INTRODUCTION

Prior to version 6.07 and the release of the REPORT procedure in a production mode, there was no true report generator within the SAS system. This odd situation lead to 1) abuse of DATA step reports, 2) a diversion of the purpose of the TABULATE procedure and 3) isolated efforts to circumvent this limitation with procedures such as QPRINT.

We think that the REPORT procedure will become the cornerstone of report writing tools in the near future. The TABULATE procedure will then get back to its origins: an extremely powerful table production language. DATA step reports will end up as the solution when page-setting constraints are numerous.

We should say instead that REPORT will become the cornerstone of report writing tools despite its poor interface and its rather surprising language. This tutorial explains tips and techniques for advanced report-writing applications. Emphasis is on real-life problems and publication-ready reports.

STATISTICS IN COLUMNS

The easiest way to present multiple statistics on a single variable is to place them in columns. Figure 4 illustrates the computation of 6 statistics on June math results. First, the COLUMNS statement defines 5 aliases for TEST3:

COLUMN METHOD SEX TEST3 TEST3=T3NOBS TEST3=T3MIN TEST3=T3MAX TEST3=T3RANGE TEST3=T3STD;

We then use the DEFINE statement to name the statistic computed in each column:

DEFINE TEST3 / MEAN WIDTH=8 CENTER SPACING=3 FORMAT=4.1 "Mean";

Adding the keyword ANALYSIS in the DEFINE statement is optional, but it makes things more clear. For such reports, a data step where the variable would be duplicated becomes useless. Figure 5 shows the report definition.

COMPUTED COLUMNS

When columns for display in the report are computed from variables in the SAS dataset, they can be created in REPORT rather than in a previous data step. In Figure 6 for example, the last three columns are computed from three variables in the dataset (TEST1, TEST2, TEST3). Figure 7 shows the program creating the report.

Computed columns must be 1) named in COLUMNS, 2) detailed in DEFINE with COMPUTED as their status and 3) described in a COMPUTE block. For example, the COMPUTE block for column TOTAL, the sum of test variables expressed in percentages, looks like:

COMPUTE TOTAL;
TOTAL=SUM(TEST1, TEST2, TEST3)*(100/90);
ENDCOMP;

The ENDCOMP statement is mandatory. Most functions and operators available in the data step can be used in the COMPUTE statement. Every variable or column defined on the left of the current column can be used in computations. To compute a character variable, its length and type must be indicated in COMPUTE:

COMPUTE FAIL / LENGTH=7 CHAR;
IF TOTAL<60 THEN FAIL="" - ";
ELSE FAIL="" * ";
ENDCOMP;

PSEUDO-COLUMNS

If the columns used in computations are not variables from the dataset but columns with an ACROSS status (pseudo-columns), these columns must be named through the _Cn_ method. _Cn_ indicates the position of the column relative to the COLUMNS statement, counting both the PRINT and NOPRINT columns.

Figure 8 illustrates a report where accidents, broken down by type of vehicle and region, are counted for each year. Ratios of change since the previous year are computed. Figure 9 shows the program with compute blocks like:

COMPUTE R8887; /* 1988 vs 1987 */
R8887=(_C4-_C3/_C3)*100;
ENDCOMP;
LABELING BREAK LINES

By default, the break line is labeled using the formatted value of the grouping or order variable. This repetition is unattractive and can be disabled by the SUPPRESS parameter in the BREAK statement. The report break line is not labeled by default.

To control this labeling, the grouping variable must be character. Let's take a simple example, derived from Figure 9, and look at the break for Montreal (Figure 1):

```
COMPUTE AFTER REGION;
  REGION='SUB-TOTAL';
ENDCOMP;
BREAK AFTER REGION / OL SUMMARIZE SKIP;
```

Figure 1 - Break for Montreal

<table>
<thead>
<tr>
<th>ADMINISTRATIVE</th>
<th>TYPE OF</th>
<th>------</th>
</tr>
</thead>
<tbody>
<tr>
<td>Montreal</td>
<td>Buses</td>
<td>124</td>
</tr>
<tr>
<td></td>
<td>Cars</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td>Others</td>
<td>221</td>
</tr>
<tr>
<td></td>
<td>Trucks</td>
<td>172</td>
</tr>
<tr>
<td>SU</td>
<td>Others</td>
<td>562</td>
</tr>
</tbody>
</table>

We got "SU" rather than "SUB-TOTAL" as a label because REGION is a character string 2 bytes-long only. We got "Others" as a label for VEHICLE because its value is blank and falls in the OTHER range.

To label the breaks correctly we need to use formats. First, in the compute block dealing with breaks (COMPUTE AFTER variable) or report break (COMPUTE AFTER), the grouping variable is assigned a dummy value ("YY" and "ZZ" for REGION, for example):

```
COMPUTE AFTER REGION;
  REGION="YY";
ENDCOMP;
COMPUTE AFTER;
  REGION="ZZ";
ENDCOMP;
```

In the VALUE statement of the FORMAT procedure, these dummy values are then given a meaning like "SUB-TOTAL" and "TOTAL" (see Figure 9).

MANY STATISTICS IN A BREAK LINE

Placing a statistic in a break line is easy through the BREAK statement. BREAK can also handle situations where many variables are printed with different statistics for each. But what if more than one statistic is required for the same column? Figure 10 is an example of such a report: variables STUDENT, TEST1 and TEST2 are displayed grouped by SEX but three statistics are to be printed in the break line for each column, namely the mean, the minimum and the maximum.

The program producing this report is shown in Figure 11. First, variables TEST1 and TEST2 need to be duplicated in a data step, one copy for each statistic to compute on each test variable (X1 for mean, Y1 for minimum and Z1 for maximum of TEST1, same for TEST2). Then TEST1 and TEST2 are described as DISPLAY variables in the DEFINE statement.

Columns X1, Y1 and Z1 are described as ANALYSIS variables, each with its own statistic. The DEFINE statement for Xs, Ys and Zs also mentions not to print them. Hiding columns is very useful in complex reports because the hidden columns are still available for computations. In the compute block executed after each sex group, the LINE statement is used to print the statistics: X1.MEAN, Y1.MIN, Z1.MAX and so on.

Take note that LINE statement behaves differently from the data step PUT statement despite similar purposes. Each LINE is centered unless the NOCENTER option is in effect or the "@" sign is followed by a column. LINE cannot be executed conditionally although no error message will signal it. Finally, no pointer is available to move back on the line or to skip lines.

The program in Figure 11 also illustrates how graphic characters supported by a printer can be used in reports. Take note that printers have their own way of handling these characters; such programs are hardware dependent.

LABELING NARROW COLUMNS

Various methods can be used to label a narrow column, like CLASS in Figure 12, while using the smallest width possible to hold both the data and label (Figure 2):
While rather simple to implement, the last one is probably the most interesting. First, you stick together the columns using SPACING=0, then you define the first column to be one or two characters longer than the longest formatted value. Finally you split the "common" label between the two columns (Figure 13 shows the code used).

FORMATS VARYING OVER THE LINES

Formatting the first line in a break group differently from other lines in the same break group is a request easily handled in data step reports. Any procedure-based report will accommodate such a request with much more difficulty. As an example, let's take a report with a dollar sign on the right of each value of the first line of a break group. This report is illustrated in Figure 12. Variables in the dataset include SCHOOL (school name), CLASS, BUDGET and EXPENSES.

First, we define two PICTURE statements, with and without currency:

PICTURE DF (ROUND) LOW-HIGH='000,009 $'
PICTURE AF (ROUND) LOW-HIGH='000,009 '

The DEFINE statements specify attributes, including AF as a format without currency for every line of both BUDGET and EXPENSES variables. The width of both fields is fixed to 8 since no values exceed 25,000 $; the left side of the picture is automatically truncated to fit in the width of 8. Both fields are also left-justified: if they were right-justified, results would be useless (Figure 3):

DEFINE BUDGET / DISPLAY WIDTH=8 LEFT FORMAT=AF. SPACING=S
DEFINE EXPENSES / DISPLAY WIDTH=8 LEFT FORMAT=DF.

Figure 3 - Misaligned values in columns.

<table>
<thead>
<tr>
<th>School / class</th>
<th>BUDGET</th>
<th>EXPENSES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Des-Bois</td>
<td>18,550 $</td>
<td>18,733 $</td>
</tr>
<tr>
<td>A1</td>
<td>14,400</td>
<td>16,980</td>
</tr>
<tr>
<td>A2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The CALL DEFINE statement used to reformat the first line of a sub-group must appear in a COMPUTE block, addressing the rightmost column of the report. Since EXPENSES is the last one, the CALL DEFINE will appear in a COMPUTE block addressing.

But the CALL DEFINE must be conditional: only the first observation in a sub-group needs attention. To format the first line of a sub-group differently, it is important to know whether or not the current line is starts a new sub-group. Currently, there is no feature like "FIRST." variable in REPORT. But we can use a variable not mentioned in the COLUMNS statement, whose value will be automatically retained from one observation to the other.

To format the first line of a sub-group differently, let's take a variable, named CHANGED for example, holding a value of 1 at the beginning of a break, and defined in a COMPUTE block before the beginning of the break:

BREAK BEFORE SCHOOL / SUPPRESS;
COMPUTE BEFORE SCHOOL;
CHANGED=1;
ENDCOMP;

Then let's place this in a COMPUTE block addressing the EXPENSES variable, as explained earlier:

COMPUTE EXPENSES;
IF CHANGED=1 THEN DO;
format-modification-for-BUDGET;
format-modification-for-EXPENSES;
CHANGED=0;
END;
ENDCOMP;

Since CHANGED is set at 0 in the loop, no other observation in that sub-group will be reformatted.

The CALL DEFINE obeys the following syntax:

CALL DEFINE(column, "attribute", "value")

where column is either the column number, an expression or the column name between quotes; attributes is a value or the column name between quotes; attributes is a value like FORMAT, COLOR, HIGHLIGHT and so on; value depends on the attribute.

To change the format of the first and only the first line in a sub-group, we code the following (see also Figure 13):

COMPUTE EXPENSES;
IF CHANGED=1 THEN DO;
CALL DEFINE("BUDGET","FORMAT", "DF.");
CALL DEFINE("EXPENSES","FORMAT", "DF.");
CHANGED=0;
END;
ENDCOMP;

COMPUTING PERCENTAGES

The main difficulty when computing percentages is in defining denominators. This is not only true of REPORT but also of the TABULATE procedure. The method shown here can be applied within most contexts where percentages must be computed through the REPORT procedure.
Let's take a look at Figure 14: two percentages have to be computed for each row and each year. The first one, called "REG. %", uses the total number of accidents within a region and a year, aggregating types of vehicles, as the denominator (for Montreal in 1987, this means 562 accidents). The second one, called "TOTAL %", uses the total number of accidents in a year as the denominator (for 1987, this means 2,687 accidents). The two percentages for car accidents in the Montreal region for 1987 are then 8.0 %, or (45/562)\times 100, and 1.7 %, or (45/2687)\times 100.

To display the number of accidents per year, region and type of vehicle, the YEAR variable has to be used twice: first as an ACROSS variable and then as an ANALYSIS variable with the statistic N:

 COLUMN REGION VEHICLE YEAR,  
 (YEAR=NUMBER PCT_REG PCT_TOT);
 DEFINE REGION / GROUP other-param.;
 DEFINE VEHICLE / GROUP other-param.;
 DEFINE YEAR / ACROSS other-param.;
 DEFINE NUMBER / N other-param.;

To compute percentages, we can use variables not mentioned in COLUMNS that will retain their value across observations. These variables serve as denominators. The numerator is the value of the NUMBER column, for each year, using \_Cn_ notation.

Denominators for the first percentage is the total number of accidents in a region and a year. As this information must be available before displaying the first line of result, we use a COMPUTE BEFORE REGION block to compute the regional sub-total (variables ST1-ST5):

 COMPUTE BEFORE REGION;
 ST1=\_C3_;
 ST2=\_C6_;
 ST3=\_C9_;
 ST4=\_C12_;  
 ST5=\_C15_;  
 ENDCOMP;

The same applies to the total number of accidents in a year (variables GT1-GT5):

 COMPUTE BEFORE;
 GT1=\_C3_;
 GT2=\_C6_;
 GT3=\_C9_;
 GT4=\_C12_;  
 GT5=\_C15_;  
 ENDCOMP;

The second step involves dividing the number of observations by the denominator, multiplying the result by 100 so as to get the percentage. In our example, the regional percentage (PCT_REG) and the total percentage (PCT_TOT) are computed in this way for each year:

 COMPUTE PCT_REG;
 \_C4_=(\_C3_/ST1)*100;
 \_C7_=(\_C6_/ST2)*100;
 \_C10_=(\_C9_/ST3)*100;
 \_C13_=(\_C12_/ST4)*100;
 \_C16_=(\_C15_/ST5)*100;
 ENDCOMP;

 COMPUTE PCT_TOT;
 \_C5_=(\_C3_/GT1)*100;
 \_C8_=(\_C6_/GT2)*100;
 \_C11_=(\_C9_/GT3)*100;
 \_C14_=(\_C12_/GT4)*100;
 \_C17_=(\_C15_/GT5)*100;
 ENDCOMP;

A final step is necessary to remove the percentages amounting to more than 100% in the "REG. %" columns. This is done in the COMPUTE AFTER block where columns with PCT_REG (\_C4_, \_C7_ and so on) are given impossible values (here -999) and are formatted in a PICTURE statement like:

 PICTURE PCTF (ROUND) 
 -999=" " (NOEDIT) 
 OTHER="009, 9" (MULT=10);

Excerpt of the final report is reproduced in Figure 14.

CONCLUSION

The REPORT procedure still retains many of its secrets. A clear understanding of its strengths and limitations is the key to successful report writing. Moreover, we found it easier to code and debug DATA _NULL_ PUT reports; its user-friendliness is comparable to TABULATE. The documentation of the interactive interface is good but the few code examples at the end should be expanded.

The writer can be contacted at

Les Services Conseils Hardy Inc.
3487 Carré de Nevers
Ste-Foy QC, Canada
G1X 2C9

SAS is a registered trademark of SAS Institute Inc., Cary NC.
### Figure 4: Report with multiple statistics in columns.

<table>
<thead>
<tr>
<th>Teaching method</th>
<th>Sex</th>
<th>Mean</th>
<th>N</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Range</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>New</td>
<td>Female</td>
<td>24.8</td>
<td>8</td>
<td>16</td>
<td>29</td>
<td>13</td>
<td>4.1</td>
</tr>
<tr>
<td></td>
<td>Male</td>
<td>24.7</td>
<td>6</td>
<td>20</td>
<td>30</td>
<td>10</td>
<td>3.4</td>
</tr>
<tr>
<td>Regular</td>
<td>Female</td>
<td>19.7</td>
<td>7</td>
<td>15</td>
<td>25</td>
<td>10</td>
<td>4.3</td>
</tr>
<tr>
<td></td>
<td>Male</td>
<td>18.7</td>
<td>7</td>
<td>14</td>
<td>22</td>
<td>8</td>
<td>3.0</td>
</tr>
</tbody>
</table>

### Figure 5: Program for the report shown in Figure 4.

```
PROC FORMAT;
  VALUE METHOF 1="Regular" 2="New";
  VALUE $SEXF "H"="Male" "F"="Female";
RUN;
PROC REPORT DATA=BIBSAS.RESULTS LS=60 PS=30 NOWINDOWS SPLIT="*" HEADSKIP;
  COLUMNS METHOD SEX ("Statistics" --" TEST3 TEST3=T3NOBS TEST3=T3MIN
                      TEST3=T3MAX TEST3=T3RANGE TEST3=T3STD);
  DEFINE METHOD / GROUP WIDTH=8 LEFT FORMAT=METHOF. "Teaching*method";
  DEFINE SEX / GROUP WIDTH=7 LEFT FORMAT=$SEXF. "Sex";
  DEFINE TEST3 / MEAN WIDTH=8 CENTER SPACING=3 FORMAT=4.1 "Mean";
  DEFINE T3NOBS / N WIDTH=B CENTER FORMAT=2. "N";
  DEFINE T3MIN / MIN WIDTH=8 CENTER FORMAT=3. "Minimum";
  DEFINE T3MAX / MAX WIDTH=8 CENTER FORMAT=3. "Maximum";
  DEFINE T3RANGE / RANGE WIDTH=8 CENTER FORMAT=3. "Range";
  DEFINE T3STD / STD WIDTH=9 CENTER FORMAT=4.1 "Standard-*deviation";
  BREAK AFTER METHOD / OL SKIP SUMMARIZE SUPPRESS;
  RBREAK AFTER / DOL SKIP SUMMARIZE;
RUN;
```

### Figure 6: Excerpt from a report with columns created through computations.

<table>
<thead>
<tr>
<th>Teaching method</th>
<th>Student number</th>
<th>September</th>
<th>December</th>
<th>June</th>
<th>Total</th>
<th>Mean</th>
<th>Failure</th>
</tr>
</thead>
<tbody>
<tr>
<td>New</td>
<td>43</td>
<td>16.0</td>
<td>18.0</td>
<td>23.0</td>
<td>63 %</td>
<td>19.0</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>44</td>
<td>14.0</td>
<td>15.0</td>
<td>20.0</td>
<td>54 %</td>
<td>16.3</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>52</td>
<td>11.0</td>
<td>12.0</td>
<td>16.0</td>
<td>43 %</td>
<td>13.0</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>57</td>
<td>16.0</td>
<td>18.0</td>
<td>24.0</td>
<td>64 %</td>
<td>19.3</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>67</td>
<td>18.0</td>
<td>19.0</td>
<td>26.0</td>
<td>70 %</td>
<td>21.0</td>
<td>-</td>
</tr>
</tbody>
</table>

---

1464
Figure 7: Program for the report shown in Figure 6.

```plaintext
PROC FORMAT;
  PICTURE PCTF (ROUND) LOW-HIGH="009 %";
RUN;
PROC REPORT DATA=BIBSAS.RESULTS LS=80 PS=45 NOWINDOWS SPLIT="*" HEADSKIP=
  ("Monthly results" "--" TEST1 TEST2 TEST3)
  ("Final results" "--" TOTAL TESTMEAN FAIL);
DEFINE METHOD / ORDER WIDTH=8 LEFT FORMAT=METHOF. "Teaching*method";
DEFINE STUDENT / ORDER WIDTH=7 CENTER FORMAT=2. "Student*number";
DEFINE TEST1 / DISPLAY WIDTH=9 CENTER FORMAT=4.1 "September";
DEFINE TEST2 / DISPLAY WIDTH=9 CENTER FORMAT=4.1 "December";
DEFINE TEST3 / DISPLAY WIDTH=9 CENTER FORMAT=4.1 "June";
DEFINE TOTAL / COMPUTED WIDTH=7 CENTER FORMAT=PCTF. "Total";
DEFINE TESTMEAN / COMPUTED WIDTH=7 CENTER FORMAT=4.1 "Mean";
DEFINE FAIL / COMPUTED WIDTH=7 CENTER FORMAT=$CHAR7. "Failure"
  COMPUTE TOTAL;
    TOTAL=SUM(TEST1,TEST2,TEST3)*(100/90);
ENDCOMP;
  COMPUTE TESTMEAN;
    TESTMEAN=MEAN(TEST1,TEST2,TEST3);
ENDCOMP;
  COMPUTE FAIL / LENGTH=7 CHAR;
    IF TOTAL<60 THEN FAIL="*" ;
    ELSE FAIL="*" ;
ENDCOMP;
BREAK AFTER METHOD / SKIP SUPPRESS;
RUN;
```

Figure 8: Excerpt from a report with computations on ACROSS columns.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Montreal</td>
<td>Buses</td>
<td>124</td>
<td>66</td>
<td>29</td>
<td>78</td>
<td>182</td>
<td>-46%</td>
<td>-56%</td>
<td>168%</td>
<td>133%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cars</td>
<td>45</td>
<td>26</td>
<td>75</td>
<td>97</td>
<td>12</td>
<td>-42%</td>
<td>198%</td>
<td>29%</td>
<td>-87%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Others</td>
<td>221</td>
<td>264</td>
<td>271</td>
<td>160</td>
<td>89</td>
<td>13%</td>
<td>2%</td>
<td>-40%</td>
<td>-44%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Trucks</td>
<td>172</td>
<td>154</td>
<td>90</td>
<td>20</td>
<td>46</td>
<td>-10%</td>
<td>-41%</td>
<td>-7%</td>
<td>130%</td>
<td></td>
</tr>
<tr>
<td>SUB-TOTAL</td>
<td></td>
<td>562</td>
<td>510</td>
<td>465</td>
<td>355</td>
<td>329</td>
<td>-9%</td>
<td>-8%</td>
<td>-23%</td>
<td>-7%</td>
<td></td>
</tr>
<tr>
<td>Quebec</td>
<td>Buses</td>
<td>39</td>
<td>44</td>
<td>104</td>
<td>92</td>
<td>12</td>
<td>12%</td>
<td>136%</td>
<td>35%</td>
<td>-34%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cars</td>
<td>184</td>
<td>50</td>
<td>139</td>
<td>104</td>
<td>100</td>
<td>-72%</td>
<td>298%</td>
<td>-47%</td>
<td>-3%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Others</td>
<td>264</td>
<td>151</td>
<td>370</td>
<td>172</td>
<td>287</td>
<td>-42%</td>
<td>145%</td>
<td>-53%</td>
<td>66%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Trucks</td>
<td>11</td>
<td>46</td>
<td>104</td>
<td>157</td>
<td>182</td>
<td>318%</td>
<td>126%</td>
<td>50%</td>
<td>15%</td>
<td></td>
</tr>
<tr>
<td>SUB-TOTAL</td>
<td></td>
<td>498</td>
<td>291</td>
<td>777</td>
<td>574</td>
<td>661</td>
<td>-41%</td>
<td>167%</td>
<td>-26%</td>
<td>15%</td>
<td></td>
</tr>
<tr>
<td>All others</td>
<td>Buses</td>
<td>346</td>
<td>224</td>
<td>143</td>
<td>321</td>
<td>404</td>
<td>-35%</td>
<td>-36%</td>
<td>124%</td>
<td>25%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cars</td>
<td>356</td>
<td>479</td>
<td>322</td>
<td>281</td>
<td>291</td>
<td>-34%</td>
<td>-32%</td>
<td>-12%</td>
<td>3%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Others</td>
<td>466</td>
<td>642</td>
<td>419</td>
<td>642</td>
<td>427</td>
<td>37%</td>
<td>-34%</td>
<td>53%</td>
<td>-33%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Trucks</td>
<td>459</td>
<td>280</td>
<td>303</td>
<td>326</td>
<td>180</td>
<td>-30%</td>
<td>8%</td>
<td>7%</td>
<td>-44%</td>
<td></td>
</tr>
<tr>
<td>SUB-TOTAL</td>
<td></td>
<td>1627</td>
<td>1625</td>
<td>1187</td>
<td>1570</td>
<td>1302</td>
<td>-0%</td>
<td>-26%</td>
<td>32%</td>
<td>-17%</td>
<td></td>
</tr>
<tr>
<td>TOTAL</td>
<td></td>
<td>2687</td>
<td>2426</td>
<td>2429</td>
<td>2499</td>
<td>2292</td>
<td>-9%</td>
<td>0%</td>
<td>2%</td>
<td>-8%</td>
<td></td>
</tr>
</tbody>
</table>

NUMBER OF ACCIDENTS ('87 - '91) AND ONE-YEAR TRENDS

1465
Figure 9: Program for the report shown in Figure 8.

```plaintext
PROC FORMAT;
  VALUE $REGF ·Ol"="Montreal· ·02"="Quebec·
       ·03", ·04", ·05", ·06"="All others" ·YY"="SUB-TOTAL" ·ZZ"="TOTAL";
  VALUE $VEHF "A"="Cars· uB"="Buses·
       ·C"="Trucks" ·Z"=" OTHER="Others";
  PICTURE PCTF (ROUND) LOW<0=" 009%" (PREFIX="-")
       0-HIGH=" 099%";
RUN;
PROC REPORT DATA=SAMPLE NOWINDOWS LS=96 PS=54 HEADSKIP SPLIT="**;
COLUMN REGION VEHICLE YEAR ("- RATIOS -" R8887 R8988 R9089 R9190);
DEFINE REGION / GROUP ORDER=INTERNAL "ADMINISTRATIVE*REGION"
       WIDTH=14 FORMAT=$REGF.;;
DEFINE VEHICLE / GROUP "TYPE OF*VEHICLE" WIDTH=12 FORMAT=$VEHF.;
DEFINE YEAR / ACROSS "ACCIDENTS PER YEAR -" WIDTH=5 FORMAT=5.;
DEFINE R8887 / COMPUTED "88/87" WIDTH=5 FORMAT=PCTF. SPACING=4;
DEFINE R8988 / COMPUTED "89/88" WIDTH=5 FORMAT=PCTF.;
DEFINE R9089 / COMPUTED "90/89" WIDTH=5 FORMAT=PCTF.;
DEFINE R9190 / COMPUTED "91/90" WIDTH=5 FORMAT=PCTF.;

COMPUTE R8887;
  R8887=((_C4-_C3)/_C3_)*100;
ENDCOMP;
COMPUTE R8988;
  R8988=((_C5-_C4)/_C4_)*100;
ENDCOMP;
COMPUTE R9089;
  R9089=((_C6-_C5)/_C5_)*100;
ENDCOMP;
COMPUTE R9190;
  R9190=((_C7-_C6)/_C6_)*100;
ENDCOMP;

COMPUTE AFTER REGION;
  REGION="YY";
  VEHICLE="Z";
ENDCOMP;
BREAK AFTER REGION / OL SUMMARIZE SKIP;

COMPUTE AFTER;
  REGION="ZZ";
  VEHICLE="Z";
ENDCOMP;
BREAK AFTER / DOL SUMMARIZE SKIP;

TITLE1 "NUMBER OF ACCIDENTS ('87 - '91) AND ONE-YEAR TRENDS";
RUN;
```

1466
Figure 10: Excerpt from a report with multiple statistics in a break line.

<table>
<thead>
<tr>
<th>Sex of student</th>
<th>Student number</th>
<th>Mathematics exams September</th>
<th>December</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female</td>
<td>4</td>
<td>13</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>20</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>17</td>
<td>17</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>28</td>
<td>11</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>41</td>
<td>18</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>42</td>
<td>16</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>52</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>67</td>
<td>18</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>68</td>
<td>17</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>18</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>81</td>
<td>18</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>82</td>
<td>21</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>86</td>
<td>22</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>89</td>
<td>20</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>97</td>
<td>20</td>
<td>24</td>
</tr>
</tbody>
</table>

Average: 17.3  17.7
Minimum: 11   10
Maximum: 22   24

Figure 11: Program for the report shown in Figure 10.

* Copy each TEST variable in the X (mean), Y (min) and Z (max) variables:

```
DATA TEMP;
  SET BIBSAS.RESULTS;
  X1=TEST1; Y1=TEST1; Z1=TEST1;
  X2=TEST2; Y2=TEST2; Z2=TEST2;
RUN;
PROC REPORT DATA=TEMP NOWINDOWS LS=78 PS=50 HEADLINE SPLIT="*";
  COLUMNS SEX STUDENT ("Mathematics exams'" "TEST1 TEST2") X1 X2 Y1 Y2 Z1 Z2;
  DEFINE SEX / .ORDER LEFT WIDTH=7 FORMAT=$SEXF. "Sex of" student;*
  DEFINE STUDENT / ORDER CENTER WIDTH=7 FORMAT=2. "Student number";
  DEFINE TEST1 / DISPLAY CENTER WIDTH=9 "September" FORMAT=2.;
  DEFINE TEST2 / DISPLAY CENTER WIDTH=9 "December" FORMAT=2.;
  DEFINE X1 / ANALYSIS MEAN NOPRINT;
  DEFINE X2 / ANALYSIS MEAN NOPRINT;
  DEFINE Y1 / ANALYSIS MIN NOPRINT;
  DEFINE Y2 / ANALYSIS MIN NOPRINT;
  DEFINE Z1 / ANALYSIS MAX NOPRINT;
  DEFINE Z2 / ANALYSIS MAX NOPRINT;
  BREAK AFTER SEX / SUPPRESS;
  COMPUTE AFTER SEX;
  LINE "* 36** --- "*;
  LINE " Average" 12** = X1.MEAN 4.1 7** = X2.MEAN 4.1 ** ;
  LINE " Minimum" 12** = Y1.MIN 2. 9** = Y2.MIN 2.  ;
  LINE " Maximum" 12** = Z1.MAX 2. 9** = Z2.MAX 2.  ;
  ENDPROC;
RUN;
```
Figure 12: Excerpts from a report with format varying over the lines.

<table>
<thead>
<tr>
<th>School / class</th>
<th>BUDGET</th>
<th>EXPENSES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Des-Bois</td>
<td>A1 18,550 $</td>
<td>18,733 $</td>
</tr>
<tr>
<td></td>
<td>A2 14,400</td>
<td>16,980</td>
</tr>
<tr>
<td></td>
<td>A3 21,350</td>
<td>20,989</td>
</tr>
<tr>
<td></td>
<td>A5 20,100</td>
<td>20,091</td>
</tr>
<tr>
<td></td>
<td>A6 17,900</td>
<td>18,026</td>
</tr>
<tr>
<td></td>
<td>A7 19,750</td>
<td>19,844</td>
</tr>
<tr>
<td>La_Source</td>
<td>01 20,450 $</td>
<td>23,338 $</td>
</tr>
<tr>
<td></td>
<td>02 21,450</td>
<td>22,675</td>
</tr>
<tr>
<td></td>
<td>03 20,750</td>
<td>19,992</td>
</tr>
<tr>
<td></td>
<td>04 22,600</td>
<td>22,458</td>
</tr>
<tr>
<td>Saint-Denis</td>
<td>01 18,150 $</td>
<td>17,330 $</td>
</tr>
<tr>
<td></td>
<td>02 16,850</td>
<td>17,022</td>
</tr>
<tr>
<td></td>
<td>04 17,150</td>
<td>19,456</td>
</tr>
<tr>
<td></td>
<td>05 17,500</td>
<td>18,635</td>
</tr>
<tr>
<td></td>
<td>07 19,800</td>
<td>19,679</td>
</tr>
</tbody>
</table>

Figure 13: Program for the report shown in Figure 12.

```
PROC FORMAT;
  PICTURE DF (ROUND) LOW-HIGH="000,009 $";
  PICTURE AF (ROUND) LOW-HIGH="000,009 ";
RUN;

PROC REPORT DATA=BIBSAS.EXPENSES NOWINDOWS LS=78 PS=30 SPLIT="**" HEADSKIP;
  COLUMNS SCHOOL CLASS BUDGET EXPENSES;
  DEFINE SCHOOL / ORDER WIDTH=12 FORMAT=$11. LEFT "School / cla";
  DEFINE CLASS / ORDER WIDTH=2 SPACING=0 FORMAT=$2. LEFT "ss";
  DEFINE BUDGET / DISPLAY WIDTH=8 FORMAT=AF. LEFT "BUDGET";
  DEFINE EXPENSES / DISPLAY WIDTH=8 FORMAT=AF. LEFT "EXPENSES";
  BREAK BEFORE SCHOOL / SUPPRESS;
  BREAK AFTER SCHOOL / SUPPRESS SKIP;
  COMPUTE BEFORE SCHOOL;
    CHANGED=1;
  ENDCOMP;
  COMPUTE EXPENSES;
    IF CHANGED=1 THEN DO;
      CALL DEFINE("BUDGET","FORMAT","DF.");
      CALL DEFINE("EXPENSES","FORMAT","DF.");
      CHANGED=0;
    END;
  ENDCOMP;
RUN;
```
### Regional and Total Percentages of Accidents

<table>
<thead>
<tr>
<th>Administrative Region</th>
<th>Type of Vehicle</th>
<th>1987</th>
<th>1988</th>
<th>1989</th>
<th>1990</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number %</td>
<td>Number %</td>
<td>Number %</td>
<td>Number %</td>
<td>Number %</td>
</tr>
<tr>
<td>Montreal</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cars</td>
<td>45  8.0</td>
<td>26  5.1</td>
<td>75 16.1</td>
<td>97 27.3</td>
<td></td>
</tr>
<tr>
<td>Buses</td>
<td>174 22.1</td>
<td>66 12.9</td>
<td>29 6.2</td>
<td>78 22.0</td>
<td></td>
</tr>
<tr>
<td>Trucks</td>
<td>172 20.6</td>
<td>30.2 6.3</td>
<td>90 19.4</td>
<td>20 5.6</td>
<td></td>
</tr>
<tr>
<td>Others</td>
<td>221 39.3</td>
<td>26.4 10.9</td>
<td>271 58.3</td>
<td>180 45.1</td>
<td></td>
</tr>
<tr>
<td><strong>Sub-Total</strong></td>
<td>562 100.0</td>
<td>20.9 21.0</td>
<td>465 100.0</td>
<td>19.1</td>
<td>355 100.0</td>
</tr>
<tr>
<td>Quebec</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cars</td>
<td>184 36.9</td>
<td>50 17.2</td>
<td>199 25.6</td>
<td>104 18.1</td>
<td></td>
</tr>
<tr>
<td>Buses</td>
<td>39 7.8</td>
<td>44 15.1</td>
<td>104 13.4</td>
<td>144 26.4</td>
<td></td>
</tr>
<tr>
<td>Trucks</td>
<td>11 2.2</td>
<td>15.6 1.9</td>
<td>104 13.4</td>
<td>157 27.4</td>
<td></td>
</tr>
<tr>
<td>Others</td>
<td>244 53.0</td>
<td>121 6.2</td>
<td>172 30.0</td>
<td>172 30.0</td>
<td></td>
</tr>
<tr>
<td><strong>Sub-Total</strong></td>
<td>498 100.0</td>
<td>18.5 12.0</td>
<td>777 100.0</td>
<td>32.0</td>
<td>574 100.0</td>
</tr>
<tr>
<td>All others</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cars</td>
<td>356 21.9</td>
<td>479 29.5</td>
<td>322 27.1</td>
<td>281 17.9</td>
<td></td>
</tr>
<tr>
<td>Buses</td>
<td>346 21.5</td>
<td>224 13.8</td>
<td>143 12.0</td>
<td>321 20.4</td>
<td></td>
</tr>
<tr>
<td>Trucks</td>
<td>459 28.2</td>
<td>280 17.2</td>
<td>303 25.5</td>
<td>326 20.8</td>
<td></td>
</tr>
<tr>
<td>Others</td>
<td>446 28.6</td>
<td>121.9 6.2</td>
<td>370 47.6</td>
<td>172 30.0</td>
<td></td>
</tr>
<tr>
<td><strong>Sub-Total</strong></td>
<td>1627 100.0</td>
<td>60.6 67.0</td>
<td>1187 100.0</td>
<td>48.9</td>
<td>1570 100.0</td>
</tr>
<tr>
<td>Total</td>
<td>2687 100.0</td>
<td>100.0 100.0</td>
<td>1187 100.0</td>
<td>100.0</td>
<td>1570 100.0</td>
</tr>
</tbody>
</table>

**Figure 15:** Program for the report shown in Figure 14.

```plaintext
PROC REPORT DATA=sample nowindows ls=132 ps=54 headskip split="*";
COLUMN REGION véhicule year, (year=number pct_reg pct_tot);
DEFINE REGION / GROUP ORDER=INTERNAL "administrative*region" width=14 format=strf;
DEFINE véhicule / GROUP ORDER=INTERNAL "type of*vehic le" width=7 format=$vhf;
DEFINE NUMBER / analysis n "number" width=6 format=5;
DEFINE PCT_REG / COMPUTED "reg." "%" width=5 spacing=1 center format=pctf;
DEFINE PCT_TOT / Computes "total." "%" width=5 spacing=1 center format=pctf;
COMPUTE BEFORE;
    GT1=_C3_; GT2=_C6_; GT3=_C9_; GT4=_C12_; GT5=_C15_; 
ENDCOMP;
COMPUTE BEFORE REGION;
    ST1=_C3_; ST2=_C6_; ST3=_C9_; ST4=_C12_; ST5=_C15_; 
ENDCOMP;
COMPUTE PCT_REG;
    _c4_ =(_C3_ /ST1)*100;
    _C7_ =(_C6_ /ST2)*100;
    _C10_ =(_C9_ /ST3)*100;
    _C13_ =(_C12_ /ST4)*100;
    _C16_ =(_C15_ /ST5)*100;
ENDCOMP;
COMPUTE PCT_TOT;
    _c5_ =(_C3_ /GT1)*100;
    _c8_ =(_C6_ /GT2)*100;
    _C11_ =(_C9_ /GT3)*100;
    _C14_ =(_C12_ /GT4)*100;
    _C17_ =(_C15_ /GT5)*100;
ENDCOMP;
COMPUTE AFTER REGION;
    REGION="yy";
    véhicule="z";
ENDCOMP;
BREAK AFTER REGION / OL SUMMARIZE SKIP;
COMPUTE AFTER;
    región="zz";
    véhicule="z";
    _c4_=-999; _C7_=-999; _C10_=-999; _c13_=-999; _C16_=-999;
ENDCOMP;
BREAK AFTER / O/L SUMMARIZE SKIP;
TITLE "regional and total percentages of accidents": 
FOOTNOTE:
RUN;
```