PROC REPORT BATCH LANGUAGE: TIPS AND TECHNIQUES

Jean Hardy
Services Conseils Hardy Inc.

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 endup 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

ADMINISTRATIVE REGION	TYPE OF VEHICLE	------ 1987
Montreal	Buses	124
	Cars	45
	Others	221
	Trucks	172
SU	Others	562

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):

Figure 2 - Labeling a narrow column

SCHOOL	CLASS	BUDGET	EXPENSES
Saint-Denis	01	18,150	17,330
	02	16,850	17,022
	.....	.....	.....
SCHOOL/CLASS		BUDGET	EXPENSES
Saint-Denis	01	18,150	17,330
SCHOOL / CLASS		BUDGET	EXPENSES
Saint-Denis 01		18,150	17,330

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=5
                " BUDGET ";
```

Figure 3 - Misaligned values in columns.

School / class		BUDGET	EXPENSES
Des-Bois	A1	18,550 $	18,733 $
	A2	14,400	16,980
	..	.....	

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 attentiont. To format the first line of a sub-group differently, it is important to know whether or not the current line is starts a now sub-group. Currently, there is no feature like "FIRST." variable in REPORT. But we can use a variable not mentionned 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 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)*100, and 1.7 %, or (45/2687)*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 _C*n*_ 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 than 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 Carre de Nevers
Ste-Foy QC, Canada
G1X 2C9

Figure 4: Report with multiple statistics in columns.

Teaching method	Sex	Statistics					
		Mean	N	Minimum	Maximum	Range	Standard-deviation
New	Female	24.8	8	16	29	13	4.1
	Male	24.7	6	20	30	10	3.4
		24.7	14	16	30	14	3.7
Regular	Female	19.7	7	15	25	10	4.3
	Male	18.7	7	14	22	8	3.0
		19.2	14	14	25	11	3.6
		22.0	28	14	30	16	4.5

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

```
PROC FORMAT;
  VALUE METHOF    1="Regular"   2="New";
  VALUE $SEXF   "H"="Male"    "F"="Female";
RUN;
PROC REPORT  DATA=BIBSAS.RESULTS  LS=80  PS=30  NOWINDOWS  SPLIT="*"  HEADSKIP;
  COLUMNS METHOD SEX ("Statistics" "--"  TEST3  TEST3=T3NOBS  TEST3=T3MIN
          TEST3=T3MAX  TEST3=T3RANGE  TEST3=T3STD);
  DEFINE METHOD    / GROUP  WIDTH=8  LEFT  FORMAT=METHOF.  "Teaching*method";
  DEFINE SEX       / GROUP  WIDTH=7  LEFT  FORMAT=$SEXF.  "Sex";
  DEFINE TEST3     / MEAN   WIDTH=8  CENTER  SPACING=3  FORMAT=4.1  "Mean";
  DEFINE T3NOBS    / N      WIDTH=8  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.

Teaching method	Student number	Mathematics					
		Monthly results			Final results		
		September	December	June	Total	Mean	Failure
New	43	16.0	18.0	23.0	63 %	19.0	
	44	14.0	15.0	20.0	54 %	16.3	-
	52	11.0	12.0	16.0	43 %	13.0	-
	57	16.0	18.0	24.0	64 %	19.3	
	67	18.0	19.0	26.0	70 %	21.0	

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

```
PROC FORMAT;
  PICTURE PCTF (ROUND) LOW-HIGH="009 %";
RUN;
PROC REPORT  DATA=BIBSAS.RESULTS  LS=80  PS=45  NOWINDOWS  SPLIT="*"  HEADSKIP;
  COLUMNS METHOD STUDENT ("Mathematics" "--"
          ("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.

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

ADMINISTRATIVE REGION	TYPE OF VEHICLE	ACCIDENTS PER YEAR 1987	1988	1989	1990	1991	RATIOS 88/87	89/88	90/89	91/90
Montreal	Buses	124	66	29	78	182	-46%	-56%	168%	133%
	Cars	45	26	75	97	12	-42%	188%	29%	-87%
	Others	221	264	271	160	89	19%	2%	-40%	-44%
	Trucks	172	154	90	20	46	-10%	-41%	-77%	130%
SUB-TOTAL		562	510	465	355	329	-9%	-8%	-23%	-7%
Quebec	Buses	39	44	104	141	92	12%	136%	35%	-34%
	Cars	184	50	199	104	100	-72%	298%	-47%	-3%
	Others	264	151	370	172	287	-42%	145%	-53%	66%
	Trucks	11	46	104	157	182	318%	126%	50%	15%
SUB-TOTAL		498	291	777	574	661	-41%	167%	-26%	15%
All others	Buses	346	224	143	321	404	-35%	-36%	124%	25%
	Cars	356	479	322	281	291	34%	-32%	-12%	3%
	Others	466	642	419	642	427	37%	-34%	53%	-33%
	Trucks	459	280	303	326	180	-38%	8%	7%	-44%
SUB-TOTAL		1627	1625	1187	1570	1302	-0%	-26%	32%	-17%
TOTAL		2687	2426	2429	2499	2292	-9%	0%	2%	-8%

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

```
PROC FORMAT;
  VALUE $REGF  "01"="Montreal"   "02"="Quebec"
               "03", "04", "05", "06"="All others"
               "YY"="SUB-TOTAL"  "ZZ"="TOTAL";
  VALUE $VEHF   "A"="Cars"        "B"="Buses"
                "C"="Trucks"      "Z"=" "
              OTHER="Others";
  PICTURE PCTF (ROUND) LOW-<0=" 009%" (PREFIX="-")
                       0-HIGH=" 009%";
RUN;
PROC REPORT DATA=SAMPLE  NOWINDOWS  LS=96  PS=54  HEADSKIP  SPLIT="*";
  COLUMN REGION VEHICLE YEAR ("- RATIOS -"  R8887 R8988 R9089 R9190);
  DEFINE REGION   / GROUP  ORDER=INTERNAL  "ADMINISTRATIVE*REGION"
                    WIDTH=14  FORMAT=$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;
  RBREAK AFTER / DOL SUMMARIZE SKIP;

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

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

Sex of student	Student number	Mathematics exams	
		September	December
Female	4	13	12
	11	20	18
	17	17	16
	28	11	10
	41	18	17
	42	16	17
	52	11	12
	67	18	19
	68	17	16
	70	18	19
	81	18	20
	82	21	23
	86	22	24
	89	20	18
	97	20	24
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.  "  │";
    LINE "└"  36*"─"  "┘";
    LINE;
  ENDCOMP;
RUN;
```

Figure 12: Excerpts from a report with format varying over the lines.

```
School / class      BUDGET        EXPENSES

Des-Bois     A1     18,550 $      18,733 $
             A2     14,400        16,980
             A3     21,350        20,989
             A5     20,100        20,091
             A6     17,900        18,026
             A7     19,750        19,844

La_Source    01     20,450 $      23,338 $
             02     21,450        22,675
             03     20,750        19,992
             04     22,600        22,458

Saint-Denis  01     18,150 $      17,330 $
             02     16,850        17,022
             04     17,150        19,456
             05     17,500        18,635
             07     19,800        19,679
```

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  SPACING=5  " BUDGET ";
  DEFINE EXPENSES / DISPLAY  WIDTH=8  FORMAT=AF.  LEFT  SPACING=5  "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;
```

Figure 14: Excerpt (left-hand side) from a report with percentages computed using various denominators.

REGIONAL AND TOTAL PERCENTAGES OF ACCIDENTS

CALENDAR YEAR

ADMINISTRATIVE REGION	TYPE OF VEHICLE	1987 Number	1987 REG. %	1987 TOTAL %	1988 Number	1988 REG. %	1988 TOTAL %	1989 Number	1989 REG. %	1989 TOTAL %	1990 Number	1990 REG. %	1990 TOT %
Montreal	Cars	45	8,0	1,7	26	5,1	1,1	75	16,1	3,1	97	27,3	3
	Buses	124	22,1	4,6	66	12,9	2,7	29	6,2	1,2	78	22,0	3
	Trucks	172	30,6	6,4	154	30,2	6,3	90	19,4	3,7	20	5,6	0
	Others	221	39,3	8,2	264	51,8	10,9	271	58,3	11,2	160	45,1	6
SUB-TOTAL		562	100,0	20,9	510	100,0	21,0	465	100,0	19,1	355	100,0	14
Quebec	Cars	184	36,9	6,8	50	17,2	2,1	199	25,6	8,2	104	18,1	4
	Buses	39	7,8	1,5	44	15,1	1,8	104	13,4	4,3	141	24,6	5
	Trucks	11	2,2	0,4	46	15,8	1,9	104	13,4	4,3	157	27,4	6
	Others	264	53,0	9,8	151	51,9	6,2	370	47,6	15,2	172	30,0	6
SUB-TOTAL		498	100,0	18,5	291	100,0	12,0	777	100,0	32,0	574	100,0	23
All others	Cars	356	21,9	13,2	479	29,5	19,7	322	27,1	13,3	281	17,9	11
	Buses	346	21,3	12,9	224	13,8	9,2	143	12,0	5,9	321	20,4	12
	Trucks	459	28,2	17,1	280	17,2	11,5	303	25,5	12,5	326	20,8	13
	Others	466	28,6	17,3	642	39,5	26,5	419	35,3	17,2	642	40,9	25
SUB-TOTAL		1627	100,0	60,6	1625	100,0	67,0	1187	100,0	48,9	1570	100,0	62
TOTAL		2687		100,0	2426		100,0	2429		100,0	2499		100

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

```
PROC REPORT DATA=SAMPLE  NOWINDOWS  LS=132  PS=54  HEADSKIP  SPLIT="*";
  COLUMN REGION VEHICLE YEAR,("---" YEAR=NUMBER PCT_REG PCT_TOT);
  DEFINE REGION   / GROUP  ORDER=INTERNAL  "ADMINISTRATIVE*REGION"  WIDTH=14  FORMAT=$REGF.;
  DEFINE VEHICLE  / GROUP  ORDER=INTERNAL  "TYPE OF*VEHICLE"  WIDTH=7  FORMAT=$VEHF.;
  DEFINE YEAR     / ACROSS  "CALENDAR YEAR*---" ";
  DEFINE NUMBER   / ANALYSIS  N  "Number"  WIDTH=6  FORMAT=5.;
  DEFINE PCT_REG  / COMPUTED  "REG.*%"  WIDTH=5  SPACING=1  CENTER  FORMAT=PCTF.;
  DEFINE PCT_TOT  / COMPUTED  "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";
    VEHICLE="Z";
  ENDCOMP;
  BREAK AFTER REGION / OL SUMMARIZE SKIP;

  COMPUTE AFTER;
    REGION="ZZ";
    VEHICLE="Z";
    _C4_ =-999;    _C7_ =-999;    _C10_=-999;    _C13_=-999;    _C16_=-999;
  ENDCOMP;
  RBREAK AFTER / DOL SUMMARIZE SKIP;

  TITLE1 "REGIONAL AND TOTAL PERCENTAGES OF ACCIDENTS";
  FOOTNOTE;
RUN;
```