An Introduction to Proc Report

Amadeus Software Limited

ABSTRACT

Proc Report is a Base SAS procedure that allows SAS users to combine the functionality of Proc Means, Proc Tabulate and Proc Print to create listings and tabulations. It is made increasingly powerful and flexible, by including a level of functionality from the data step.

This workshop is aimed at SAS programmers who wish to learn the basic features and capabilities of this powerful reporting procedure. No prior knowledge of Proc Report will be assumed, although an understanding of SAS Procedures is helpful.

This workshop will focus on creating a report, defining appearance, adding summary statistics, and using the Output Delivery System.

INTRODUCTION TO PROC REPORT

The Proc Report procedure allows SAS users to combine the functionality of Proc Means, Proc Tabulate and Print to create listing and tabulations. By including a level of functional from the data step, Proc Report is recognised as a powerful and flexible procedure which has become the preferred report writing procedure for may SAS programmers.

All examples will be run using the nowindows option, invoking Proc report through the non-windows environment.

COLUMN STATEMENT

The column statement is used to list the columns that will appear left to right across the report. This is similar to the VAR statement in Proc Print.

Consider the example below, we are listing the columns staffno, gender, age, weight, and height, in that order.

```
Proc report data=views.demog nowindows;

column staffno gender age weight height;

run;
```

```
                      The SAS System

                                         Weight     Height
 Subject                       Age at        at         at
 Employee   Subject          Baseline  Baseline   Baseline
 Number     Gender            (Years)      (Kg)       (cm)
 001        M                      22        60         64
 002        F                      26        74         87
 003        M                      29        62         55
 004        M                      28        74         84
 005        F                      23        88         84
 006        M                      30        76         87
 007        F                      34        72         78
 008        F                      38        56         69
 009        F                      56        60         52
 0010       F                      52        69         66
```

Each Column is presented with its default width, which is taken from any associated format, character length, or length of eight if numeric.

DEFINE STATEMENT

The define statement can appear anywhere in the procedure, however, it is good practice to list the DEFINE statement after the COLUMN statement and in the same order as variables appear in the report.

```
Proc report data=views.demog nowindows;

column staffno gender age weight height;

define staffno / order       width=10;
define gender  / display     width=8;
define age     / display     width=8;
define weight  / display     width=8;
define height  / display     width=11;

run;
```

```
                       The SAS System

                                        Weight
 Subject                       Age at        at      Height at
 Employee      Subject       Baseline  Baseline       Baseline
 Number        Gender         (Years)      (Kg)           (cm)
 001           M                   22        60             64
 0010          F                   52        69             66
 0011          F                   60        71           88.5
 0012          F                   56        61             61
 0013          F                   46        76             63
 0014          F                   65        81             87
 0015          F                   23        80             78
 0016          M                   28        82             67
 0017          M                   25        76             65
 0018          M                   34        64             73
 0019          M                   36        65             61
 002           F                   26        74             87
 0020          M                   33        75           76.5
 0021          M                   40        81             66
```

The code above has a define statement for each variable in the COLUMN statement. The data is sorted by Staffno, using the order option, whilst the remaining variables are displayed as normal, and the width of each column is controlled.

ORDER

Assiging the usage of group or order will result in a variable being sorted. If the intention is not to consolidate the data, then the Order type should be chosen.

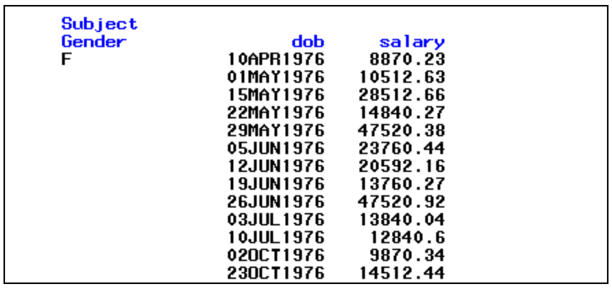
The combination of variables listed on a column statement defined as order variabes, can be considered as being similar to using a BY statement in a Proc Sort. The ORDER= option on the DEFINE statement is used to select sorting options, determining whether Proc Report orders a column by the formatted value of the data (default), the raw value, the frequency count of data times, or by the input data order.

A report may be required that presents the data sorted by gender, dob, and salary.

```
Proc report data=views.demog nowindows;

column Gender dob salary;

define Gender   / order      width=10;
define dob      / order      width=12;
define salary   / display;

run;
```

```
   Subject
   Gender                  dob      salary
   F                 01JAN1977    13840.86
                     01MAY1976    10512.63
                     02APR1977    15410.32
                     02OCT1976     9870.34
                     03DEC1977    13500.56
                     03JUL1976    13840.04
                     04DEC1976    22592.02
                     05JUN1976    23760.44
                     06NOV1976    12512.46
                     07MAY1977    13085.33
                     08OCT1977    16975.49
                     09JUL1977    12675.25
```

The output is not as desired; the order option has sorted the dates by their internally sorted values. If we add the **order=internal** option, then we get the desired result.

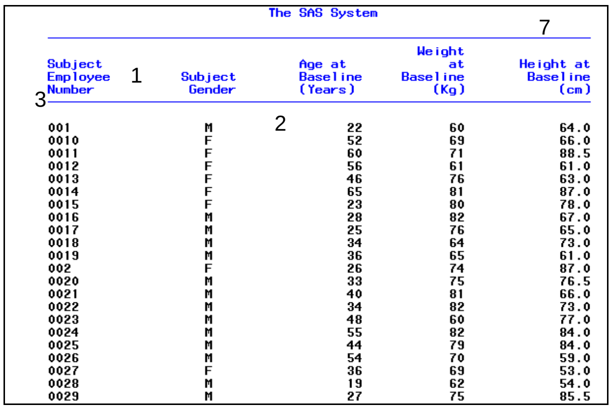


```
Subject
Gender                   dob       salary
F                  10APR1976      8870.23
                   01MAY1976     10512.63
                   15MAY1976     28512.66
                   22MAY1976     14840.27
                   29MAY1976     47520.38
                   05JUN1976     23760.44
                   12JUN1976     20592.16
                   19JUN1976     13760.27
                   26JUN1976     47520.92
                   03JUL1976     13840.04
                   10JUL1976      12840.6
                   02OCT1976      9870.34
                   23OCT1976     14512.44
```

TAKING CONTROL OF PROC REPORTS APPEARANCE

By adding just a few options we can turn output that looks similar to Proc Print listings, into professionally appearing reports**.**

```
Proc report data=views.demog nowd 1spacing=5
      2 headskip headline ls=100 ps=50 split='~';
                          3      4      5         6
column ('--' staffno gender age weight height );
           7
define staffno   / order     width=10;
define gender    / display   width=12 center;
define age       / display   width=8 left;
define weight    / display   width=8 right;
define height    / display   width=11 format=8.1;

run;
```



```
                              The SAS System
                                                                  7
     ___________________________________________________________________
                                                  Weight
     Subject                           Age at        at     Height at
     Employee  1       Subject         Baseline  Baseline   Baseline
    3Number            Gender          (Years)      (Kg)        (cm)
     ___________________________________________________________________
                                    2
     001                  M                22        60        64.0
     0010                 F                52        69        66.0
     0011                 F                60        71        88.5
     0012                 F                56        61        61.0
     0013                 F                46        76        63.0
     0014                 F                65        81        87.0
     0015                 F                23        80        78.0
     0016                 M                28        82        67.0
     0017                 M                25        76        65.0
     0018                 M                34        64        73.0
     0019                 M                36        65        61.0
     002                  F                26        74        87.0
     0020                 M                33        75        76.5
     0021                 M                40        81        66.0
     0022                 M                34        82        73.0
     0023                 M                48        60        77.0
     0024                 M                55        82        84.0
     0025                 M                44        79        84.0
     0026                 M                54        70        59.0
     0027                 F                36        69        53.0
     0028                 M                19        62        54.0
     0029                 M                27        75        85.5
```

The sample above adds seven options to control the appearance.

Firstly, the space to the left of each column is increased from the default two spaces to five (1). The column headers are enhanced by adding a solid line (3) and a blank row (2) between the column headers and the first row of data on every page.

A feature of the column statement is then used to draw a line over column headers, in this example, over all the headers (7). This is achieved by grouping together the columns to be spanned with parentheses and adding the header text before the first column name. The example shows how column may be overlined with a solid line , i.e. by specifying two hyphens with no text.

The line and page sizes for the Proc Report output can be controlled on the procedure statement (4 & 5); these settings override the equivalent settings which may be set with an OPTIONS statement.

The split character is altered (6) from the default '#'. Proc Report treats the split symbol as a marker for wrapping lines of data in the body of the report, or column labels in the header area of the report. The split symbol will not appear in the output, therefore it is common practice to use a character uncommonly used in reporting, to avoid accidental non-display and wrapping of data.

Common alternative splits characters are:

| ¬ ~

Attention must always be given to the contents of the raw data when using a split character to ensure no conflicts will occur. The function of the split character can be stopped completely by setting the Proc Report options:

Split=""

With no spaces between the quotes.

DEFINING THE APPEARANCE OF INDIVIDUAL COLUMNS

```
Proc report data=views.demog nofs headskip
                 headline split='|';

column ('--' gender dob age weight );

define gender    / order width=10  right;  1
define dob       / "Date of" "Subject's Birth"  2
                   order order=internal width=12 center;
define age       / order "Age"   width=10 left flow;   4
define weight    / display "Subject's | Weight"
                   width=12 left flow;    3

break after age / skip;   5

run;
```

The alignment of any column can be specified with Proc Report using any of the LEFT, CENTER or RIGHT keywords on the DEFINE statement, as shown in this example (1) above.

By default Proc Report displays the label associated with a variable for the column header (unless OPTIONS NOLABEL has been specified). However, any text can be specified as the column header. There are two ways of determining where the label wraps in the column header as shown above. Individual strings may be used (2) or the split character (3) can be embedded in the text.

When character variables form part of a report the text to be printed may frequently be longer than the column width available to the report. To avoid truncating the text, the FLOW option can be specified (4) which wraps the text string onto the next line within the column.

Finally in this example, the BREAK statement is introduced. Where listings have many rows for a grouped item, the presentation can be improved by adding a blank row between groups of data. This is achieved through the BREAK statement (5) and the SKIP option, which skips a line in the output every time the value in the break variable changes. Only variables defined as GROUP or ORDER can be referred to in the BREAK statement.

In this example age is the break variable with the skip option, the after option specifies where the blank row is written relative to each break point, as can be seen in the partial output below.

```
                              The SAS System

                  Date of
     Subject     Subject's                          Subject's
      Gender       Birth         Age                 Weight

           F     10APR1976              26               74

                 01MAY1976              23               88

                 15MAY1976              34               72

                 22MAY1976              38               56

                 29MAY1976              56               60

                 05JUN1976              52               69

                 12JUN1976              60               71

                 19JUN1976              56               61

                 26JUN1976              46               76
```

BREAK STATEMENT

Any variable in the column statement, defined as a GROUP or ORDER variable can be used in a BREAK statement.

Break statements can be used to separate logical groups of data with a single line, as seen in the previous example. They can also be used to insert page breaks, summary statistics and customised information; either at the beginning and end of groups or reports.

The _BREAK_ internal variable is created whenever the BREAK or RBREAK statements are used. It's value contains the name of the breaking variable at the observation when break occurs. We will see more of BREAK and RBREAK in summarising data section.

SUMMARISING DATA

Proc Report can produce many of the summary statistics that procedures such as MEANS, SUMMARY and TABULATE can.

The summary statistics requested must always be associated with a variable in the input data set, except for N, and are presented for each level of group variable to the left of the statistics in the column statement.

The number of non-missing observations in the input data set statistic, N, can be requested anywhere, as it does not require the value of a column to compute any statistics.

Important Note: Proc Report consolidates or sorts data when a column is defined as GROUP or ORDER. If one or more values of a GROUP or ORDER variable are missing, then the whole row will be excluded from the report unless the MISSING option is used in the Proc Report or Define statements. To ensure that all data are always presented in a report it is strongly suggested that the MISSING options always placed in the Proc Report statement.

```
Proc report data=views.demog nofs missing
            headskip headline;                        1

column ('--' treatment age,     2
        ('--' n mean median std min max ));
                          3
define treatment     / group left;
define age           / analysis;
define n             / 'n';        4
define mean          / 'Mean' f=8.1;
define median        / 'Median' f=8.1;
define std           / 'SD' f=8.2;
define min           / 'Minimum';
define max           / 'Maximum';

break after treatment / skip;
rbreak after / ol summarize;
                                        5
run;
```

```
                                   The SAS System

                                      Age at Baseline (Years)
 treatment                 n      Mean    Median        SD   Minimum   Maximum

 Active                   20      37.2      33.5     13.64        22        65
 Non Active               30      38.7      40.0     11.97        18        61
 Not Known                54      38.2      38.5     12.19        18        64

                    --------  --------  --------  --------  --------  --------
                         104      38.2      37.5     12.30        18        65
```

We have specified the missing option (1) to ensure no data is omitted from our report. We are using a comma (2) to tell the procedure to nest the following column (or columns grouped by parentheses) within the age variable.

The nested columns requested are the statistics n, mean, median, standard deviation, minimum and maximum (3). The column Treatment is defined as a GROUP variable, so data are consolidated and AGE is defined as an analysis variable (4). The overall summaries are requested after the report (5) with the RBREAK statement and the summaries will be displayed with an over-line via the OL option.

PERCENTAGES

Percentages can be added to report, the following example demonstrates how this can be done.

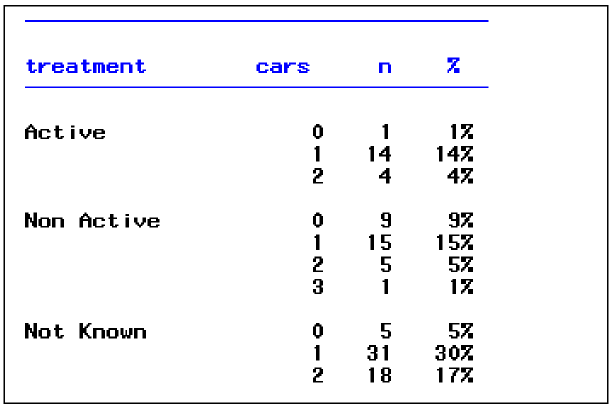
```
Proc report data=views.demog nofs missing
         headskip headline;

column ('--' treatment cars, (n pctsum));


define treatment    / group left;
define cars         / group left;
define n            / 'n' width=3 f=3.0;
define pctsum       / '%' width=5 f=percent5.0 center;

break after treatment / skip;

run;
```

```
ERROR: A GROUP variable appears above or below other report items.
       Name                               Usage
       ---------------------------------- --------
       cars                               GROUP
       n                                  STATISTIC
ERROR: A GROUP variable appears above or below other report items.
       Name                               Usage
       ---------------------------------- --------
       cars                               GROUP
       pctsum                             STATISTIC
NOTE: The SAS System stopped processing this step because of errors.
NOTE: PROCEDURE REPORT used:
      real time           0.00 seconds
      cpu time            0.00 seconds
```

Although the summary statistics are associated with the numeric variable AGE, this varibale is defined as a GROUP variable and nested statistics require an Analysis variable to allow the count and percentage to be determined.

By introducing a dummy variable X, where each record has the value 1, and using this variable on the column statement, we can now produce the statistics required and allow the user to achieve the desired report.

```
Proc report data=views.demog(where=(cars ne .))
         nofs missing headskip headline;

column ('--' treatment cars x, (n pctsum));


define treatment    / group left;
define cars         / group left;          1
define x            / analysis ' ';
define n            / 'n' width=3 f=3.0;
define pctsum       / '%' width=5 f=percent5.0 center;

break after treatment / skip;

run;
```



```
treatment                cars       n      %

Active                      0       1     1%
                            1      14    14%
                            2       4     4%

Non Active                  0       9     9%
                            1      15    15%
                            2       5     5%
                            3       1     1%

Not Known                   0       5     5%
                            1      31    30%
                            2      18    17%
```

To prevent the label X from being displayed in the report we place a ' ' on the define statement (1).

A possible problem is that Proc Report has produced the percentage based on the sum of the counts across all treatment groups.

You cannot show percentages based on the sum of individual groups of data with Proc Report statistics. Also, it is not possible to list the summary statistics in the column and the group's column headers. Therefore you may have to pre-summarise data with another Base SAS summarising procedure.

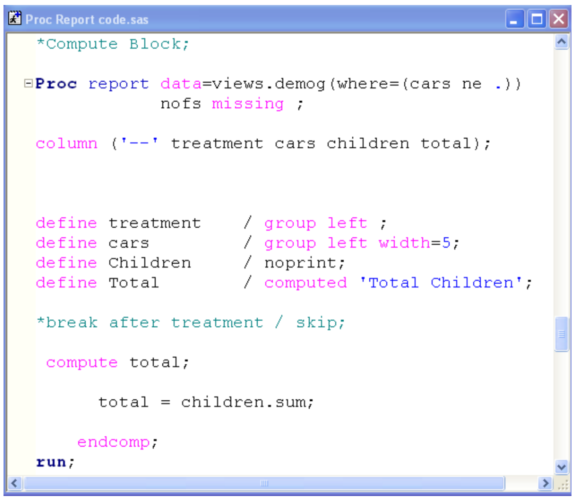
COMPUTE BLOCKS

Compute blocks are areas of Proc report code, which allow most data step and other programming code to contribute towards the building of a report. A compute block may be directly associated with a variable in the column statement, or with a location such as: before or after group of data in a group or order column, at the top or bottom of each page.

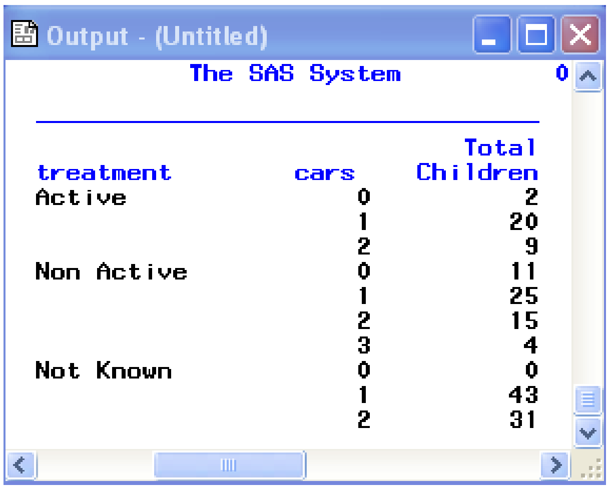
The following may be used in a compute block:

%INCLUDE	GOTO	RETURN
Variable assignment	IF-THEN-ELSE	SELECT
CALL functions	LENGTH	Comments
DO-END (all forms)	LINK	All data step Functions

Below we have an example of applying a compute block which shows how text can be written out above (or below) the report as a whole for each page. It also shows how groups of data can be presented as section headings rather than in a column of its own:



```
*Compute Block;

Proc report data=views.demog(where=(cars ne .))
            nofs missing ;

column ('--' treatment cars children total);


define treatment    / group left ;
define cars         / group left width=5;
define Children     / noprint;
define Total        / computed 'Total Children';

*break after treatment / skip;

 compute total;

     total = children.sum;

   endcomp;
run;
```



```
                  The SAS System                    0

treatment               cars       Total
                                 Children
Active                     0           2
                           1          20
                           2           9
Non Active                 0          11
                           1          25
                           2          15
                           3           4
Not Known                  0           0
                           1          43
                           2          31
```

OUTPUTTING WITH ODS

The output produced by Proc Report can be sent to different outputs using ODS, where it can be influenced by styles. Attributes such as font face to the amount of white space that appears around the text in each statement can be controlled.

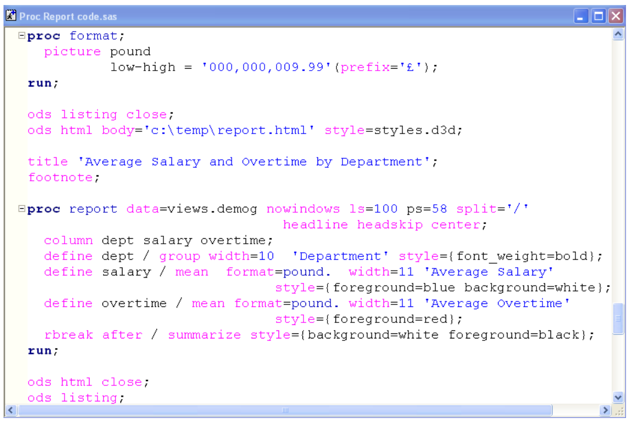
HTML

Proc Report has style options on the:

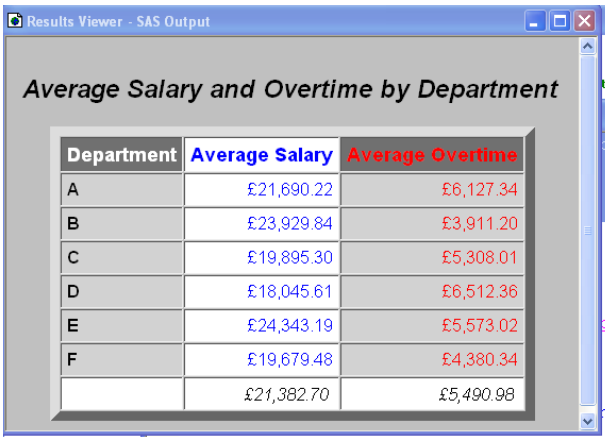
- PROC statement
- BREAK statement
- COMPUTE statement
- CALL DEFINE statement
- DEFINE statement
- RBREAK statement

Syntax:

```
STYLE <(location(s))>=<style-element-
name><[style-attribute-specification(s)]>
```



```
proc format;
  picture pound
          low-high = '000,000,009.99'(prefix='£');
run;

ods listing close;
ods html body='c:\temp\report.html' style=styles.d3d;

title 'Average Salary and Overtime by Department';
footnote;

proc report data=views.demog nowindows ls=100 ps=58 split='/'
                               headline headskip center;
  column dept salary overtime;
  define dept / group width=10  'Department' style={font_weight=bold};
  define salary / mean  format=pound.  width=11 'Average Salary'
                              style={foreground=blue background=white};
  define overtime / mean format=pound. width=11 'Average Overtime'
                              style={foreground=red};
  rbreak after / summarize style={background=white foreground=black};
run;

ods html close;
ods listing;
```


Results Viewer - SAS Output

Average Salary and Overtime by Department

Department	Average Salary	Average Overtime
A	£21,690.22	£6,127.34
B	£23,929.84	£3,911.20
C	£19,895.30	£5,308.01
D	£18,045.61	£6,512.36
E	£24,343.19	£5,573.02
F	£19,679.48	£4,380.34
	£21,382.70	£5,490.98

TRAFFIC LIGHTING EFFECTS

The technique of applying colours to data depending on the value of each datum is known as 'traffic lighting'.

In SAS this is achieved by defining a format and applying it through an ODS foreground style to a column of data.

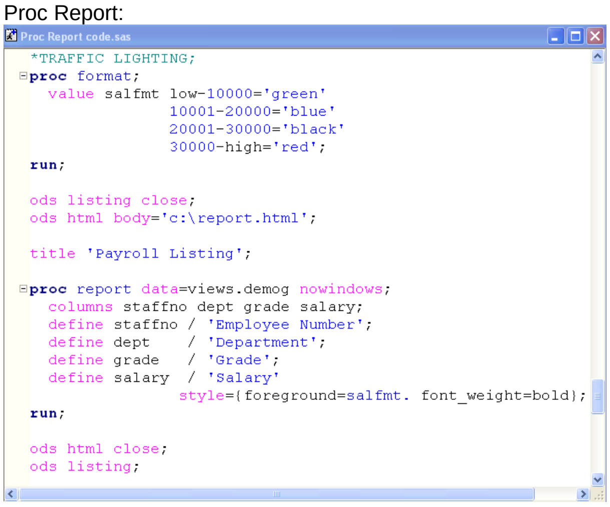
Syntax:

style={foreground=*format_name*.}

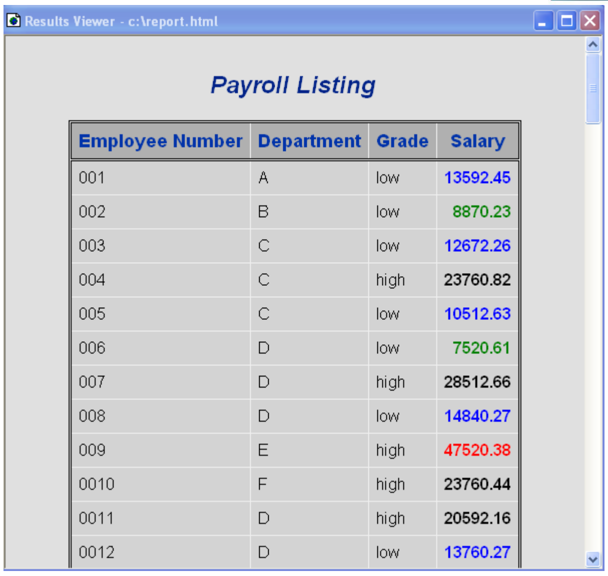
where *format_name* is the name of a character or numeric format, depending on the column type.

USING TRAFFIC LIGHTING WITH PROC REPORT

The following example shows traffic lighting being used with Proc Report:


Proc Report code.sas

```
*TRAFFIC LIGHTING;
proc format;
   value salfmt low-10000='green'
                10001-20000='blue'
                20001-30000='black'
                30000-high='red';
run;

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

title 'Payroll Listing';

proc report data=views.demog nowindows;
   columns staffno dept grade salary;
   define staffno / 'Employee Number';
   define dept    / 'Department';
   define grade   / 'Grade';
   define salary  / 'Salary'
                 style={foreground=salfmt. font_weight=bold};
run;

ods html close;
ods listing;
```


Results Viewer - c:\report.html

Payroll Listing

Employee Number	Department	Grade	Salary
001	A	low	13592.45
002	B	low	8870.23
003	C	low	12672.26
004	C	high	23760.82
005	C	low	10512.63
006	D	low	7520.61
007	D	high	28512.66
008	D	low	14840.27
009	E	high	47520.38
0010	F	high	23760.44
0011	D	high	20592.16
0012	D	low	13760.27

RTF

Sending Proc Report output to a Word document, requires two additional lines of code.

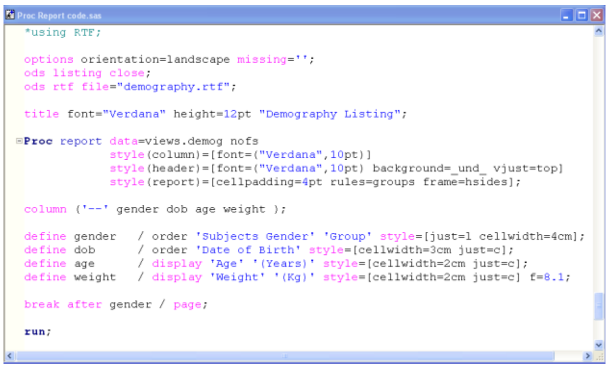
ODS RTF FILE="*filename*.rtf" ;

PROC REPORT…

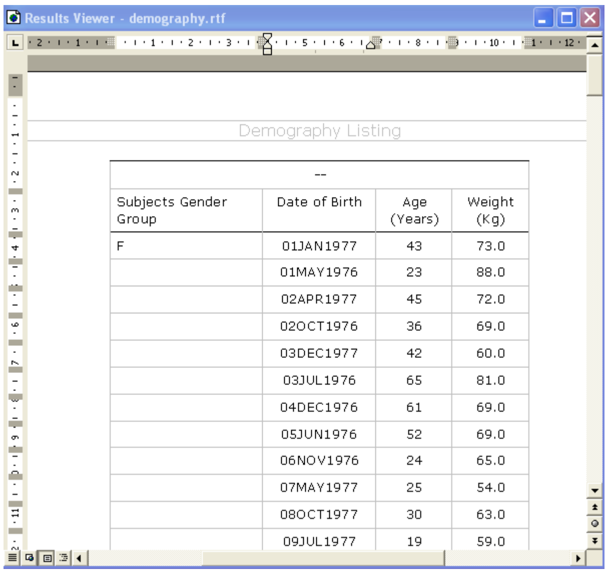
ODS RTF CLOSE;

This first line of code opens the Rich Text Format (RTF) destination, which is a format recognised by Microsoft Word.

Any procedures which generate output are then called, followed by the final ODS RTF CLOSE statement, which actually saves the contents of the RTF file to disk.


Proc Report code.sas

```
*using RTF;

options orientation=landscape missing='';
ods listing close;
ods rtf file="demography.rtf";

title font="Verdana" height=12pt "Demography Listing";

Proc report data=views.demog nofs
            style(column)=[font=("Verdana",10pt)]
            style(header)=[font=("Verdana",10pt) background=_und_ vjust=top]
            style(report)=[cellpadding=4pt rules=groups frame=hsides];

column ('--' gender dob age weight );

define gender   / order 'Subjects Gender' 'Group' style=[just=l cellwidth=4cm];
define dob      / order 'Date of Birth' style=[cellwidth=3cm just=c];
define age      / display 'Age' '(Years)' style=[cellwidth=2cm just=c];
define weight   / display 'Weight' '(Kg)' style=[cellwidth=2cm just=c] f=8.1;

break after gender / page;

run;
```



Demography Listing

--			
Subjects Gender Group	Date of Birth	Age (Years)	Weight (Kg)
F	01JAN1977	43	73.0
	01MAY1976	23	88.0
	02APR1977	45	72.0
	02OCT1976	36	69.0
	03DEC1977	42	60.0
	03JUL1976	65	81.0
	04DEC1976	61	69.0
	05JUN1976	52	69.0
	06NOV1976	24	65.0
	07MAY1977	25	54.0
	08OCT1977	30	63.0
	09JUL1977	19	59.0

The SAS System cannot currently append to RTF files and overwrites any file that may already exist. If a SAS program is attempting to overwrite an existing file and that file has been left open by a previous process, an error message will appear in the log and the file will not be recreated. Ensure that all RTF files are closed before submitting code to recreate them.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact Amadeus at:

Company:	Amadeus Software Limited
Address:	The Old School Hall, 11 Wesley Walk, Witney, Oxon, OX28 6ZJ
Work Phone:	+44 (0) 1993 848010
Email:	info@amadeus.co.uk
Web:	www.amadeus.co.uk