A Gentle Introduction to the Powerful REPORT Procedure
Ben Cochran, The Bedford Group, Raleigh, NC

ABSTRACT
When the REPORT Procedure was first introduced by SAS with the advent of Version 6, most of the SAS world was mainframe users. This new procedure brought with it a great deal of power and flexibility that added much strength to SAS’ arsenal of report generating procedures. It had powerful formatting, summarizing, and analysis capabilities that made it easier to create anything from a simple listing to a very complex report. However, some critics have stated that, while it has awesome features, it looks like a mainframe report. That was true until SAS released the Output Delivery System (ODS) experimentally in Version 7, and in production mode in Version 8. If the REPORT procedure was a cake, then ODS delivers the icing to generate truly beautiful reports. This paper offers a quick overview of the types of reports that can be generated with Proc REPORT, and how to add some of the ODS features to deliver stunning reports.

INTRODUCTION
This paper is intended for users who have not worked with the REPORT procedure, or at least not lately. First, a look is taken at the general syntax, and then a series of reports will be generated starting with a very simple report and then working up to more complex reports.

TYPICAL SYNTAX
The REPORT procedure is made up of a PROC statement, a COLUMN statement, several DEFINE statements, and other optional statements that help with calculations and summarizations.

```
PROC REPORT data= SAS-data-set options ;
    COLUMNS variable_1 …. variable_n;
    DEFINE variable_1;
    DEFINE variable_2;
    . . .
    DEFINE variable_n;
    COMPUTE blocks
    BREAK … ;
    RBREAK … ;
RUN;
```

COLUMNS statement defines which columns appear in the report, and their order.
DEFINE statements declare how variables are to be used in the report.
COMPUTE blocks allow calculations to be performed in the report.
BREAK / RBREAK statements allow summarization and some kinds of formatting at certain places in the report.

The REPORT procedure also has many options that can be used. Some of the most often used options are:

- **DATA=** specifies the dataset to be processed,
- **PROMPT** invokes the prompting mode, sort of like a wizard
- **NOWINDOWS** suppresses the REPORT window and directs the report to the output window
- **REPORT =** specifies a stored report to be used in generating a new report
- **OUTREPT=** names a location to store the report
- **OUT=** creates a SAS data set
- **HEADLINE** creates a horizontal line between the column headers and the body of the report
- **HEADSKIP** creates a blank line between the column headers and the body of the report

Now we are ready to take these statements and options and begin creating reports using the SASHELP.CLASS data set.
Task 1: Generate a basic Report using the REPORT procedure.

```sas
proc report data=sashelp.class nowindows;
    columns name sex age height weight;
    define name / display 'Name' width=10;
    define sex / display 'Gender' width=6;
    define age / display 'Age' width=4;
    define height / analysis 'Height' format=8.1;
    define weight / analysis 'Weight' format=8.1;
run;
```

Program 1.

Notice the DEFINE statements. The term following the ‘/’ specifies the way the REPORT procedure uses the column. Columns can be defined as:

- **GROUP** - puts observations into categories
- **DISPLAY** - displays values for each observation
- **ANALYSIS** - contributes values to a calculation or statistic
- **ORDER** - defines the order of the report rows
- **ACROSS** - creates columns for each of its values
- **COMPUTED** - its values are created in a COMPUTE block.

Now, let's look at the output created from the above program.

```
   The SAS System

   Name    Gender Age Height  Weight
    Alfred  M    14    69.0   112.5
      Alice F    13    56.5    84.0
  Barbara  F    13    65.3    98.0
       Carol F    14    62.8   102.5
       Henry M    14    63.5   102.5
       James M    12    57.3    83.0
        Jane F    12    59.8    84.5
       Janet F    15    62.5   112.5
     Jeffrey M    13    62.5    84.0
         John M    12    59.0    99.5
        Joyce F   11    51.3    50.5
         Judy F    14    64.3    90.0
```

Output 1 – Partial PROC REPORT output.

At first glance, this looks a little like PROC PRINT output without the OBS column. Aesthetically, the output could use some improvement, so let's enhance the report.

Task 2: Add a Compute Block and a total row at the end of the report.

```sas
proc report data=sashelp.class nowindows headline headskip;
    columns name sex age height weight ratio;
    define name / display 'Name' width=10;
    define sex / display 'Gender' width=6;
    define age / display 'Age' width=4;
    define height / analysis mean 'Height' format=8.1;
    define weight / analysis mean 'Weight' format=8.1;
    define ratio / computed format=6.2;
    compute ratio;
        ratio = height.mean / weight.mean;
    endcompute;
run;
```

```
   The SAS System

   Name    Gender Age Height  Weight  Ratio
    Alfred  M    14    69.0   112.5  
      Alice F    13    56.5    84.0  
  Barbara  F    13    65.3    98.0  
       Carol F    14    62.8   102.5  
       Henry M    14    63.5   102.5  
       James M    12    57.3    83.0  
        Jane F    12    59.8    84.5  
       Janet F    15    62.5   112.5  
     Jeffrey M    13    62.5    84.0  
         John M    12    59.0    99.5  
        Joyce F   11    51.3    50.5  
         Judy F    14    64.3    90.0  
```

At first glance, this looks a little like PROC PRINT output without the OBS column. Aesthetically, the output could use some improvement, so let's enhance the report.
Program 2.

You can see the new column, \textit{ratio}, as well as the averages for \textit{Height}, \textit{Weight} and \textit{ratio} at the end of the report.

\begin{tabular}{|l|c|c|c|c|}
\hline
\textbf{Name} & \textbf{Gender} & \textbf{Age} & \textbf{Height} & \textbf{Weight} & \textbf{ratio} \\
\hline
Alfred & M & 14 & 69.0 & 112.5 & 0.61 \\
Alice & F & 13 & 56.5 & 84.0 & 0.67 \\
Barbara & F & 13 & 65.3 & 98.0 & 0.67 \\
Carol & F & 14 & 62.8 & 102.5 & 0.61 \\
Henry & M & 14 & 63.5 & 102.5 & 0.62 \\
James & M & 12 & 57.3 & 83.0 & 0.69 \\
Jane & F & 12 & 59.8 & 84.5 & 0.71 \\
Janet & F & 15 & 62.5 & 112.5 & 0.56 \\
Jeffrey & M & 13 & 62.5 & 84.0 & 0.74 \\
John & M & 12 & 59.0 & 99.5 & 0.59 \\
Joyce & F & 11 & 51.3 & 50.5 & 1.02 \\
Judy & F & 14 & 64.3 & 90.0 & 0.71 \\
Louise & F & 12 & 56.3 & 77.0 & 0.73 \\
Mary & F & 15 & 66.5 & 112.0 & 0.59 \\
Philip & M & 16 & 72.0 & 150.0 & 0.48 \\
Robert & M & 12 & 64.8 & 128.0 & 0.51 \\
Ronald & M & 15 & 67.0 & 133.0 & 0.50 \\
Thomas & M & 11 & 57.5 & 85.0 & 0.68 \\
William & M & 15 & 66.5 & 112.0 & 0.59 \\
\hline
\multicolumn{4}{|c|}{-------} & \multicolumn{1}{|c|}{-------} & \multicolumn{1}{|c|}{-------} \\
\hline
\multicolumn{4}{|c|}{62.3} & \multicolumn{1}{|c|}{100.0} & \multicolumn{1}{|c|}{0.62} \\
\hline
\end{tabular}

Output 2.

Notice the columns defined as ANALYSIS and COMPUTED have \textsc{means} generated with the \textsc{rbreak} statement.
The \textsc{dul} stands for double underline and \textsc{dol} stands for double overline. Why is there no \textsc{average} for \textit{age} at the end of the report?

\textbf{Task 3}: Generate subtotals and add a blank line after each gender.

\begin{verbatim}
proc report data=sashelp.class nowindows headline headskip;
columns sex name age height weight ratio;
define sex / group Gender width=10;
define name / display Name width=6;
define age / analysis mean Age width=4;
define height / analysis mean Height format=8.1;
define weight / analysis mean Weight format=8.1;
define ratio / computed format=6.2;
compute ratio;
   ratio = height.mean / weight.mean;
endcompute;
break after sex / skip summarize dol dul;
run;
\end{verbatim}

Program 3.

Notice the \textsc{break} statement and its options. Also notice that the \textsc{break} statement has been dropped.
Look at the definition of \textit{age}; it has been changed from display to analysis. Also note the addition of the \textsc{mean}
statistic.

Notice that the order of the variables have rearranged. \textit{Sex} and \textit{name} have been switched. \textit{Sex} has been redefined
as a \textsc{group} variable.
Task 4: Calculate Percentages with PROC REPORT.

In the next task, calculate the percentages that each row represents for each group. In this case, the percentages are to add up to 100 for each group. One sub-task that needs to be done is to add up weight for each group, then divide each person’s weight by the group total.

```sas
title 'Calculating Percentages with Proc Report';
proc report data=sashelp.class nowindows headline headskip;
columns sex name height weight weight_pct;
define sex / group 'Gender' width=10;
define name / display 'Name' width=6;
define height / analysis mean 'Height' format=8.1;
define weight / analysis 'Weight' format=8.1;
define weight_pct / '% of Weight' format=percent8.2;
*------ Calculations for each row -----------------------------*;
compute weight_pct;
  weight_pct = weight.sum / weight_sum;
endcompute;
*-------------------------------------------------------------*;
compute before sex;
  weight_sum = weight.sum;
endcompute;
bbreak after sex / skip summarize dol dul;
run;
```

Program 4.
Notice the two compute blocks. The second one creates `weight_sum` which is the total weight for each group. The second compute block calculates each person’s percent of the group. Also notice that the statistic has been dropped from the DEFINE statement for `weight`.

<table>
<thead>
<tr>
<th></th>
<th>Name</th>
<th>Height</th>
<th>Weight</th>
<th>% of Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>F</strong></td>
<td>Alice</td>
<td>56.5</td>
<td>84.0</td>
<td>10.36%</td>
</tr>
<tr>
<td></td>
<td>Barbar</td>
<td>65.3</td>
<td>98.0</td>
<td>12.08%</td>
</tr>
<tr>
<td></td>
<td>Carol</td>
<td>62.8</td>
<td>102.5</td>
<td>12.64%</td>
</tr>
<tr>
<td></td>
<td>Jane</td>
<td>59.8</td>
<td>84.5</td>
<td>10.42%</td>
</tr>
<tr>
<td></td>
<td>Janet</td>
<td>62.5</td>
<td>112.5</td>
<td>13.87%</td>
</tr>
<tr>
<td></td>
<td>Joyce</td>
<td>51.3</td>
<td>50.5</td>
<td>6.23%</td>
</tr>
<tr>
<td></td>
<td>Judy</td>
<td>64.3</td>
<td>90.0</td>
<td>11.10%</td>
</tr>
<tr>
<td></td>
<td>Louise</td>
<td>56.3</td>
<td>77.0</td>
<td>9.49%</td>
</tr>
<tr>
<td></td>
<td>Mary</td>
<td>66.5</td>
<td>112.0</td>
<td>13.81%</td>
</tr>
</tbody>
</table>

| **M**         | Alfred| 69.0   | 112.5  | 10.33%      |
|               | Henry | 63.5   | 102.5  | 9.41%       |
|               | James | 57.3   | 83.0   | 7.62%       |
|               | Jeffre| 62.5   | 84.0   | 7.71%       |
|               | John  | 59.0   | 99.5   | 9.13%       |
|               | Philip| 72.0   | 150.0  | 13.77%      |
|               | Robert| 64.8   | 128.0  | 11.75%      |
|               | Ronald| 67.0   | 133.0  | 12.21%      |
|               | Thomas| 57.5   | 85.0   | 7.80%       |
|               | Willia| 66.5   | 112.0  | 10.28%      |

Output 4.

Notice the `weight` column. The total weight is displayed for it, while the average is displayed for the `height` column. Is the subtotal for `weight` really realistic for this report? Maybe not, but it is needed here to calculate the percents. What we need is a way to have access to the sum of `weight`, but not display it in the report. Is there a way to do this in PROC REPORT?

**Task 5.** Enhance the report by not displaying the WEIGHT column.

```sas
proc report data=sashelp.class nowindows headline headskip;
columns sex name height weight weight_pct;
define sex / group 'Gender' width=10;
define name / display 'Name' width=6;
define height / analysis mean 'Height' format=8.1;
define weight / analysis noprnt format=8.1;
define weight_pct / ' % of Weight' format=percent8.2;
*--------- Calculations for each row ----------*;
compute weight_pct;
   weight_pct = weight.sum / weight_sum;
endcompute;
```

Partial Program 5.
Notice the DEFINE statement for *weight*; there is a NOPRINT option that ‘hides’ the weight column from the output.

<table>
<thead>
<tr>
<th>Gender</th>
<th>Name</th>
<th>Height</th>
<th>Weight %</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>Alice</td>
<td>56.5</td>
<td>10.36%</td>
</tr>
<tr>
<td></td>
<td>Barbara</td>
<td>65.3</td>
<td>12.08%</td>
</tr>
<tr>
<td></td>
<td>Carol</td>
<td>62.8</td>
<td>12.64%</td>
</tr>
<tr>
<td></td>
<td>Jane</td>
<td>59.8</td>
<td>10.42%</td>
</tr>
<tr>
<td></td>
<td>Janet</td>
<td>62.5</td>
<td>13.87%</td>
</tr>
<tr>
<td></td>
<td>Joyce</td>
<td>51.3</td>
<td>6.23%</td>
</tr>
<tr>
<td></td>
<td>Judy</td>
<td>64.3</td>
<td>11.10%</td>
</tr>
<tr>
<td></td>
<td>Louise</td>
<td>56.3</td>
<td>9.49%</td>
</tr>
<tr>
<td></td>
<td>Mary</td>
<td>66.5</td>
<td>13.81%</td>
</tr>
</tbody>
</table>

Partial Output 5.

Notice the absence of the *weight* column. But what if we wanted a column showing *weight*, and subtotals showing the average *weight* for each group in addition to the weight % column. Can this be done? What would be needed in the Proc REPORT step?

**Task 6.** Add a column to display *weight* and its AVERAGE for each group.

```
proc report data=sashelp.class nowindows headline headskip;
   define sex / group  'Gender' width=10;
   define name / display  'Name' width=6;
   define height / analysis mean 'Height' format=8.1;
   define weight / analysis noprint format=8.1;
   define weight2 / analysis mean format=8.1;
   define weight_pct / '% of Weight' format=percent8.2;
   *---------- Calculations for each row ---------------*;
   compute weight_pct;
      weight_pct = weight.sum / weight_sum;
   endcompute;
   *---------------------------------------------*;
   compute before sex;
      weight_sum = weight.sum;
   endcompute;
   break after sex / skip summarize dol dul;
run;
```

Program 6.

Notice the ‘*weight = weight2*’ in the COLUMNS statement. This creates an alias for *weight*, giving this report two weight columns, *weight* and *weight2*, each with its own definition. The statistic for *weight* defaults to *sum* and is used in calculating percentages, but is not displayed in the report because of the NOPRINT definition. *Weight2* is used to display the averages for each value of the *sex* variable.
The above output display only the first half of the report.

Up to this point, the report has only shown a single statistic for each column. What if you wanted to see more than one statistic for each variable. Can this be done with PROC REPORT?

**Task 7. Calculate multiple statistics for each analysis variable.**

In order to calculate multiple statistics for each column, we are going to have to manipulate the data. The trick that makes this work is to have a duplicate group column for each statistic. In other words, in this example, two statistics are displayed (mean and median), so we need to have two columns that are identical. Notice the following DATA step that manipulates the data.

```
data prep;
  length name $16;
  set sashelp.class;
  gender = sex;
run;
```

Once the data is manipulated, it can now be processed by the REPORT procedure.

```
proc report data=prep nowindows headline headskip;
  columns sex gender name weight=weight mn weight=weight_md;
  define sex / group 'Gender' width=6;
  define gender / group noprint;
  define name / group 'Name' width=16;
  define weight / analysis format=8.2
      compute after sex;
      name='Median Weight';
      weight.sum = weight_md;
    endcompute;
*-----------------------------------------------------------------------*;
  compute after gender;
      name='Average Weight';
      weight.sum = weight_mn;
    endcompute;
*-----------------------------------------------------------------------*;
  break after sex / skip summarize
dol;
  break after gender / summarize dol;
run;
```

Program 7.
Notice there is the `weight` column and two aliases: `weight_md` and `weight_mn`. `Weight` is displayed, but `weight_mn` and `weight_md` have the NOPRINT definition.

You can only calculate one row (statistic) for each column in a compute block. You can only have one compute block for each variable (column). So, if you want two statistics for each column, then you need two compute blocks, but they have to be for different variables. Hence, in the new data set, we have the variable `sex`, and the variable `gender`. Both have the same values for each row. We have a compute block for each variable, one for `sex`, and one for `gender`. Notice the assignment statement for `name` in each. Notice the output below:

```
Gender   Name       Weight
----      --------   ------
   F         Alice    84.00
            Barbara  98.00
            Carol    102.50
            Jane     84.50
            Janet    112.50
            Joyce    50.50
            Judy     90.00
            Louise   77.00
            Mary     112.00
---       =============
 F Average Weight  90.11
---       =============
 F Median Weight   90.00
---       =============
```

Output 7. Partial REPORT Procedure output

**Task 8.** Use conditional logic to create multiple summary rows at the end of the report.

In this example, we want to see multiple averages calculated for the same column. An overall average, a female average, and a male average for the `weight` variable will be shown. In order to accomplish this task, the data set will have to be pre-processed again. The following DATA step does this.

```plaintext
data prep2;
  length name $15.
  set sashelp.class;
  f = 1;
  m = 1;
  goal = 99;
run;
```

You can use what some programmers call 'holding variables' in PROC REPORT. In the following example, these variables are put to work in the compute blocks.

```
proc report data=prep2 nowindows;
  columns m f goal sex name weight weight=f_weight weight=m_weight;
  define name  / display width=12;
  define sex   / display width=12;
  define m     / group noprint;
  define f     / group noprint;
  define goal  / group noprint;
  define weight / analysis mean format=5.1;
  define f_weight / sum noprint;
  define m_weight / sum noprint;
*-----------------------------------------------------*;
```

Partial REPORT Procedure, Program 8.
The variable `wholdm` and `wholdf` are 'holding variables', you can use them in calculations, but they do not appear in the report. Since the variables `f`, `m`, and `goal` have the same values for all observations, the BREAK statements have the same effect as the RBREAK statement; appending rows at the end of the report. Notice the output.

<table>
<thead>
<tr>
<th>Sex</th>
<th>name</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>Alfred</td>
<td>112.5</td>
</tr>
<tr>
<td>F</td>
<td>Alice</td>
<td>84.0</td>
</tr>
<tr>
<td>F</td>
<td>Barbara</td>
<td>98.0</td>
</tr>
<tr>
<td>F</td>
<td>Carol</td>
<td>102.5</td>
</tr>
<tr>
<td>M</td>
<td>Henry</td>
<td>102.5</td>
</tr>
<tr>
<td>M</td>
<td>James</td>
<td>83.0</td>
</tr>
<tr>
<td>F</td>
<td>Jane</td>
<td>84.5</td>
</tr>
<tr>
<td>F</td>
<td>Janet</td>
<td>112.5</td>
</tr>
<tr>
<td>M</td>
<td>Jeffrey</td>
<td>84.0</td>
</tr>
<tr>
<td>M</td>
<td>John</td>
<td>99.5</td>
</tr>
<tr>
<td>F</td>
<td>Joyce</td>
<td>50.5</td>
</tr>
<tr>
<td>F</td>
<td>Judy</td>
<td>90.0</td>
</tr>
<tr>
<td>F</td>
<td>Louise</td>
<td>77.0</td>
</tr>
<tr>
<td>F</td>
<td>Mary</td>
<td>112.0</td>
</tr>
<tr>
<td>M</td>
<td>Philip</td>
<td>150.0</td>
</tr>
<tr>
<td>M</td>
<td>Robert</td>
<td>128.0</td>
</tr>
<tr>
<td>M</td>
<td>Ronald</td>
<td>133.0</td>
</tr>
<tr>
<td>M</td>
<td>Thomas</td>
<td>85.0</td>
</tr>
<tr>
<td>M</td>
<td>William</td>
<td>112.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Goal</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>99.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>90.0</td>
<td>Female Avg</td>
</tr>
<tr>
<td></td>
<td>109.0</td>
<td>Male Avg</td>
</tr>
<tr>
<td></td>
<td>100.0</td>
<td>Overall Avg</td>
</tr>
</tbody>
</table>

Output 8.
Task 9. Use ODS to enhance the report.

With the Output Delivery System, you can send your output to any number of locations as well as create special formatting. In the next example, the last PROC REPORT step is ‘sandwiched’ between basic ODS statements.

```
ods listing close;
ods rtf file = 'c:\sugi30.rtf';
   previous proc report step
ods pdf close;
```

The first ODS statement closes the output window, while the second ODS statement opens a file for the output. The third one closes the rtf file.

Output 9. ODS.

Notice the fonts of the column headers as well as the summary rows. Also notice other aesthetic considerations such as the color of the fonts, and background color of each of the cells. We can use ODS to control every single attribute of the report.
Task 10. Enhance the report with a few ODS features.

With ODS comes some new features in the syntax, including the **STYLE(area)**= option, where area = some part of the report. The areas that will be effected in this task are the columns, summary rows, and headers. We are going to put the **STYLE(area)**= option to work on the **PROC** statement first. Notice the code below only contains the **PROC** statement. Also notice which attributes are going to be effected first.

```
ods rtf file='c:\sugi30.rtf';
proc report data=prep2(where=(age lt 15)) nowindows
   style(column) = (font_face='Arial')
   style(summary) = (font=('Arial', Helvetica, Helv') font_size=12.25pt)
   style(header) = (font_face='Arial' font_size=13.70pt);
```

**PROC statement for Task 10.**

The rest of **PROC REPORT** is the same as it was for the previous task except for the **BREAK** and the **RBREAK** statements. Notice the style= options and the attributes that they are controlling in the rest of the **PROC REPORT** step. Also notice that because they are on a **BREAK** or **RBREAK** statement, they do not have the area in parenthesis.

```
/*----------------------------------------*/
break after f / summarize style=[font_weight=bold font_size=12.50pt
   background=cyan font_face='Arial'];
   compute after f;
      name='Female Avg';
      weight.mean = wholdf/wf;
   endcompute;
break after n / summarize style=[font_weight=bold font_size=12.50pt
   background=light green font_face='Arial'];
   compute after m;
      name='Male Avg';
      weight.mean=wholdn/mw;
   endcompute;
break after goal / summarize style=[font_weight=bold font_size=12.50pt
   background=pink font_face='Arial'];
   compute after goal;
      name='Goal';
      weight.mean=goal;
   endcompute;
rbreak after / summarize style=[font_weight=bold font_size=13.50pt
   background=yellow font_face='Arial'];
   compute after;
      name='Overall Avg';
      weight=weight.mean;
   endcompute;
run;
ods rtf close;
```

**Last part of PROC REPORT step.**

The effects of these style options can be seen in the output below. Notice that each of the summary rows in the report have different colors and font sizes. The higher the level of detail, the smaller the font. This output is intended to show some of the things you can do with the style options of ODS and the **REPORT** procedure. Hopefully, this output will give you some ideas of the possibilities that are available with the ODS. The sky is truly the limit when it comes to generating attractive reports with the SAS system.
Output 10.

<table>
<thead>
<tr>
<th>Sex</th>
<th>Name</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>Alfred</td>
<td>112.5</td>
</tr>
<tr>
<td>F</td>
<td>Alice</td>
<td>84.0</td>
</tr>
<tr>
<td>F</td>
<td>Barbara</td>
<td>98.0</td>
</tr>
<tr>
<td>F</td>
<td>Carol</td>
<td>102.5</td>
</tr>
<tr>
<td>M</td>
<td>Henry</td>
<td>102.5</td>
</tr>
<tr>
<td>M</td>
<td>James</td>
<td>83.0</td>
</tr>
<tr>
<td>F</td>
<td>Jane</td>
<td>84.5</td>
</tr>
<tr>
<td>M</td>
<td>Jeffrey</td>
<td>84.0</td>
</tr>
<tr>
<td>M</td>
<td>John</td>
<td>99.5</td>
</tr>
<tr>
<td>F</td>
<td>Joyce</td>
<td>50.5</td>
</tr>
<tr>
<td>F</td>
<td>Judy</td>
<td>90.0</td>
</tr>
<tr>
<td>F</td>
<td>Louise</td>
<td>77.0</td>
</tr>
<tr>
<td>M</td>
<td>Robert</td>
<td>128.0</td>
</tr>
<tr>
<td>M</td>
<td>Thomas</td>
<td>85.0</td>
</tr>
<tr>
<td></td>
<td>Goal</td>
<td>99.0</td>
</tr>
<tr>
<td></td>
<td>Female Avg</td>
<td>83.8</td>
</tr>
<tr>
<td></td>
<td>Male Avg</td>
<td>99.2</td>
</tr>
<tr>
<td></td>
<td>Overall Avg</td>
<td>91.5</td>
</tr>
</tbody>
</table>

ACKNOWLEDGMENTS
I would like to acknowledge and greatly thank the Technical Support Department at SAS Institute for their helpful knowledge and expertise that they so freely gave.

CONTACT INFORMATION
Your comments and questions are valued and encouraged. Contact the author at:
    Ben Cochran
    The Bedford Group
    Raleigh, NC 27607
    Office: 919.831.1191
    Fax: 919.831.1191
    Email: bedfordgroup@nc.rr.com
    Web: www.bedford-group.com

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.