};
  by status;
  id status / style(data)={foreground=blue background=white};
  sum salary / style(total)={font_weight=bold};
  label gender='Gender' grade='Job Grade'
        salary='Salary' status='Marital Status';
run;

```

Marital Status	Gender	Job Grade	Salary
W	F	high	47520.92
	F	low	10520.93
	M	high	23023.33
W			81065.18
			2202418.37

CONCLUSION

This paper aimed to highlight some key options available to enhance your report using proc print. It was a simple explanation of some forgotten features. I hope even if nothing new has been learnt, that your memory has been refreshed, and that this paper has proved that proc print still has a lot to offer, and should not be overlooked.

CONTACT INFORMATION

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

Author Name	Lynn Pope
Company:	Amadeus Software Limited
Address:	Orchard Farm, Witney Lane, Leafield, Oxon OX29 9PG
Work Phone:	01993 878287
Fax:	01993 878042
Email:	LynnPope@amadeus.co.uk
Web:	www.amadeus.co.uk

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.

