Report Writing

Table of Contents

Introduction 1

Detail Reports versus Summary Reports 1

SAS Report Writing Tools 2

Introduction to the REPORT Procedure 3

Creating Detail Reports 4

Invoking the REPORT Procedure 4

Shaping the Basic Report Layout 6

Selecting Variables 7
Defining Variable Usage 10
Defining Variable Attributes 13
Defining Statistics 15

Report Definitions 19

Storing Report Definitions 20
Using Stored Report Definitions 22

Enhancing the Appearance of the Detail Report 23

Modifying Column Appearance 24
Setting Report Options 29
Setting System Options 34
Adding Titles and Footnotes 37
Refreshing the Report Display 40

PROC REPORT Language Statements 41

Viewing the PROC REPORT Language Statements 41
Storing the PROC REPORT Language Statements 41

Ordering and Totaling Rows 44

Defining an ORDER Variable 45
Modifying the Default Ordering Sequence 49
Creating Break Lines 50
Paging Through the Report 57

Adding Variables to the Report 59

Adding a Computed Variable 59
Hiding Unnecessary Variables 62

Creating and Applying User-defined Formats 66

Creating User-defined Formats 66
Applying User-defined Formats in PROC REPORT 69

Subsetting the Report **72**
Subsetting the Report Temporarily *73*
Subsetting the Report Permanently *74*
Subsetting the Report Based on a Condition *75*
Altering the Report Structure **82**

Creating Summary Reports **84**

Grouping Rows **84**
Defining a GROUP Variable *87*
Breaking on GROUP Variables *91*
Creating Summarized Output Data **93**
Calculating Percentages **95**
Traffic Lighting **98**
The CALL DEFINE Statement *98*
Customizing Break Lines **102**
The LINE Statement *102*
The $VARYING. Format *102*
The PUT Function *103*
The LENGTH Function *103*

Advanced Report Writing Techniques **109**

Creating Cross-tabular Reports **109**
Defining an ACROSS Variable *109*
Presenting the Same Column in Different Ways **120**
Complex Grouping **123**
Creating Multi-column Reports **130**

Introduction

Detail Reports versus Summary Reports

A report is a tool for communicating information to a reader in a concise manner. Reports enable you to organize and interpret data. With the SAS System, you can produce detail and summary reports using SAS procedures.

Detail reports contain one row for every observation selected for the report. Each row is a detail row representing a single observation. Summary reports consolidate data. Each row represents multiple observations.

Both detail and summary reports can contain summary lines as well as detail rows. A summary line summarizes numerical data for either a set of rows or all detail rows.

SAS Report Writing Tools

Many SAS software products produce reports. Also Base SAS software offers a wide variety of report writing tools. You can use

- PROC TABULATE to display a variety of statistics in a tabular format
- PROC FORMAT to define custom formats
- FILE and PUT statements in a DATA step to create custom reports
- PROC MEANS to calculate summary statistics
- the SAS macro facility to include variable values in titles and footnotes.

The focus of this course is on the REPORT procedure. You can use PROC REPORT to

- select columns and control the order in which they appear in the report
- enhance reports with system options, labels, formats, and titles
- include data set variables, computed variables, and statistics
- order data values and suppress repetitious printing of values
- group data values
- summarize the data within each group
- establish breaks to generate subtotals within the report
- place different groups on separate pages
- use WHERE processing to create reports from subsets of data
- establish breaks to generate a grand total for the report
- create customized break lines
- display values of a variable across the report as column headings
- group columns within values of a variable
- include descriptive statistics
- create multi-column reports
- …

Introduction to the REPORT Procedure

The REPORT procedure is an easy-to-use, ad hoc report generator that simplifies the report development process. It combines features from the PRINT, MEANS, and TABULATE procedures with features of DATA step report writing to provide a powerful report writing tool.

PROC REPORT enables you to

- create custom reports
- develop and store report templates
- view previously defined report templates
- generate reports in windowing and non-windowing environments
- generate multiple reports from one report definition.

Creating Detail Reports

Invoking the REPORT Procedure

You can use PROC REPORT in three ways:

- in a windowing environment with a prompting facility that guides you as you build a report.
- in a windowing environment without the prompting facility.
- in a non-windowing environment. In this case, you submit a series of statements with the PROC REPORT statement, just as you do in other SAS procedures. You can submit these statements from the PROGRAM EDITOR window with the NOWINDOWS option in the PROC REPORT statement.

SYNTAX

```
PROC REPORT DATA = SAS-data-set
            WINDOWS|NOWINDOWS
            PROMPT;
RUN;
```

DATA = *SAS-data-set*	specifies the input SAS data set to be used by the REPORT procedure. If you omit the DATA = option, the most recently created SAS data set is used.
WINDOWS|NOWINDOWS	selects a windowing or non-windowing environment. When you use WINDOWS, SAS opens the REPORT window, which enables you to modify a report repeatedly and to see the modifications immediately. The WINDOWS option is the default. When you use NOWINDOWS, PROC REPORT runs without the REPORT window and sends its output to the SAS procedure output.
PROMPT	opens the REPORT window and starts the PROMPT facility. This facility guides you through creating a new report or adding more data set variables or statistics to an existing report. If you start PROC REPORT with prompting, the first window gives you a chance to limit the number of observations that are used during prompting. When you exit the prompter, PROC REPORT removes the limit.

EXAMPLE

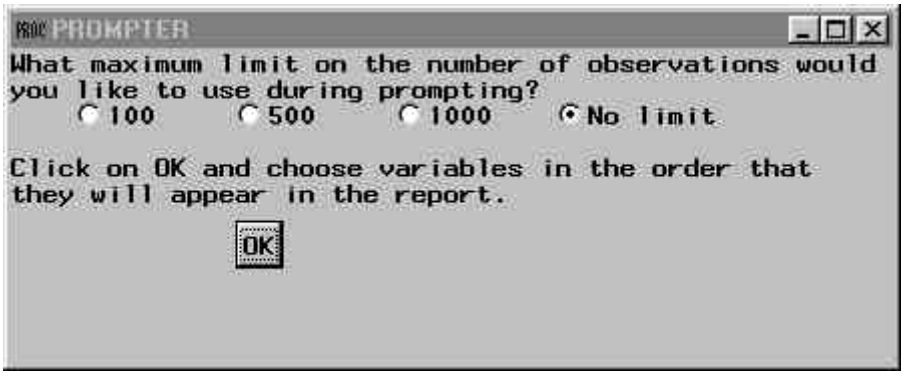
From the PROGRAM EDITOR window, invoke PROC REPORT with prompting to produce a simple detail report of the REPORT.JULY1998 SAS data set.

REPORT.JULY1998 (obs = 10)

S_CTRY	S_CITY	S_TYPE	P_GRP	P_SUB	SALES
B	B-A	R	Beverages	Waters	26136
B	B-A	R	Beverages	Fruit Juices	18159
B	B-A	R	Beverages	Soft Drinks	143016
B	B-A	R	Beverages	Beers	123917
B	B-A	R	Beverages	Wines	10272
B	B-A	R	Beverages	Alcoholic Drinks	46593
B	B-A	R	Butchery	Pork	16530
B	B-A	R	Butchery	Veal	36989
B	B-A	R	Butchery	Lamb	32245
B	B-A	R	Butchery	Chicken	16327

```
proc report data = report.july1998 prompt;
run;
```

PROMPTER Window



If you start PROC REPORT with prompting, the first window gives you a chance to limit the number of observations used during prompting. When you exit the prompter, PROC REPORT removes the limit.

Shaping the Basic Report Layout

Report writing is simplified if you approach it with a clear understanding of what you want the report to look like. The most important thing to determine is the layout of the report.

To determine the layout, ask yourself these kinds of questions:

- Do I have all the necessary data needed for the report?
- What kind of statistics do I need?
- What do I want to display in each column of the report?
- In what order do I want the columns to appear?
- Do I want to display a column for each value of a particular variable?
- Do I want a row for every observation in the report, or do I want to consolidate information for multiple observations into one row?
- In what order do I want the rows to appear?

Selecting Variables

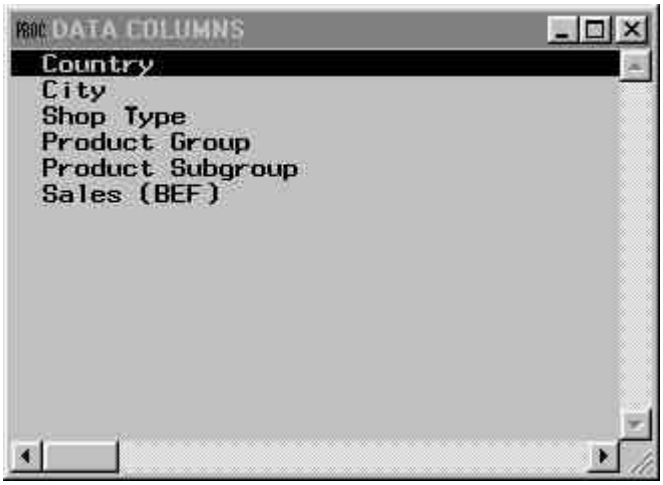
The first step in shaping a report is choosing the columns you want to appear in the report.

By default, the report contains a column for each variable in the input SAS data set and the columns are in the same order as the variables in the SAS data set.

The DATA COLUMNS Window

The DATA COLUMNS window lists all variables in the input SAS data set and enables you to select one or more variables to add to the report. When you select the first variable, it moves to the top of the list in the window and is highlighted. To cancel a selection and remove the highlighting, repeat the process. If you select multiple variables, subsequent selections move to the bottom of the list of selected variables. An asterisk (*) identifies each selected variable. The order of selected variables from top to bottom determines their order in the report from left to right.

DATA COLUMNS Window



The COLUMN Statement

You can use a COLUMN statement to list the items to appear in the columns and describe the order of the columns from left to right.

SYNTAX

```
COLUMN report-item(s);
```

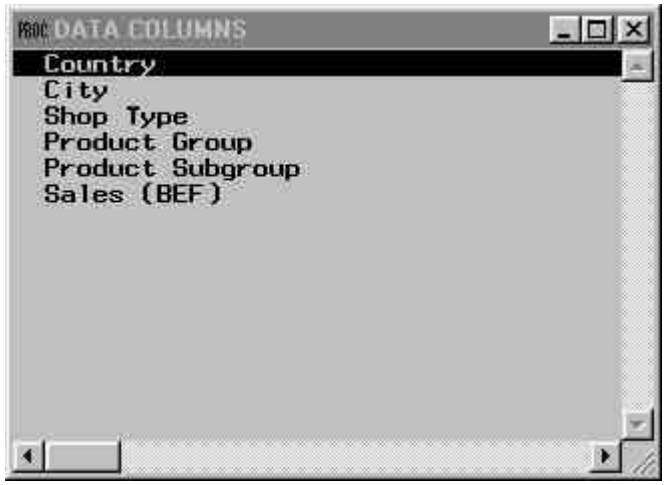
A report-item can be

- a SAS data set variable
- a statistic calculated by the REPORT procedure
- a variable you compute based on the other items in the report (a computed variable).

EXAMPLE

Use the DATA COLUMNS window to select the variables to include in the report. The order in which you select the variables determines their order in your initial report.

DATA COLUMNS Window



Select these variables in this order:

- City
- Product Group
- Product Subgroup
- Sales (BEF)

Select **File - Accept Selection** to close the DATA COLUMNS window. The initial report is displayed in the REPORT window and the DEFINITION window is opened.

Partial PROC REPORT Output

```
                          The SAS System      09:11 Monday, August 7, 2000   1

                                                                      Sales
City    Product Group          Product Subgroup                       (BEF)
B-A     Beverages              Waters                                 26136
B-A     Beverages              Fruit Juices                           18159
B-A     Beverages              Soft Drinks                           143016
B-A     Beverages              Beers                                 123917
B-A     Beverages              Wines                                  10272
B-A     Beverages              Alcoholic Drinks                       46593
B-A     Butchery               Pork                                16529.75
B-A     Butchery               Veal                                36989.15
B-A     Butchery               Lamb                                 32245.3
B-A     Butchery               Chicken                                16327
```

Defining Variable Usage

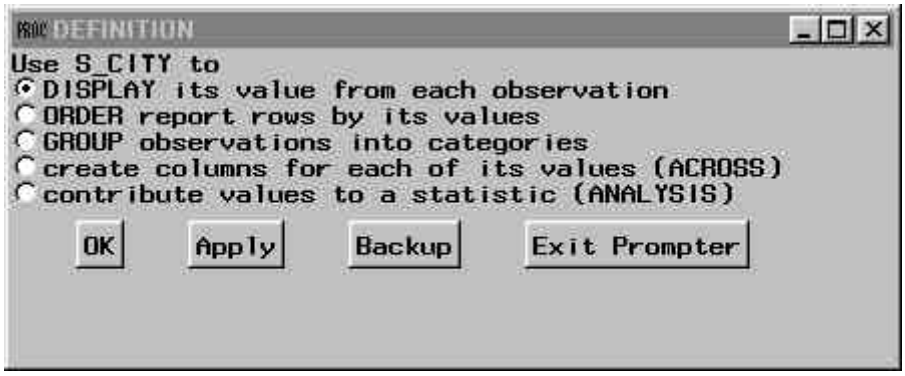
Much of a report's layout is determined by the way you use variables in the report. In addition to DISPLAY variables, you can define ORDER, GROUP, ACROSS and ANALYSIS variables.

- If you want the values of the variable to appear as they do in the input SAS data set, then specify DISPLAY as usage in the DEFINITION window.
- If you want the values of the variable to determine the order of the rows in the report, then specify ORDER as usage in the DEFINITION window.
- If you want the values of the variable to consolidate into one row all observations from the input SAS data set that have the same variable value, then specify GROUP as usage in the DEFINITION window.
- If you want the values of the variable to form column headers, then specify ACROSS as usage in the DEFINITION window.
- If you want the values of the variable to calculate a statistic for all observations that have a unique combination of values for all group variables, then specify ANALYSIS as usage in the DEFINITION window.

A report can also contain variables that are not in the input SAS data set. These variables must have a usage of COMPUTED.

The DEFINITION Window

DEFINITION Window



In prompting mode, the DEFINITION window contains 4 buttons:

- **OK** applies the information in the open window to the report and continues the prompting process.
- **Apply** applies the information in the open window to the report and keeps the window open.
- **Backup** returns you to the previous PROMPTER window.
- **Exit Prompter** closes the PROMPTER window without applying any more changes to the report.

The DEFINE Statement

You can use the DEFINE statement to specify how to use a report item.

SYNTAX

```
DEFINE report-item / usage;
```

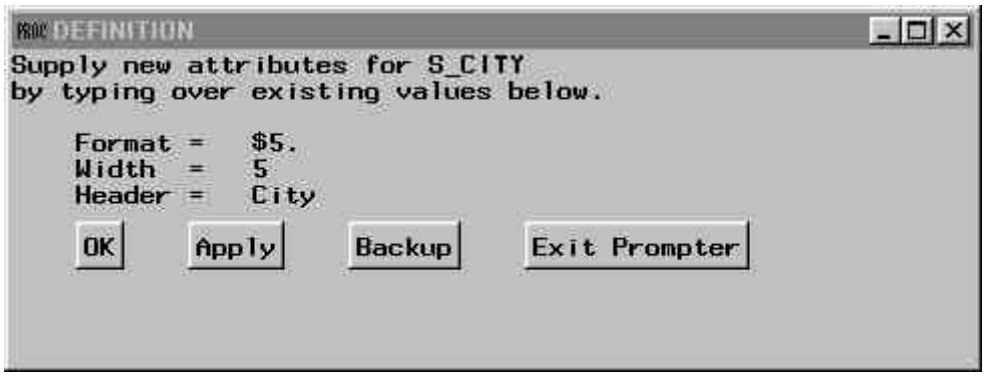
The usage option in the DEFINE statement can be

- DISPLAY
- ORDER
- GROUP
- ACROSS
- COMPUTED
- SUM, MEAN, …

Defining Variable Attributes

The DEFINITION Window

DEFINITION Window



Next, specify attributes for the variable S_CITY. By default, the DEFINITION window displays the attributes of the variables using information from the descriptor portion of the SAS data set. To modify the default format, width or header attributes, type over the existing values.

The DEFINE Statement

You can also use the DEFINE statement to specify attributes for the report item.

SYNTAX

```
DEFINE report-item / usage attribute(s);
```

Selected attributes include:

ATTRIBUTE	MEANING
FORMAT =	assigns a SAS or user-defined format to the report item.
WIDTH =	defines the width of the column in which PROC REPORT displays the report item.
"column-header"	defines the column header for the report item.

Defining Statistics

The DEFINITION Window

The variable SALES is the first numeric variable to define. Numeric variables have a default definition of ANALYSIS.

DEFINITION Window

ANALYSIS variables must have statistics associated with them. The default statistic is SUM. Other statistics include:

KEYWORD	STATISTIC
N	the number of observations in the subgroup having non-missing values for the variable
NMISS	the number of observations in the subgroup having missing values for the variable
MEAN	the arithmetic mean
USS	the uncorrected sum of squares
MIN	the minimum value
MAX	the maximum value
RANGE	the range
STD	the standard deviation
STDERR	the standard error of the mean
VAR	the variance
CV	the coefficient of variation
CSS	the corrected sum of squares

(continued on next page)

KEYWORD	STATISTIC
T	student's *t* for testing the hypothesis that the population mean is 0
PRT	the probability of a greater absolute value for the *t*-value above
SUMWGT	the sum of the WEIGHT variable values

The DEFINE Statement

You can also use the DEFINE statement to associate a statistic with an ANALYSIS variable.

SYNTAX

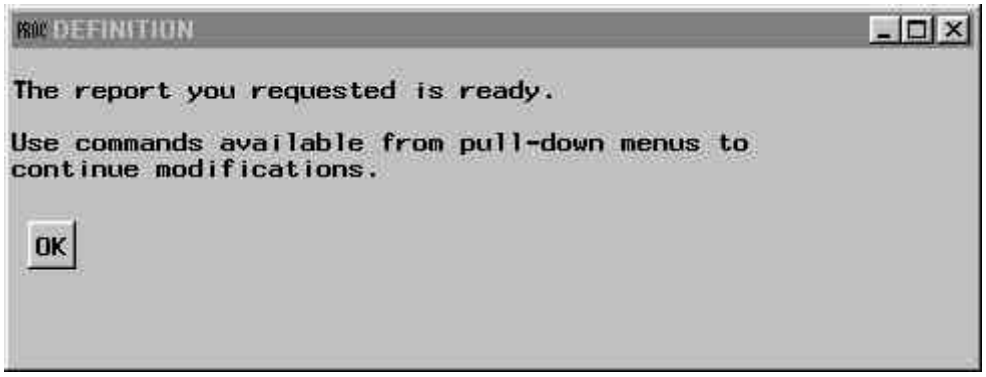
```
DEFINE report-item / usage;
```

When report-item refers to an ANALYSIS variable the usage option in the DEFINE statement can be

- SUM
- MEAN
- …

After you define all of the variables, the DEFINITION window indicates the report is ready.

DEFINITION Window



The initial report is displayed based on the definitions you specified for the variables. You modify and enhance this report later in this course.

EXAMPLE

Make appropriate selections in prompting mode to define the variables S_CITY, P_GRP, P_SUB and SALES as follows:

VARIABLE	DEFINITION	VALUE
S_CITY	**Usage**	DISPLAY
P_GRP	**Usage**	DISPLAY
P_SUB	**Usage**	DISPLAY
SALES	**Usage** **Statistic** **Attributes** Format = Width =	ANALYSIS SUM COMMAX10. 10

Partial PROC REPORT Output

The SAS System 09:11 Monday, August 7, 2000 1

City	Product Group	Product Subgroup	Sales (BEF)
B-A	Beverages	Waters	26.136
B-A	Beverages	Fruit Juices	18.159
B-A	Beverages	Soft Drinks	143.016
B-A	Beverages	Beers	123.917
B-A	Beverages	Wines	10.272
B-A	Beverages	Alcoholic Drinks	46.593
B-A	Butchery	Pork	16.530
B-A	Butchery	Veal	36.989
B-A	Butchery	Lamb	32.245
B-A	Butchery	Chicken	16.327

Report Definitions

After you select and define variables for your report, you can save a template of the report. This template is a report definition.

A stored report definition

- contains a set of instructions used to produce the report
- is an internal representation of the REPORT language
- does not contain data or lines of output
- is stored as a catalog entry with a type of REPT
- can be used with any SAS data set containing variables with the same name and type as those referenced in the original SAS data set
- can be used as a starting point to create similar reports
- can produce reports in a windowing or non-windowing environment
- cannot be viewed or edited directly by the user.

When you store a report definition,

- titles, footnotes, subsetting criteria, and system options are not saved
- report options are saved.

Storing Report Definitions

Select **File - Save Report ...** to store the report definition. The SAVE DEFINITION window appears.

SAVE DEFINITION Window

A report definition is stored in a catalog entry with a four-level name.

SYNTAX

```
libref.catalog.entry-name.REPT
```

libref	is the logical name of the SAS data library to which the catalog belongs.
catalog	is a valid SAS name for the catalog.
entry-name	is a valid SAS name for a catalog entry.
REPT	is assigned automatically by the SAS System when the entry is created.

EXAMPLE

Store the report definition in the following REPT catalog entry:

FIELD	VALUE
LIBNAME	REPORT
CATALOG	REP_DEF
REPORT NAME	DETAIL1
DESCRIPTION	Simple List Report

If the catalog does not already exists, the MESSAGES window opens stating that a new catalog has been created.

MESSAGES Window

A note appears at the bottom of the display confirming that the report definition is stored.

```
NOTE: Definition stored in REPORT.REP_DEF.DETAIL1.
```

Using Stored Report Definitions

To load a stored report definition and generate a report, specify the REPORT = option in the PROC REPORT statement.

SYNTAX

```
PROC REPORT DATA = SAS-data-set
            REPORT = libref.catalog.entry;
RUN;
```

DATA = *SAS-data-set*	specifies the input SAS data set to be used by the REPORT procedure.
REPORT = *libref.catalog.entry*	specifies the report definition to use.

EXAMPLE

Use the stored report definition REPORT.REP_DEF.DETAIL1.REPT to generate the simple list report in the REPORT window.

```
proc report data = report.july1998
            report = report.rep_def.detail1;
run;
```

Enhancing the Appearance of the Detail Report

You can further enhance the appearance of the simple list report by

- controlling the appearance of columns and column headers
- specifying report options
- setting system options
- adding titles and footnotes.

Modifying Column Appearance

The DEFINITION Window

The DEFINITION window displays the characteristics associated with an item in the report and lets you change them. To open the DEFINITION window, select an item in the REPORT window and then select **Edit - Define ...**

DEFINITION Window

The DEFINITION window fields display information about the selected variable:

- **Usage** controls how the report uses the variable:
  - DISPLAY defines the selected item as a display variable. DISPLAY is the default for character variables.
  - ORDER defines the selected item as an ORDER variable.
  - GROUP defines the selected item as a GROUP variable.
  - ACROSS defines the selected item as an ACROSS variable.
  - ANALYSIS defines the selected item as an ANALYSIS variable. You must specify a statistic for an analysis variable. ANALYSIS is the default for numeric variables.
  - COMPUTED defines the selected item as a COMPUTED variable. Computed variables are variables that you define for the report. They are not in the input SAS data set, and PROC REPORT does not add them to the input SAS data set. However, computed variables are included in an output SAS data set if you create one.

- **Attributes** control the display of the variable:
  - Format = assigns a SAS or user-defined format to the item.
  - Spacing = defines the number of blank characters to leave between the column being defined and the column immediately to its left.
  - Width = defines the width of the column in which PROC REPORT displays the selected item.
  - Statistic = associates a statistic with an analysis variable. You must associate a statistic with every analysis variable in its definition. PROC REPORT uses the statistic that you specify to calculate values for the analysis variable for the observations represented by each cell of the report.

    You can use the following values:

N	NMISS	MEAN
STD	MIN	MAX
RANGE	SUM	USS
CSS	STDERR	CV
T	PRT	VAR
SUMWGT	PCTN	PCTSUM

  - Order = orders the values of a GROUP, ORDER, or ACROSS variable according to the specified order:
    - DATA orders values according to their order in the input SAS data set.
    - FORMATTED orders values by their formatted (external) values. By default, the order is ascending.
    - FREQ orders values by ascending frequency count.
    - INTERNAL orders values by their unformatted values, which yields the same order that PROC SORT would yield. This order is operating environment-dependent. This sort sequence is particularly useful for displaying dates chronologically.
  - Justify = justifies the placement of the column header and of the values of the item that you are defining within a column in one of three ways:
    - LEFT left-justifies the formatted values of the item that you are defining within the column width and left-justifies the column header over the values.
    - RIGHT right-justifies the formatted values of the item that you are defining within the column width and right-justifies the column header over the values.
    - CENTER centers the formatted values of the item that you are defining within the column width and centers the column header over the values. This option has no effect on the setting of the SAS system option CENTER.
  - Data type = shows you if the report item is numeric or character. You cannot change this field.
  - Item help = references a HELP or CBT entry that contains help information for the selected item. Use PROC BUILD in SAS/AF software to create a HELP or CBT entry for a report item.
  - Alias = creates an alias for the report item that you are defining. Aliases let you distinguish between different uses of the same report item.

- **Options** control special display characteristics for the variable:
  - NOPRINT suppresses the display of the item that you are defining.
  - NOZERO suppresses the display of the item that you are defining if its values are all zero or missing.
  - DESCENDING reverses the order in which PROC REPORT displays rows or values of a GROUP, ORDER, or ACROSS variable.
  - PAGE inserts a page break just before printing the first column containing values of the selected item.
  - FLOW wraps the value of a character variable in its column. The FLOW option honours the split character. If the text contains no split character, PROC REPORT tries to split text at a blank.
  - ID column specifies that the item that you are defining is an ID variable. An ID variable and all columns to its left appear at the left of every page of a report. ID ensures that you can identify each row of the report when the report contains more columns than will fit on one page.
- **Color** specifies the color of the column. You can specify the following colors:

BLUE	RED	PINK
GREEN	CYAN	YELLOW
WHITE	ORANGE	BLACK
MAGENTA	GRAY	BROWN

- **Header =** specifies the column header. You can type a new header over the existing value in this field.

The DEFINE Statement

You can also use the DEFINE statement to associate characteristics with an item in the report.

SYNTAX

```
DEFINE report-item / usage
                     attribute(s)
                     option(s)
                     justification
                     COLOR = color
                     "column-header";
```

usage — specifies how to use a report item:

DISPLAY|ORDER|GROUP|ACROSS|COMPUTED

attribute(s) — specify attributes for a report item:

FORMAT = *format-name*
SPACING = *horizontal-positions*
WIDTH = *column-width*
statistic (SUM, MEAN, …)
ORDER = DATA|FORMATTED|FREQ|INTERNAL
ITEMHELP = *entry-name*
ALIAS = *alias-name*

option(s) — specify options for a report item:

NOPRINT
NOZERO
DESCENDING
PAGE
FLOW
ID

justification — controls the placement of values and column headers:

LEFT|RIGHT|CENTER

COLOR = *color* — specifies the color of the column header and of the values of the item that you are defining:

BLUE|RED|PINK|GREEN|CYAN|YELLOW|WHITE|ORANGE|BLACK|
MAGENTA|GRAY|BROWN

"column-header" — defines the column header for the report item.

EXAMPLE

Enhance the appearance of the variables S_CITY, P_GRP, P_SUB and SALES as follows:

VARIABLE	DEFINITION	VALUE
S_CITY	**Attributes** Justify = Header =	 CENTER Shop/City
P_GRP	**Attributes** Header =	 Product/Group
P_SUB	**Attributes** Header =	 Product/Subgroup
SALES	**Attributes** Header =	 Sales/(BEF)

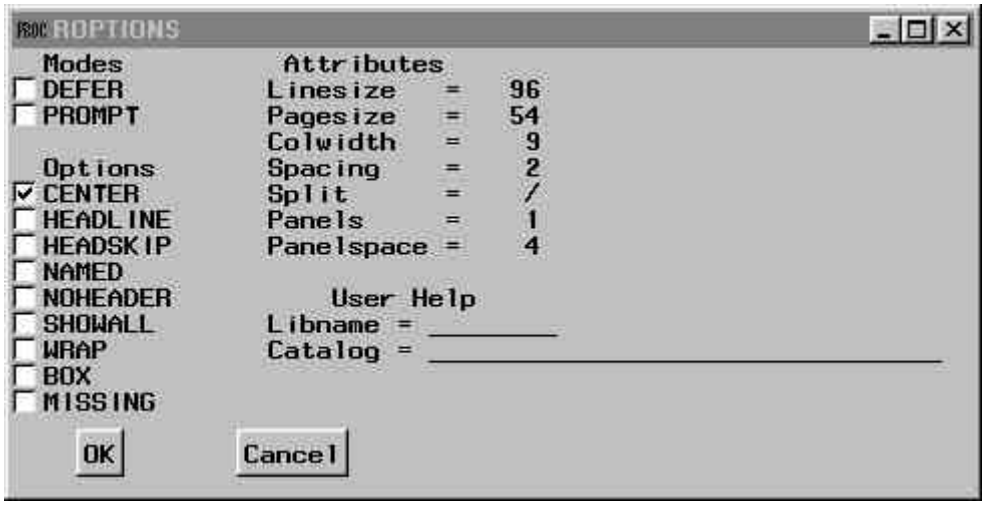
Setting Report Options

You can control some aspects of the page layout of a report by using report options. Report options are local. They only affect the report you are currently creating.

The ROPTIONS Window

The ROPTIONS window displays choices that control the layout and display of the entire report and identifies the SAS data library and catalog containing CBT or HELP entries for items in the report. . To open the ROPTIONS window, select **Tools - Options - Report ...**

ROPTIONS Window



The ROPTIONS window fields display information about the entire report:

- **Modes**
  - DEFER stores the information for changes and makes the changes all at once when you turn DEFER mode off or select **View - Refresh.**

    DEFER is particularly useful when you know that you need to make several changes to the report but do not want to see the intermediate reports.

    By default, PROC REPORT redisplays the report in the REPORT window each time you redefine the report by adding or deleting an item, by changing information in the DEFINITION window, or by changing information in the BREAK window.
  - PROMPT opens the PROMPTER window the next time that you add an item to the report.

♦ **Options**

- CENTER centers the report and summary text (customized break lines). If CENTER is not selected, the report is left-justified.
- HEADLINE underlines all column headers and the spaces between them at the top of each page of the report.
- HEADSKIP writes a blank line beneath all column headers (or beneath the underlining that the HEADLINE option writes) at the top of each page of the report.
- NAMED writes *name* = in front of each value in the report, where *name* is the column header for the value.

  Use NAMED in conjunction with WRAP to produce a report that wraps all columns for a single row of the report onto consecutive lines rather than placing columns of a wide report on separate pages.
- NOHEADER suppresses column headers, including those that span multiple columns.

  Once you suppress the display of column headers in the windowing environment, you cannot select any report items.
- SHOWALL overrides the parts of a definition that suppress the display of a column (NOPRINT and NOZERO). You define a report item with a DEFINE statement or in the DEFINITION window.
- WRAP displays one value from each column of the report, on consecutive lines if necessary, before displaying another value from the first column. By default, PROC REPORT displays values for only as many columns as it can fit on one page. It fills a page with values for these columns before starting to display values for the remaining columns on the next page.

  Typically, you use WRAP in conjunction with NAMED to avoid wrapping column headers.
- BOX uses formatting characters to add line-drawing characters to the report. These characters
  - surround each page of the report
  - separate column headers from the body of the report
  - separate rows and columns from each other.
- MISSING considers missing values as valid values for GROUP, ORDER, or ACROSS variables. Special missing values used to represent numeric values (the letters A through Z and the underscore character) are each considered as a different value. A group for each missing value appears in the report. If you omit the MISSING option, PROC REPORT does not include observations with a missing value for one or more GROUP, ORDER, or ACROSS variables in the report.

♦ **Attributes**

- Linesize = specifies the line size for a report. PROC REPORT honors the first of these linesize specifications that it finds:
  - LS = in the PROC REPORT statement or Linesize = in the ROPTIONS window
  - the LS = setting stored in the report definition loaded with REPORT= in the PROC REPORT statement
  - the SAS system option LINESIZE =.
- Pagesize = specifies the page size for a report. PROC REPORT honors the first of these pagesize specifications that it finds:
  - PS = in the PROC REPORT statement or Pagesize = in the ROPTIONS window
  - the PS = setting stored in the report definition loaded with REPORT = in the PROC REPORT statement
  - the SAS system option PAGESIZE =.
- Colwidth = specifies the default number of characters for columns containing computed variables or numeric data set variables.

  When setting the width for a column, PROC REPORT first looks at WIDTH = in the definition for that column. If WIDTH = is not present, PROC REPORT uses a column width large enough to accommodate the format for the item. If no format is associated with the item, the column width depends on variable type:
  - If the variable is a character variable in the input SAS data set , then the column width is the length of the variable.
  - If the variable is a numeric variable in the input SAS data set, then the column width is the value of the COLWIDTH = option.
  - If the variable is a computed variable (numeric or character), then the column width is the value of the COLWIDTH = option.
- Spacing = specifies the number of blank characters between columns.
- Split = specifies the split character. PROC REPORT breaks a column header when it reaches that character and continues the header on the next line. The split character itself is not part of the column header although each occurrence of the split character counts toward the 40-character maximum for a label.
- Panels = specifies the number of panels on each page of the report. If the width of a report is less than half of the line size, you can display the data in multiple sets of columns so that rows that would otherwise appear on multiple pages appear on the same page. Each set of columns is a panel. A familiar example of this kind of report is a telephone book, which contains multiple panels of names and telephone numbers on a single page.

  When PROC REPORT writes a multi-panel report, it fills one panel before beginning the next.
- Panelspace = specifies the number of blank characters between panels. PROC REPORT separates all panels in the report by the same number of blank characters.

- **User Help**
  - Libname = identifies the library containing user-defined help for the report.
  - Catalog = identifies the catalog containing user-defined help for the report. This help can be in CBT or HELP catalog entries. You can write a CBT or HELP entry for each item in the report with the BUILD procedure in SAS/AF software. You must store all such entries for a report in the same catalog.

    Specify the entry name for help for a particular report item in the DEFINITION window for that report item or in a DEFINE statement.

The PROC REPORT Statement

You can also specify report options in the PROC REPORT statement.

SYNTAX

```
PROC REPORT DATA = SAS-data-set
            mode
            option(s)
            attribute(s)
            user-help;
RUN;
```

mode	PROMPT
option(s)	CENTER|NOCENTER HEADLINE HEADSKIP NAMED NOHEADER|HEADER SHOWALL WRAP BOX MISSING
attribute(s)	LS = *line-size* PS = *page-size* COLWIDTH = *column-width* SPACING = *space-between-columns* SPLIT = "*character*" PANELS = *number-of-panels* PSPACE = *space-between-panels*
user-help	HELP = *libref.catalog*

EXAMPLE

Use the HEADLINE and HEADSKIP report options to add a line under the column headings and skip a line between the headings and the text of the report. Also use the LS = 75 and PS = 65 report options to specify a line size of 75 characters and a page size of 65 lines.

Setting System Options

You can also use system options to control the layout and display of the entire report. System options are global. They stay in effect for the rest of your SAS session or until you change them.

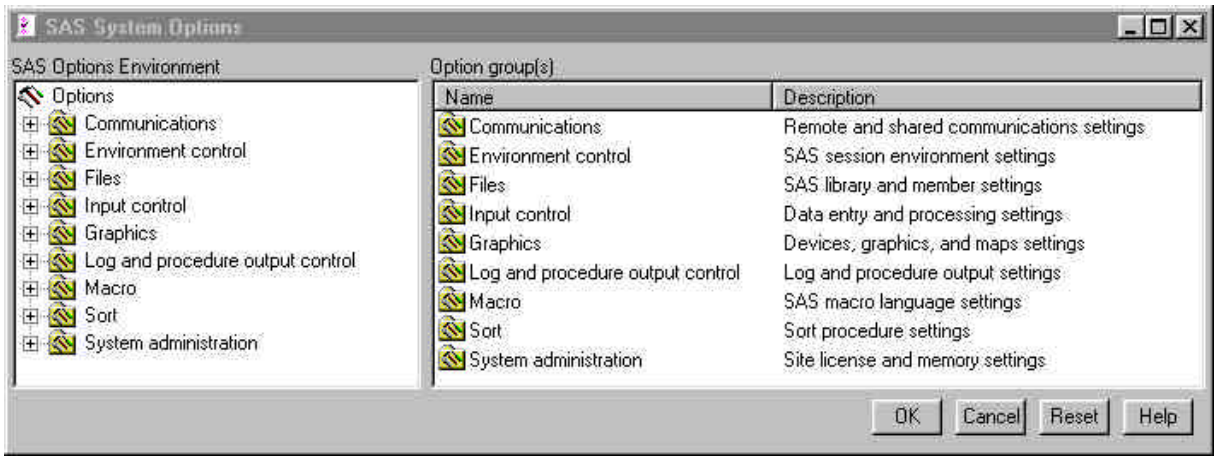
System options are specified

- in the OPTIONS window
- in an OPTIONS statement
- at SAS invocation.

The SAS System Options Window

You can use the SAS System Options window to change SAS system option settings. Options are grouped by function and each option group has at least one subgroup. A description of the kind of options that are located in the group or subgroup appears in the right side of the window. To find individual options, open a group and select the appropriate subgroup. The options in that subgroup are listed in the right side of the window. To open the SAS System Options window, select **Tools - Options - System …**

SAS System Options Window

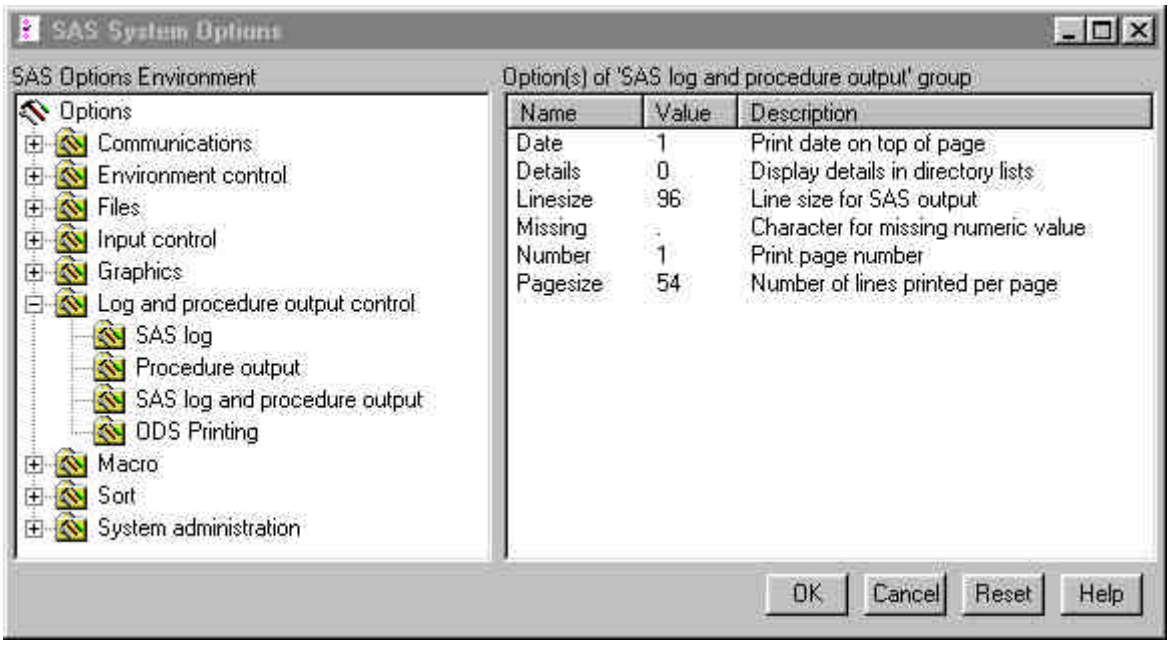


Steps

1. Open the group **Log and procedure output control** and select the subgroup **SAS Log and procedure output**.
2. Select the option that you want to change and select **Modify Value** or **Set to Default** from the pop-up menu.
3. If you choose **Modify Value**, a dialog box appears in which you can edit the option value. You can also double-click the option to modify the value in some operating environments.

   If you choose **Set to Default**, the value is reset to the default SAS value.
4. Option changes are saved only after you select **OK** in the main System Options window. If you enter an incorrect value in the Modify Value dialog, an error message will appear only after you try to save the option by selecting **OK** in the main System Options window.

SAS System Options Window



System option changes remain in effect for the duration of the current SAS session. Default system option settings are used when the SAS System is restarted.

Selecting **Reset** returns all changed options to their previous values.

The OPTIONS Statement

System options are also often specified in an OPTIONS statement.

SYNTAX

```
OPTIONS options;
```

Selected options include:

OPTION	MEANING
(NO)DATE	controls whether the date and time that the SAS job began are printed at the top of each page of the SAS log and any print file created by the SAS System.
(NO)NUMBER	controls whether the page number prints on the first title line of each page of printed output.
PAGENO = *n*	specifies a beginning page number for the next page of printed output.

EXAMPLE

Suppress the date, time, and page number on the report.

```
options nodate nonumber;
```

Adding Titles and Footnotes

You can further enhance the appearance of the simple list report by adding titles and footnotes.

TITLE Statements

TITLE statements specify title lines to be printed on the report. Each TITLE statement specifies one title line. If no title is specified, the default title (The SAS System) is automatically printed.

SYNTAX

```
TITLEn "text";
```

n specifies the relative line containing the title line. *N* can range from 1 to 10. The title line with the highest number appears on the bottom line. If you omit *n*, the SAS System assumes a value of 1. Therefore, TITLE or TITLE1 may be specified for the first title line. Skipping some values of *n* in a series of TITLE statements causes the corresponding lines to be blank.

"text" specifies the text of the title enclosed in single or double quotes. The text of each title can be up to 200 characters long.

A TITLE statement takes effect when the step with which it is associated executes. Once you specify a title for a line, it is used for all subsequent output until you cancel the title or define another title for that line. A TITLE statement for a given line cancels the previous TITLE statement for that line and for all lines with larger *n* numbers.

To cancel all existing titles, specify a TITLE statement without the *n* value:

```
TITLE;
```

To suppress the n^{th} title and all titles below it, use the following statement:

```
TITLEn;
```

The SAS System also allows you to create titles with the display manager TITLES window.

EXAMPLE

Add the following titles to the simple list report.

```
title1 "SOLID Stores";
title2 "============";
title3 " ";
title4 "Sales Figures (July 1998)";
title5 "-------------------------";
title6 " ";
```

FOOTNOTE Statements

FOOTNOTE statements print lines of text at the bottom of the report. No footnote is printed, unless one is specified.

SYNTAX

```
FOOTNOTEn "text";
```

n specifies the relative line to be occupied by the footnote. For footnotes, lines are pushed up from the bottom. The FOOTNOTE statement with the highest number appears on the bottom line. *N* can range from 1 to 10. If you omit *n*, the SAS System assumes a value of 1. As with the TITLE statement, you can print blank footnote lines by skipping over some values of *n*.

"text" specifies the text of the footnote enclosed in single or double quotes. The text of each footnote can be up to 200 characters long.

A FOOTNOTE statement takes effect when the step with which it is associated executes. Once you specify a footnote for a line, the SAS System repeats the same footnote on all pages until you cancel or redefine the footnote for that line. When a FOOTNOTE statement is specified for a given line, it cancels the previous FOOTNOTE statement for that line and for all footnote lines with larger *n* numbers.

To cancel all existing footnotes, specify a FOOTNOTE statement without the *n* value:

```
FOOTNOTE;
```

To suppress the n^{th} footnote and all footnotes below it, use the following statement:

```
FOOTNOTEn;
```

The SAS System also allows you to create footnotes with the display manager FOOTNOTES window.

Refreshing the Report Display

There are 2 types of modifications you can make to a report:

- Modifications that affect the report definition and are specific to the Design Report tool, such as assigning a format or deleting a column. These modifications cause the window to be redisplayed or refreshed automatically.
- Modifications that do not affect the report definition and are global to the SAS environment, such as changing system options or adding a title or footnote. These modifications require you to redisplay the report manually by selecting **View - Refresh**.

You can turn off the automatic redisplay process by activating the DEFER option in the ROPTIONS window. The DEFER option

- prevents the REPORT procedure from redisplaying your report after every change
- saves time if the redisplay process is slow
- prevents error conditions from occurring while a sequence of interrelated changes is made.

EXAMPLE

Select **View - Refresh** to refresh the report display. Notice that the date, time, and page number are suppressed. Also notice that the new titles are displayed.

Partial PROC REPORT Output

SOLID Stores
============

Sales Figures (July 1998)

Shop City	Product Group	Product Subgroup	Sales (BEF)
B-A	Beverages	Waters	26.136
B-A	Beverages	Fruit Juices	18.159
B-A	Beverages	Soft Drinks	143.016
B-A	Beverages	Beers	123.917
B-A	Beverages	Wines	10.272
B-A	Beverages	Alcoholic Drinks	46.593
B-A	Butchery	Pork	16.530
B-A	Butchery	Veal	36.989
B-A	Butchery	Lamb	32.245
B-A	Butchery	Chicken	16.327

PROC REPORT Language Statements

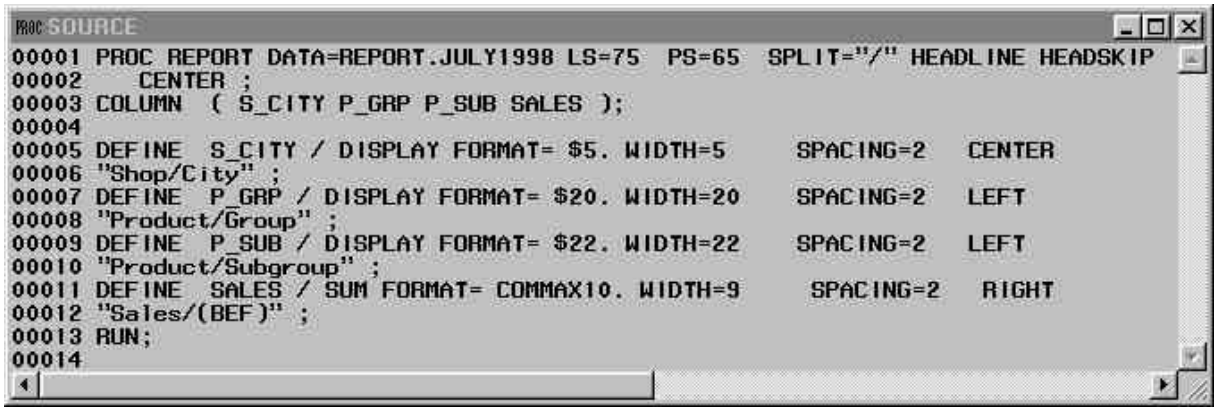
PROC REPORT language statements

- are a set of instructions used to produce a report
- do not contain data or lines of output
- can be stored in external files or SAS catalogs
- can be used as a starting point in creating other, similar reports
- can be directly viewed and edited by the user
- can include titles, footnotes, system options and all report options.

Viewing the PROC REPORT Language Statements

When you build a report in the windowing environment, the Design Report tool automatically generates the PROC REPORT language statements behind the scenes. To view these generated PROC REPORT language statements, open the SOURCE window by selecting **Tools - Report Statements**.

SOURCE Window



Storing the PROC REPORT Language Statements

From the SOURCE window, you can store the PROC REPORT language statements in an external file by selecting **File - Save as …** or in a SAS catalog entry with type of SOURCE by selecting **File - Save as Object …**

EXAMPLE

Store the generated PROC REPORT language statements in the following SOURCE catalog entry:

ENTRY	VALUE
DESTINATION	REPORT.REP_PGM
NAME	DETAIL2
DESCRIPTION	Enhanced List Report
TYPE	SOURCE

EXAMPLE

Close the SOURCE window and store the report definition in the following REPT catalog entry:

FIELD	VALUE
LIBNAME	REPORT
CATALOG	REP_DEF
REPORT NAME	DETAIL2
DESCRIPTION	Enhanced List Report

Comparing Storage Methods

The following table compares the advantages and disadvantages of report definitions and PROC REPORT language statements:

	REPORT DEFINITION	PROC REPORT STATEMENTS
Can be easily edited?	NO	YES
Is partially compiled?	YES	NO
Can be used as a starting point in creating other reports?	YES	YES
Can include titles, footnotes, system options and subsetting criteria?	NO	YES
Can be stored in a SAS catalog?	YES (REPT)	YES (SOURCE)
Can be stored in an external file?	NO	YES
Does not contain data or output?	YES	YES

Ordering and Totaling Rows

You can further enhance the previous list report by

- ordering the rows within the report
- adding totals for the entire report
- adding subtotals.

EXAMPLE

Use the stored report definition REPORT.REP_DEF.DETAIL2.REPT to generate the enhanced list report in the REPORT window. The stored report definition does not include any titles, footnotes, system options, and subsetting criteria. So also respecify the desired system options and titles.

```
options nodate nonumber;

title1 "SOLID Stores";
title2 "============";
title3 " ";
title4 "Sales Figures (July 1998)";
title5 "-------------------------";
title6 " ";

proc report data = report.july1998
            report = report.rep_def.detail2;
run;
```

Defining an ORDER Variable

You can define one or more ORDER variables to control the ordering of the rows and suppress repetitive values in your report.

When you define a variable as an ORDER variable

- rows are arranged in either ascending or descending order of the variable values
- repetitious printing of variable values is suppressed.

The Design Report tool builds reports from left to right. Therefore, to order rows you should define the left-most variable as an ORDER variable.

You can define more than one variable as an ORDER variable, but ORDER variables must precede other variables in the report. When you define an ORDER variable, any DISPLAY variables to the right of the ORDER variables are automatically redefined as ORDER variables.

EXAMPLE

Use the input SAS data set WORK.SAMPLE to illustrate the usage of ORDER variables.

WORK.SAMPLE

S_CTRY	S_TYPE	SALES
B	R	100
NL	W	200
NL	R	300
B	W	400
NL	R	500
B	W	600
B	R	700
NL	W	800
NL	R	900
B	W	1000

Define S_CTRY and S_TYPE as DISPLAY variables.

```
column s_ctry s_type sales;
define s_ctry / display;
define s_type / display;
```

PROC REPORT Output

S_CTRY	S_TYPE	SALES
B	R	100
NL	W	200
NL	R	300
B	W	400
NL	R	500
B	W	600
B	R	700
NL	W	800
NL	R	900
B	W	1000

Define S_CTRY as an ORDER variable and S_TYPE as a DISPLAY variable.

```
column s_ctry s_type sales;
define s_ctry / order;
define s_type / display;
```

PROC REPORT Output

S_CTRY	S_TYPE	SALES
B	R	100
	W	400
	W	600
	R	700
	W	1000
NL	W	200
	R	300
	R	500
	W	800
	R	900

Define S_CTRY and S_TYPE as ORDER variables.

```
column s_ctry s_type sales;
define s_ctry / order;
define s_type / order;
```

PROC REPORT Output

S_CTRY	S_TYPE	SALES
B	R	100
		700
	W	400
		600
		1000
NL	R	300
		500
		900
	W	200
		800

EXAMPLE

Change the usage definition of the variable S_CITY from DISPLAY to ORDER.

Partial PROC REPORT Output

SOLID Stores
============

Sales Figures (July 1998)

Shop City	Product Group	Product Subgroup	Sales (BEF)
B-A	Bakery	Bread	29.849
		Pastry	18.555
	Beverages	Alcoholic Drinks	46.593
		Beers	123.917
		Fruit Juices	18.159
		Soft Drinks	143.016
		Waters	26.136
		Wines	10.272
	Butchery	Chicken	16.327
		Fish	50.364

Open the SOURCE window to display the PROC REPORT language statements. Notice that the DISPLAY variables P_GRP and P_SUB to the right of the ORDER variable S_CITY are also automatically redefined as ORDER variables.

```
define s_city / order format = $5. width = 5 spacing = 2 center
                "Shop/City";
define p_grp / order format = $20. width = 20 spacing = 2 left
               "Product/Group";
define p_sub / order format = $22. width = 22 spacing = 2 left
               "Product/Subgroup";
```

Values of S_CITY, P_GRP, and P_SUB are arranged in ascending order with repetitive values suppressed.

Modifying the Default Ordering Sequence

The default sort sequence for an ORDER variable is ascending. To modify the default ordering sequence, select DESCENDING from the Options list in the DEFINITION window.

You can also specify the sort order for the values of an ORDER variable. You can specify 4 different sort orders using the Order = attribute in the DEFINITION window:

- DATA orders values according to their order in the input SAS data set.
- FORMATTED orders values by their formatted (external) values. By default, the order is ascending.
- FREQ orders values by ascending frequency count.
- INTERNAL orders values by their unformatted values, which yields the same order that PROC SORT would yield. This order is operating environment-dependent. This sort sequence is particularly useful for displaying dates chronologically.

Creating Break Lines

Break lines are lines of text (including blanks) that appear in particular locations, or breaks, of a report. A report can contain multiple breaks. You can use break lines to separate parts of a report visually, to summarize information, or both.

You can place break lines

- wherever the value of an ORDER or GROUP variable changes
- at the beginning or end of a report.

Break lines can contain

- text and break group separators
- summaries of statistics
- report variables
- computed variables
- macro variables.

EXAMPLE

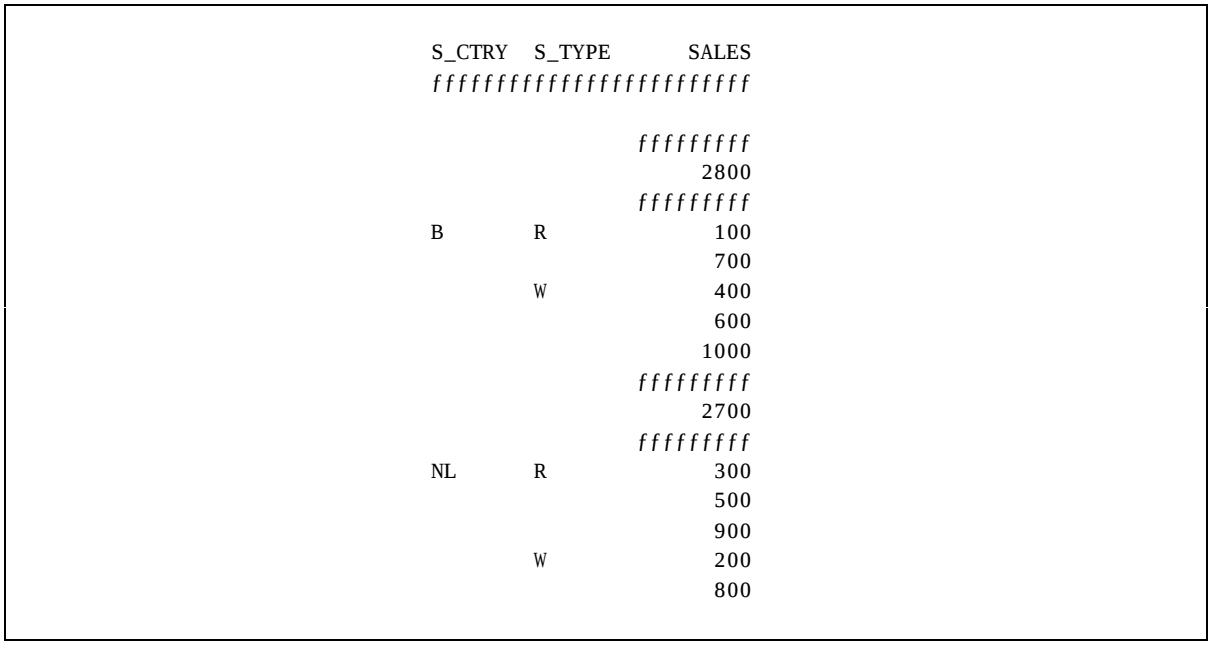
Use the input SAS data set WORK.SAMPLE to illustrate report breaks.

WORK.SAMPLE

S_CTRY	S_TYPE	SALES
B	R	100
NL	W	200
NL	R	300
B	W	400
NL	R	500
B	W	600
B	R	700
NL	W	800
NL	R	900
B	W	1000

Place a subtotal before each unique value of the ORDER variable S_CTRY.

Break Before Detail

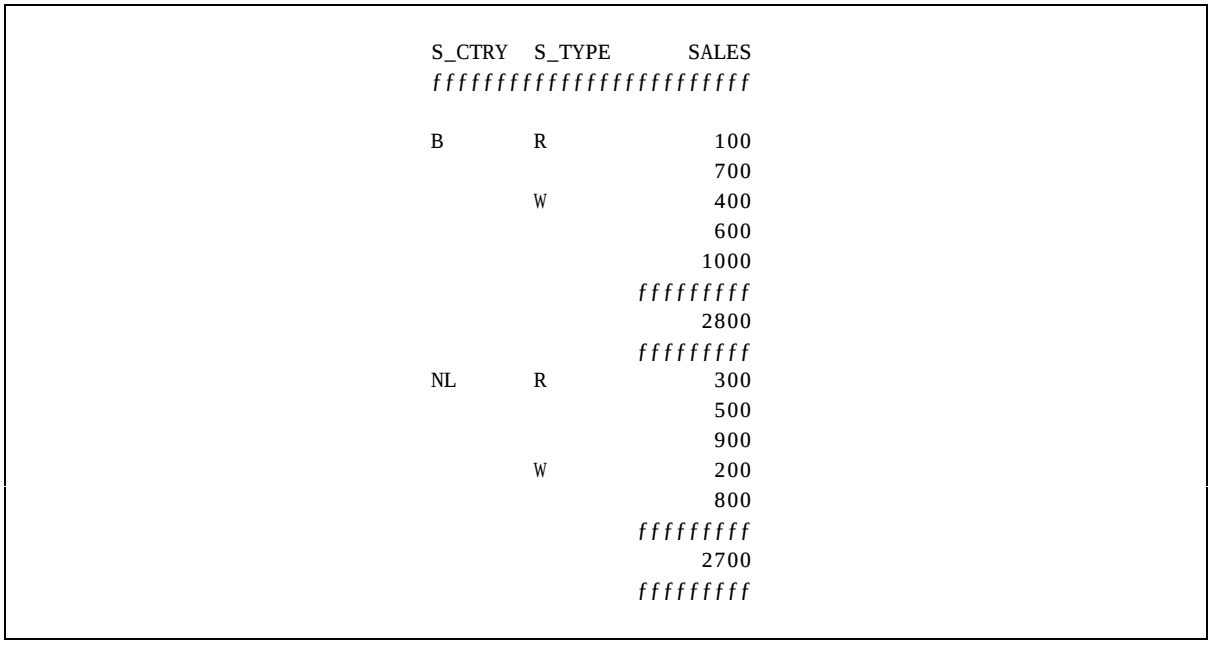


```
S_CTRY  S_TYPE      SALES
ffffffffffffffffffffffff

                  fffffffff
                       2800
                  fffffffff
B       R              100
                       700
        W              400
                       600
                      1000
                  fffffffff
                       2700
                  fffffffff
NL      R              300
                       500
                       900
        W              200
                       800
```

Place a subtotal after each unique value of the ORDER variable S_CTRY.

Break After Detail

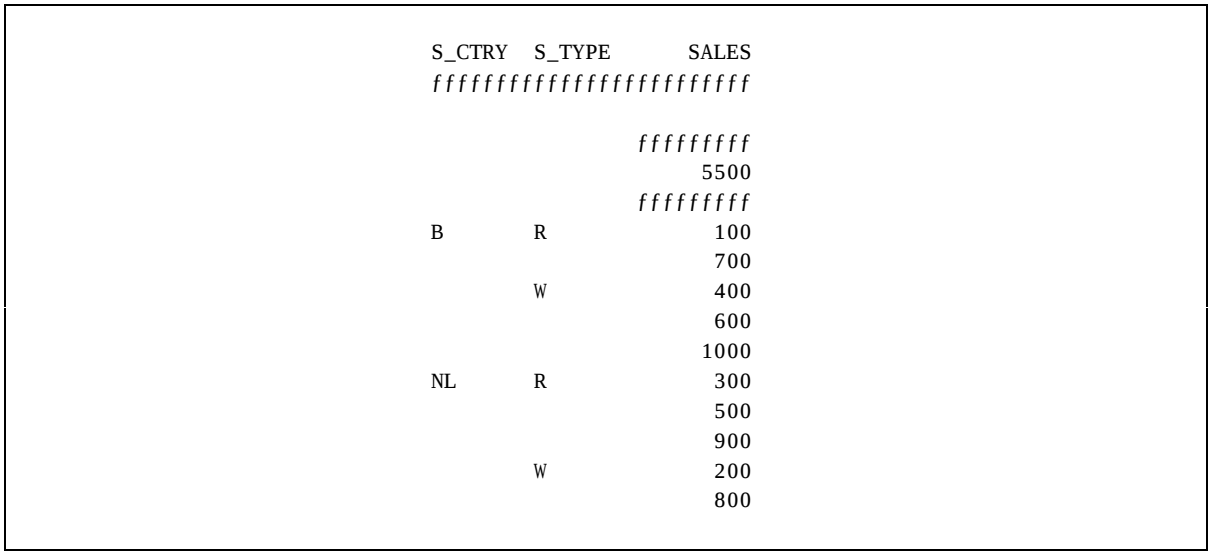


```
S_CTRY  S_TYPE      SALES
ffffffffffffffffffffffff

B       R              100
                       700
        W              400
                       600
                      1000
                  fffffffff
                       2800
                  fffffffff
NL      R              300
                       500
                       900
        W              200
                       800
                  fffffffff
                       2700
                  fffffffff
```

Place a grand total before the detail rows of the report.

Report Break Before Detail

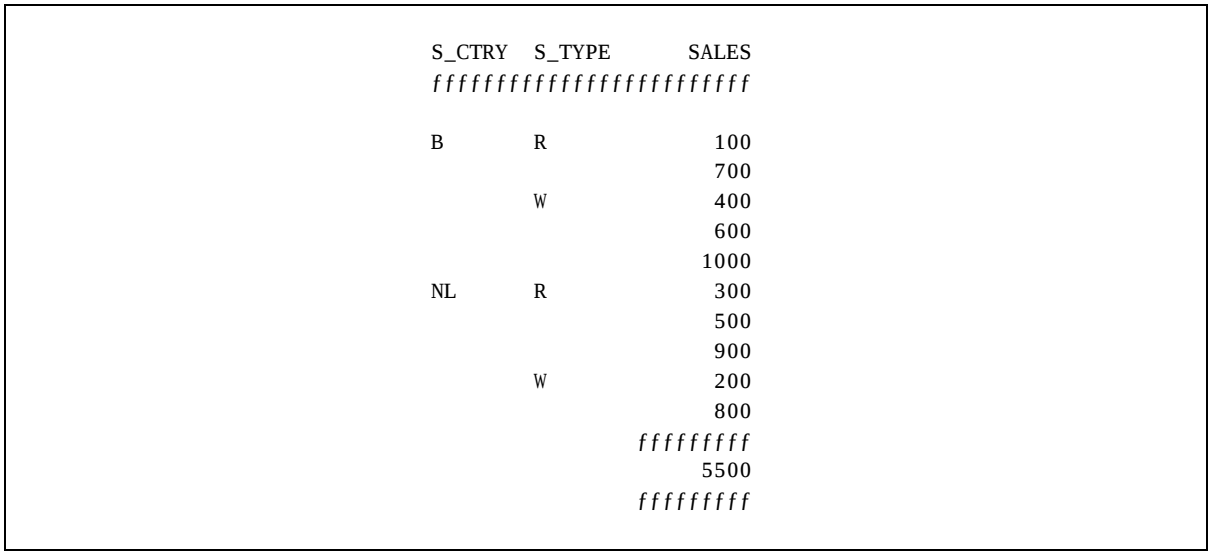


```
S_CTRY  S_TYPE      SALES
ffffffffffffffffffffffff

                 fffffffff
                      5500
                 fffffffff
B       R              100
                       700
        W              400
                       600
                      1000
NL      R              300
                       500
                       900
        W              200
                       800
```

Place a grand total after the detail rows of the report.

Report Break After Detail



```
S_CTRY  S_TYPE      SALES
ffffffffffffffffffffffff

B       R              100
                       700
        W              400
                       600
                      1000
NL      R              300
                       500
                       900
        W              200
                       800
                 fffffffff
                      5500
                 fffffffff
```

The BREAK Window

The BREAK window controls actions at a change in the value of a GROUP or ORDER variable or at the top or bottom of a report. To open the BREAK window, select **Edit - Summarize Information**. PROC REPORT offers you four choices for the location of the break:

- Before Item
- After Item
- At the Top
- At the Bottom

After you select a location, the BREAK window opens.

BREAK Window

To create a break before or after detail lines (when the value of a GROUP or ORDER variable changes), you must select a variable before you open the BREAK window.

Once you identify the break variable and break location, you can

- select break group separators
- request summarization
- specify text to appear on break lines
- compute your own summary variables for specific columns.

Within the BREAK window, you can select any of these options to control what appears at the break points:

- **Overline summary** prints a single line above information at each break point.
- **Double overline summary** prints a double line above information at each break point.
- **Underline summary** prints a single line below information at each break point.
- **Double underline summary** prints a double line below information at each break point.
- **Skip line after break** places a blank line at each break point.
- **Page after break** places each break group on a separate page.
- **Summarize analysis columns** summarizes analysis and computed variables at each break point.
- **Suppress break value** suppresses the printing of the break variable at each break point.

The BREAK and RBREAK Statements

The BREAK statement enables you to place text or values

- before each unique value of an ORDER or GROUP variable
- after each unique value of an ORDER or GROUP variable.

The RBREAK statement enables you to place text or values

- before the detail rows of a report
- after the detail rows of a report.

SYNTAX

```
BREAK location break-variable / option(s);
RBREAK location / option(s);
```

location controls the placement of the break lines:

AFTER	places the break lines immediately after the last row of each set of rows that have the same value for the break variable.
BEFORE	places the break lines immediately before the first row of each set of rows that have the same value for the break variable.

break-variable is a GROUP or ORDER variable. The REPORT procedure writes break lines each time the value of this variable changes.

option(s) specify options for the break line:

OL	prints a single line above information at each break point.
DOL	prints a double line above information at each break point.
UL	prints a single line below information at each break point.
DUL	prints a double line below information at each break point.
SKIP	places a blank line at each break point.
PAGE	places each break group on a separate page.
SUMMARIZE	summarizes analysis and computed variables at each break point.
SUPPRESS	suppresses the printing of the break variable at each break point.
COLOR = *color*	specifies the color of the break lines in the REPORT window. You can use the following colors: BLUE|RED|PINK|GREEN|CYAN|YELLOW|WHITE ORANGE|BLACK|MAGENTA|GRAY|BROWN

EXAMPLE

Display a grand total at the bottom of the previously generated report. Select **Edit - Summarize Information - At the Bottom**. Notice that it is not necessary to select a variable before generating a grand total for a report. The BREAK window opens. Within the BREAK window, select the following options:

- Double overline summary
- Double underline summary
- Summarize analysis columns
- Color = RED

Also include subtotals for each shop city. First select the variable **Shop City** and then select **Edit - Summarize Information - After Item**. Notice that now it is necessary to select a variable before generating subtotals for a report. The BREAK window opens. Within the BREAK window, select the following options:

- Overline summary
- Underline summary
- Skip line after break
- Summarize analysis columns
- Color = BLUE

Finally also include subtotals for each product group within each shop city. First select the variable **Product Group** and then select **Edit - Summarize Information - After Item**. The BREAK window opens. Within the BREAK window, select the following options:

- Overline summary
- Skip line after break
- Summarize analysis columns
- Suppress break value
- Color = GREEN

Open the SOURCE window to display the PROC REPORT language statements.

```
break after p_grp / ol skip summarize suppress color = green;
break after s_city / ol ul skip summarize color = blue;
rbreak after / dol dul summarize color = red;
```

Paging Through the Report

You must page through the report to see the break lines. By selecting **View**, PROC REPORT offers you the following choices to scroll through the report:

- Next Page
- Previous Page
- Display Page ...
- Scroll Down
- Scroll Up
- Scroll Left
- Scroll Right

Partial PROC REPORT Output

```
                              SOLID Stores
                              ============

                          Sales Figures (July 1998)
                          -------------------------


Shop   Product              Product                       Sales
City   Group                Subgroup                      (BEF)
fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff

 B-A   Bakery               Bread                        29.849
                            Pastry                       18.555
                                                      fffffffffff
                                                         48.404

                                  ...

       Vegetables           Canned Vegetables            64.476
                            Deep-freeze Vegetables       23.178
                            Fresh Vegetables             31.603
                                                      fffffffffff
                                                        119.257

fffff                                                 fffffffffff
NL-R                                                   2.455.700
fffff                                                 fffffffffff

                                                      ===========
                                                      14.366.965
                                                      ===========
```

Store the report definition in the following REPT catalog entry:

FIELD	VALUE
LIBNAME	REPORT
CATALOG	REP_DEF
REPORT NAME	DETAIL3
DESCRIPTION	List Report with Totals and Subtotals

Adding Variables to the Report

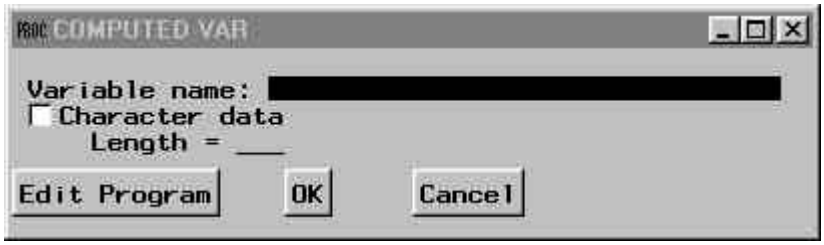
Adding a Computed Variable

You can create a computed column based on the values in other columns of your report. For example, you might want a column which shows the sales figure expressed in EURO instead of in BEF. Computed columns exist in the report definition but are not added as variables to the underlying SAS data set.

The COMPUTED VAR and COMPUTE Windows

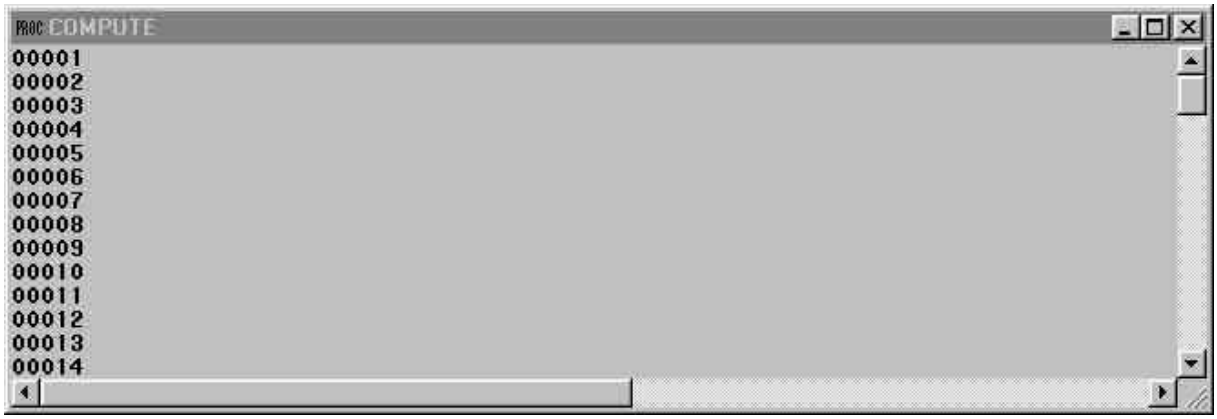
The COMPUTED VAR window adds a variable that is not in the input SAS data set to the report. To add a computed column to the report, first select a column and then select **Edit - Add Item - Computed Column.** After you select **Computed Column**, PROC REPORT prompts you for the location of the computed column relative to the column that you have selected. After you select a location, the COMPUTED VAR window opens.

COMPUTED VAR Window



Enter the name of the variable at the prompt. If it is a character variable, select the **Character data** check box and, if you want, enter a value in the **Length** field. The length can be any integer between 1 and the linesize. If you leave the field blank, PROC REPORT assigns a default length of 8 to the variable. After you enter the name of the variable, select **Edit Program** to open the COMPUTE window.

COMPUTE Window



Use programming statements in the COMPUTE window to define the computed variable. Statements in the COMPUTE window

- usually include an assignment statement
- can include most DATA step statements
- can only reference variables contained in the report definition or temporary variables created in the current COMPUTE window
- must reference analysis variables by *`variable-name.statistic`*
- end with semicolons.

After closing the COMPUTE and COMPUTED VAR windows, open the DEFINITION window to describe how to display the computed variable.

The position of a computed variable is important. PROC REPORT assigns values to the columns in a row of a report from left to right. Consequently, you cannot base the calculation of a computed variable on any variable that appears to its right in the report.

The COMPUTE Statement

To compute a column in a report,

- decide on a name for the column and place it in the COLUMN statement
- use a DEFINE statement to identify the column's attributes (the usage attribute must be COMPUTED)
- use a COMPUTE / ENDCOMP code segment to calculate the column values.

SYNTAX

```
COMPUTE computed-variable / type-specification;
   code-segment
ENDCOMP;
```

computed-variable specifies a computed variable to associate the COMPUTE block with. You must include the report item in the COLUMN statement and specify a DEFINE statement for it.

type-specification can specify CHARACTER and LENGTH =.

CHARACTER specifies that the computed variable is a character variable. The default is numeric.

LENGTH = specifies the length of a computed character variable. If you do not specify a length, the variable's length is 8.

code-segment can contain

- DATA step statements
- DATA step functions
- LINE statements.

When you use DATA step statements to reference the current value of an item in a report, the form of the name you use depends on the way you are using the item in the report. For example, to create a computed variable TOTAL, if items X and Y are defined as DISPLAY, ORDER, or GROUP refer to their values by the names of the variables:

```
total = x + y;
```

If items X and Y are defined as ANALYSIS refer to their values by a compound name that includes the name of the variable and the name of the statistic, separated by a period:

```
total = x.sum + y.sum;
```

Hiding Unnecessary Variables

The NOPRINT option in the DEFINITION window or in the DEFINE statement suppresses the display of a report item. Use this option

- if you do not want to show the item in the report but you need to use its values to calculate other values that you use in the report
- to establish the order of rows in the report
- if you do not want to use the item as a column but want to have access to its values in summaries.

The SHOWALL option in the ROPTIONS window or in the PROC REPORT statement overrides all occurrences of NOPRINT.

EXAMPLE

Add a new variable named EURO to the right of the variable SALES which shows the sales figure expressed in EURO instead of in BEF.

Steps

1. First select the variable **Sales (BEF)** and then select **Edit - Add Item - Computed Column - Right**. The COMPUTED VAR window opens.
2. Use the COMPUTED VAR window to name and define the computed variable. Type EURO in the **Variable name:** field.
3. Select **Edit Program** to define the new variable. The COMPUTE window opens. Enter the following SAS statement in this window to create the new variable:

```
euro = sales.sum / 40.3399;
```

   When you close the COMPUTE window, the statements you entered are checked for syntax errors.
4. In the COMPUTED VAR window, select **OK** to close the COMPUTED VAR window and display the values of the new variable.
5. To define the attributes for the new variable EURO, first select the variable **EURO** and then select **Edit - Define ...** The DEFINITION window opens. Modify the default attributes of the variable EURO as follows:

VARIABLE	DEFINITION	VALUE
EURO	**Attributes**	
	Format =	COMMAX10.2
	Width =	10
	Header =	Sales/(EURO)

6. With the Sales (EURO) column in your report, there is no need to display the Sales (BEF) column. To hide the Sales (BEF) column, first select the variable **Sales (BEF)** and then select **Edit - Define ...** In the DEFINITION window, select NOPRINT in the Options list.

Partial PROC REPORT Output

```
                              SOLID Stores
                              ============

                          Sales Figures (July 1998)
                          -------------------------


Shop   Product              Product                      Sales
City   Group                Subgroup                    (EURO)
ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff

 B-A   Bakery               Bread                        739,94
                            Pastry                       459,97
                                                      fffffffffff
                                                        1.199,90

                                  ...


       Vegetables           Canned Vegetables          1.598,32
                            Deep-freeze Vegetables       574,57
                            Fresh Vegetables             783,42
                                                      fffffffffff
                                                        2.956,30

fffff                                                 fffffffffff
NL-R                                                   60.875,20
fffff                                                 fffffffffff

                                                      ===========
                                                      356.147,75
                                                      ===========
```

Open the SOURCE window to display the PROC REPORT language statements.

```
proc report data = report.july1998
            ls = 75 ps = 65 split = "/" headline headskip center;
   column s_city p_grp p_sub sales euro;
   define s_city / order format = $5. width = 5 spacing = 2 center
                    "Shop/City";
   define p_grp / order format = $20. width = 20 spacing = 2 left
                  "Product/Group";
   define p_sub / order format = $22. width = 22 spacing = 2 left
                  "Product/Subgroup";
   define sales / sum format = commax10. width = 10 spacing = 2
                  noprint right "Sales/(BEF)";
   define euro / computed format = commax10.2 width = 10
                 spacing = 2 right "Sales/(EURO)";

   compute euro;
      euro = sales.sum / 40.3399;
   endcomp;

   break after p_grp / ol skip summarize suppress color = green;
   break after s_city / ol ul skip summarize color = blue;
   rbreak after / dol dul skip summarize color = red;
run;
```

Store the report definition in the following REPT catalog entry:

FIELD	VALUE
LIBNAME	REPORT
CATALOG	REP_DEF
REPORT NAME	DETAIL4
DESCRIPTION	List Report with Computed Column

Creating and Applying User-defined Formats

Creating User-defined Formats

Creating Temporary User-defined Formats

In addition to formats provided by the SAS System you can create your own formats using the FORMAT procedure. The VALUE statement in the FORMAT procedure enables you to create formats to establish descriptive labels for coded numeric or character values.

SYNTAX

```
PROC FORMAT;
   VALUE $char-fmt
         'value-1' = 'formatted-value-1'
         'value-2' = 'formatted-value-2'
          …
         'value-n' = 'formatted-value-n';
   VALUE num-fmt
         value-1 = 'formatted-value-1'
         value-2 = 'formatted-value-2'
         …
         value-n = 'formatted-value-n';
RUN;
```

Rules for defining your own character and numeric formats:

- You must specify a format name in the VALUE statement. The format name must be a valid SAS name up to eight characters long, not ending in a number.
- The format name for character variables must have a dollar sign ($) as the first character and no more than seven additional characters.
- A user-defined format cannot have a standard SAS format name.
- Values for character formats are quoted.
- Values for numeric formats are not quoted.
- Formatted values for all formats (character and numeric) are quoted.
- Formatted values can be up to 200 characters long and can contain mixed-case text.
- A character format can be applied only to a character variable.
- A numeric format can be applied only to a numeric variable.
- Refer to the format later by using the format name followed by a period (.).

Creating Permanent User-defined Formats

To avoid recreating formats in subsequent SAS jobs or sessions, you can store formats in a permanent catalog. The LIBRARY = option in the PROC FORMAT statement specifies the data library, and optionally, the catalog in which to store the formats.

SYNTAX

```
PROC FORMAT LIBRARY = libref.catalog;
   VALUE …;
   VALUE …;
   …
   VALUE …;
RUN;
```

- Without the LIBRARY = option, formats are stored in the WORK.FORMATS catalog and only exist for the duration of the SAS session.
- If the LIBRARY = option specifies only a *libref*, formats are permanently stored in *libref*.FORMATS.
- If the LIBRARY = option specifies *libref.catalog*, formats are permanently stored in that catalog.
- Write access to the *libref* data library is required to create formats.

Using Permanent User-defined Formats

To use permanent formats stored in a catalog that is not named FORMATS or to search multiple catalogs, use the FMTSEARCH = system option to identify the catalog(s) to be searched for formats.

SYNTAX

```
OPTIONS FMTSEARCH = (item-1 item-2 … item-n);
```

item is either a libref or libref.catalog.

- If *item* is only a libref, FORMATS is assumed as the catalog name.
- The WORK.FORMATS catalog is always searched first, unless it appears in the FMTSEARCH list.
- If the LIBRARY libref is assigned, the LIBRARY.FORMATS catalog, is searched after WORK.FORMATS and before anything else in the FMTSEARCH list, unless it appears in the list.
- Catalogs in the list are searched in the order in which they appear in the list.

Applying User-defined Formats in PROC REPORT

The Format = field in the DEFINITION window or the FORMAT = option in the DEFINE statement enables you to associate a user-defined format with a variable. When specifying the format, you must include a period in the format name.

You might also need to adjust the Width = field in the DEFINITION window or the WIDTH = option in the DEFINE statement to accommodate the custom format.

EXAMPLE

Enhance the previous report by creating and using a custom format for the variable S_CITY.

Steps

1. Use PROC FORMAT to create a custom format named $CITYFMT for the variable S_CITY. Permanently store the format in the REPORT library in the default FORMATS catalog.

```
proc format lib = report;
   value $cityfmt "B-A"   = "Antwerp"
                  "B-B"   = "Brussels"
                  "B-L"   = "Liège"
                  "B-N"   = "Namur"
                  "NL-A"  = "Amsterdam"
                  "NL-DH" = "Den Haag"
                  "NL-R"  = "Rotterdam";
run;
```

2. Make sure the REPORT.FORMATS catalog is searched for user-defined formats.

```
options fmtsearch = (report);
```

3. Use the DEFINITION window to assign the previously defined custom format to the variable S_CITY. In the DEFINITION window, type $CITYFMT. in the **Format =** field. Also increase the **Width =** field to 9 to accommodate the custom format.

Partial PROC REPORT Output

```
                                SOLID Stores
                                ============

                            Sales Figures (July 1998)
                            -------------------------


  Shop       Product              Product                      Sales
  City       Group                Subgroup                    (EURO)
ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff

Amsterdam  Bakery               Bread                         846,21
                                Pastry                        488,95
                                                           fffffffff
                                                            1.335,15

                                ...

           Vegetables           Canned Vegetables           1.598,32
                                Deep-freeze Vegetables        574,57
                                Fresh Vegetables              783,42
                                                           fffffffff
                                                            2.956,30

fffffffff                                                  fffffffff
Rotterdam                                                   60.875,20
fffffffff                                                  fffffffff

                                                           ==========
                                                           356.147,75
                                                           ==========
```

Note that the rows of the report are now ordered according to the formatted values of S_CITY.

Open the SOURCE window to display the PROC REPORT language statements.

```
define s_city / order format = $cityfmt. width = 9 spacing = 2
                center "Shop/City";
```

Store the report definition in the following REPT catalog entry:

FIELD	VALUE
LIBNAME	REPORT
CATALOG	REP_DEF
REPORT NAME	DETAIL5
DESCRIPTION	List Report with User-defined Format

Subsetting the Report

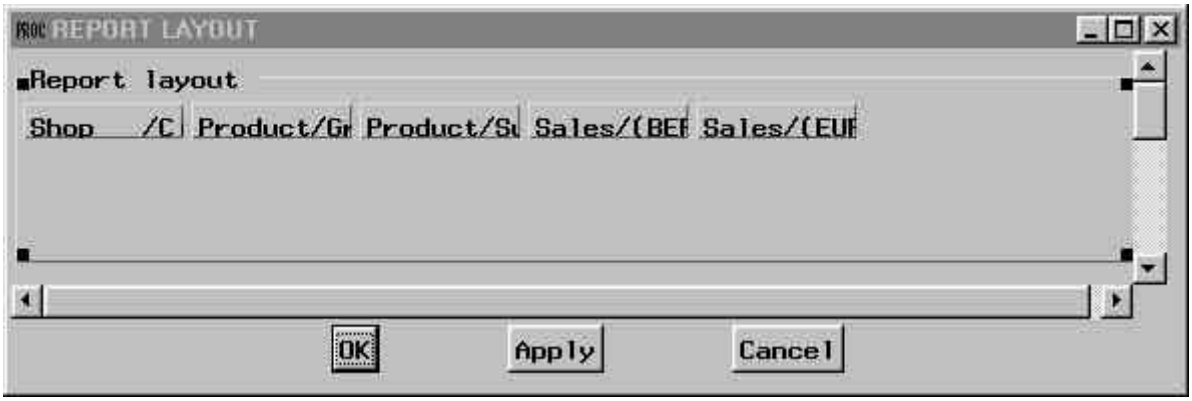
You can subset data for your reports in several ways:

- You can control the variables processed.
- You can limit the number of observations displayed.
- You can select observations that meet a particular condition.

Subsetting the Report Temporarily

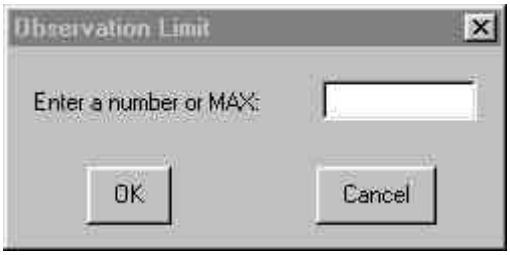
To limit the number of observations displayed in the report without permanently altering the report, you can use features of the REPORT LAYOUT window. To open the REPORT LAYOUT window, specify the **LAYOUT** command.

REPORT LAYOUT Window



By using the REPORT LAYOUT window to limit the number of observations displayed, you can see the effect without permanently changing your report. Select **Edit - Observation Limit ...** to open the OBSERVATION LIMIT window.

OBSERVATION LIMIT Window



Type an observation limit in the **Enter a number or MAX:** field. Select **OK** to close the OBSERVATION LIMIT window and select **Apply** to see the effect of the limit on the report.

The observation limit in the REPORT LAYOUT window remains in effect only until you close the REPORT LAYOUT window or you apply a new limit.

Subsetting the Report Permanently

Data set options enable you to control the data that is displayed in your report. You can use data set options to

- specify a range of observations to use in a report
- control variables processed in a PROC REPORT step.

Unlike system options, data set options do not remain in effect across PROC or DATA steps.

SYNTAX

```
PROC REPORT DATA = SAS-data-set (data-set-options) … ;
```

Selected data set options include:

OPTION	MEANING
FIRSTOBS = *n*	specifies the first observation to process.
OBS = *n*	specifies the last observation to process.
KEEP = *variable-list*	names variables to include in processing.
DROP = *variable-list*	names variables to exclude from processing.

Subsetting the Report Based on a Condition

You can subset the data in your report by specifying one or more WHERE clauses. A WHERE clause enables you to select observations that meet a particular condition before the SAS System brings the observations into the PROC REPORT step. Because the WHERE clause is applied directly to the SAS data set, you can use data set variables only in the WHERE expression, not columns created in your report.

The WHERE expression is a sequence of operands and operators.

Operands can be

- variable names
- functions
- constants.

You can use all of the common comparison operators in a WHERE expression.

MNEMONIC	SYMBOL	MEANING
LT	<	less than
GT	>	greater than
EQ	=	equal to
LE	<=	less than or equal to
GE	>=	greater than or equal to
NE	¬= ^= ~=	not equal to
IN		equal to one of a list

The IN operator is used to compare a value to a list of values. If the value matches one in the list, the expression is TRUE, otherwise the expression is FALSE.

There are also special comparison operators that can be used in a WHERE expression.

MNEMONIC	MEANING
CONTAINS (?)	selects rows that include the substring specified
IS NULL	selects rows in which the value of the column is missing.
BETWEEN - AND	selects rows in which the value of the column falls within a range of values
LIKE	selects rows by comparing character values to specified patterns % replaces any number of characters _ replaces one character
=*	selects rows that contain a spelling variation of the word(s) specified.

You can specify multiple expressions in a WHERE clause by using logical operators.

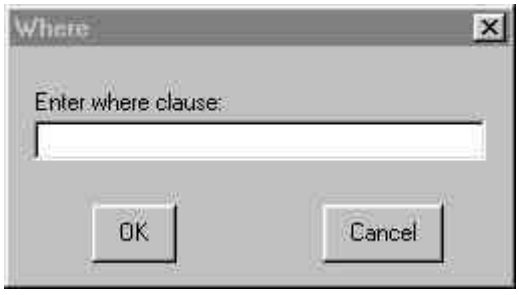
MNEMONIC	SYMBOL	MEANING
AND	&	and - both
OR	|	or - either
NOT	¬	not - negation

By default, the AND operator has a higher priority than the OR operator in the evaluation of expressions. You can use parentheses to change the order of the evaluation of the expressions. Evaluating expressions with parentheses begins at the deepest level of parentheses and moves outward.

The WHERE and WHERE ALSO Windows

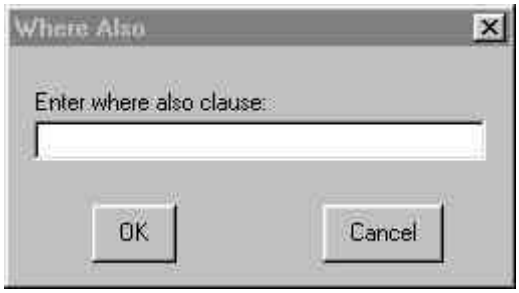
To specify a WHERE expression, select **Subset - Where …**

WHERE Window



To augment the WHERE expression, select **Subset - Where Also…**

WHERE ALSO Window



To clear the last WHERE expression, select **Subset - Undo Last Where**.

To clear all WHERE expressions, select **Subset - Where …** and leave the WHERE expression blank.

WHERE clauses are not stored with the report definition.

The WHERE Statement

The WHERE statement enables you to specify a condition that the data must satisfy before the SAS system brings the observation from an existing SAS data set into the PROC REPORT step.

SYNTAX

```
WHERE expression;
```

WHERE statements are not saved when you store the report definition.

EXAMPLE

First subset the previous report by only selecting the shops located in "Brussels" (B-B) and "Amsterdam" (NL-A). Then further subset the report by only selecting the "Fruits" and "Vegetables" product groups.

Steps

1. Select **Subset - Where ...** to open the WHERE window. Specify a WHERE expression that selects only the shops located in "Brussels" and "Amsterdam". Type the following condition in the **Enter where clause:** field:

```
s_city in ('B-B' 'NL-A')
```

2. Select **Subset - Where Also...** to open the WHERE ALSO window. Specify a WHERE ALSO expression that selects only the "Fruits" and "Vegetables" product groups. Type the following condition in the **Enter where also clause:** field:

```
p_grp in ('Fruits' 'Vegetables')
```

PROC REPORT Output

```
                                SOLID Stores
                                ============

                             Sales Figures (July 1998)
                             -------------------------


  Shop       Product                Product                        Sales
  City       Group                  Subgroup                     (EURO)
ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff

Amsterdam    Fruits                 Canned Fruits                  884,53
                                    Fresh Fruits                 3.260,69
                                                              fffffffffff
                                                               4.145,23

             Vegetables             Canned Vegetables            2.358,06
                                    Deep-freeze Vegetables         887,81
                                    Fresh Vegetables             1.493,41
                                                              fffffffffff
                                                               4.739,28

fffffffff                                                     fffffffffff
Amsterdam                                                      8.884,50
fffffffff                                                     fffffffffff

Brussels     Fruits                 Canned Fruits                  812,25
                                    Fresh Fruits                 2.670,98
                                                              fffffffffff
                                                               3.483,23

             Vegetables             Canned Vegetables            1.916,17
                                    Deep-freeze Vegetables         652,51
                                    Fresh Vegetables             1.077,07
                                                              fffffffffff
                                                               3.645,75

fffffffff                                                     fffffffffff
Brussels                                                       7.128,97
fffffffff                                                     fffffffffff

                                                              ==========
                                                              16.013,48
                                                              ==========
```

Open the SOURCE window to display the PROC REPORT language statements.

```
proc report data = report.july1998
            ls = 75 ps = 65 split = "/" headline headskip center;
   column s_city p_grp p_sub sales euro;
   define s_city / order format = $cityfmt. width = 9 spacing = 2
                    center "Shop/City";
   define p_grp / order format = $20. width = 20 spacing = 2 left
                  "Product/Group";
   define p_sub / order format = $22. width = 22 spacing = 2 left
                  "Product/Subgroup";
   define sales / sum format = commax10. width = 10 spacing = 2
                  noprint right "Sales/(BEF)";
   define euro / computed format = commax10.2 width = 10
                 spacing = 2 right "Sales/(EURO)";

   compute euro;
      euro = sales.sum / 40.3399;
   endcomp;

   break after p_grp / ol skip summarize suppress color = green;
   break after s_city / ol ul skip summarize color = blue;
   rbreak after / dol dul skip summarize color = red;
run;
```

Note that WHERE statements are not displayed in the SOURCE window.

Store the report definition in the following REPT catalog entry:

FIELD	VALUE
LIBNAME	REPORT
CATALOG	REP_DEF
REPORT NAME	DETAIL6
DESCRIPTION	Subsetted List Report

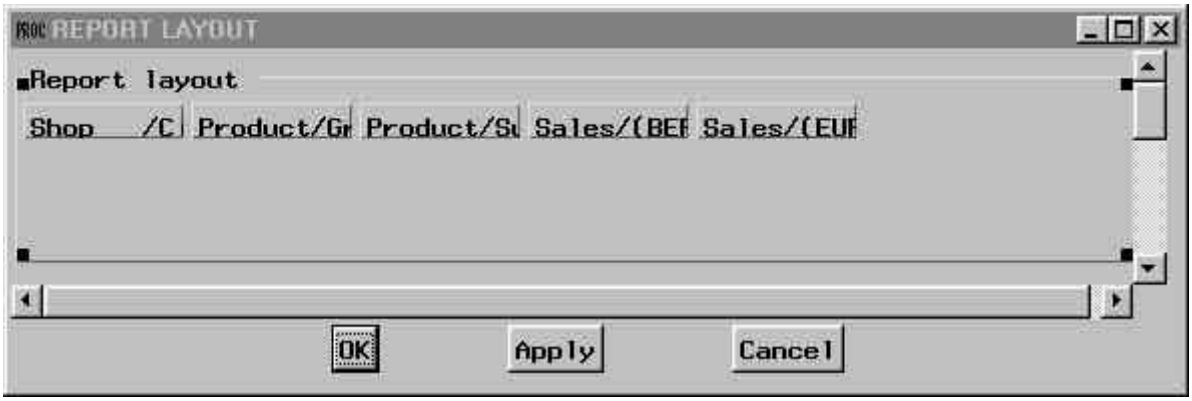
Use the stored report definition REPORT.REP_DEF.DETAIL6.REPT to generate the subsetted list report in the REPORT window. The stored report definition does not include any subsetting criteria. So you need to invoke the report with the equivalent WHERE clauses to reproduce the same report.

```
proc report data = report.july1998
            report = report.rep_def.detail6;
   where s_city in ('B-B' 'NL-A')
         and p_grp in ('Fruits' 'Vegetables');
run;
```

Altering the Report Structure

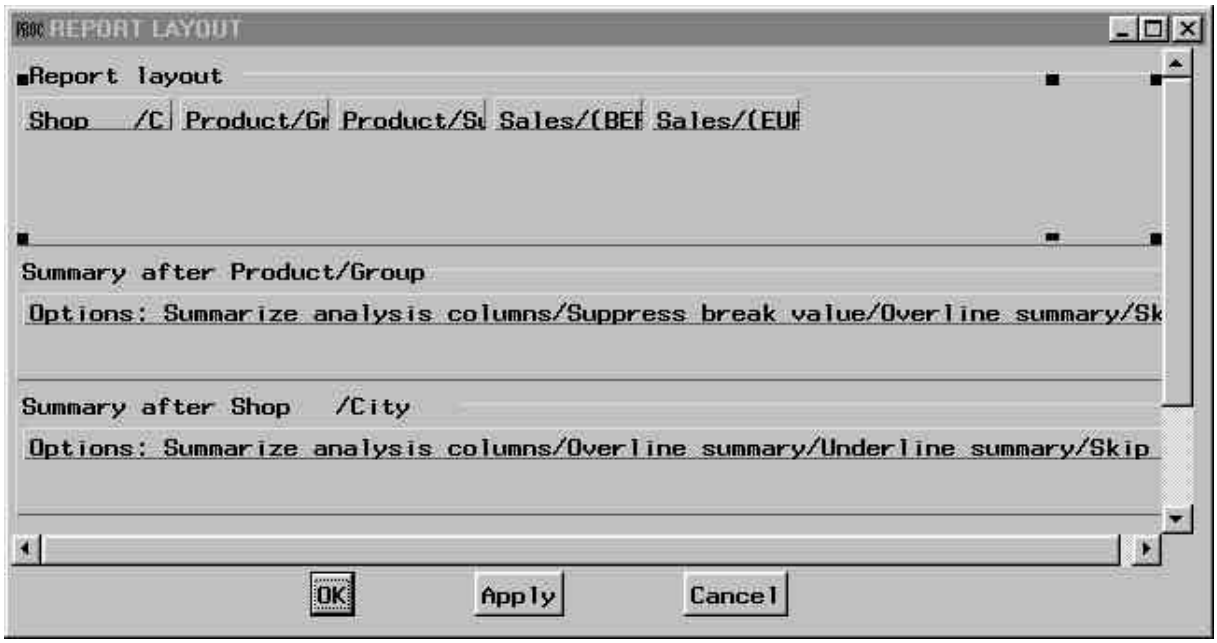
To manipulate the report structure, you can use the REPORT LAYOUT window. To open the REPORT LAYOUT window, specify the **LAYOUT** command.

REPORT LAYOUT Window



Select **Edit - Zoom Layout** to see the entire window.

Zoomed REPORT LAYOUT Window



You can use the drag-and-drop features of the REPORT LAYOUT window or the **Edit** menu to

- move, delete, and add columns
- alter column attributes
- alter report attributes
- move, delete, and add summary lines
- alter summary line attributes
- limit observations.

Creating Summary Reports

Grouping Rows

As the size of your SAS data set increases, it becomes less useful to see a listing that includes each observation. You can create summary reports using the REPORT procedure by defining one or more variables as GROUP variables. When you define a variable as a GROUP variable, observations with the same values of that variable are collapsed into one row of the report.

All of the variables in a summary report must be defined as either GROUP, ANALYSIS, COMPUTED, or ACROSS, because PROC REPORT must be able to summarize all columns across a row in order to collapse rows.

EXAMPLE

Use the input SAS data set WORK.SAMPLE to illustrate the usage of GROUP variables.

WORK.SAMPLE

S_CTRY	S_TYPE	SALES
B	R	100
NL	W	200
NL	R	300
B	W	400
NL	R	500
B	W	600
B	R	700
NL	W	800
NL	R	900
B	W	1000

Define S_CTRY as a GROUP variable.

```
column s_ctry sales;
define s_ctry / group;
```

PROC REPORT Output

```
S_CTRY       SALES
ffffffffffffffff

B              2800
NL             2700
```

You can define more than one variable as a GROUP variable, but GROUP variables must precede other variables. Nesting is determined by the order of the variables in the COLUMN statement. Define S_CTRY and S_TYPE as GROUP variables.

```
column s_ctry s_type sales;
define s_ctry / group;
define s_type / group;
```

PROC REPORT Output

```
S_CTRY  S_TYPE       SALES
fffffffffffffffffffffffff

B       R              800
        W             2000
NL      R             1700
        W             1000
```

Define S_CTRY and S_TYPE as ORDER variables.

```
column s_ctry s_type sales;
define s_ctry / order;
define s_type / order;
```

PROC REPORT Output

```
S_CTRY  S_TYPE   SALES
ffffffffffffffffffff

B       R          100
                   700
        W          400
                   600
                  1000
NL      R          300
                   500
                   900
        W          200
                   800
```

How do ORDER and GROUP variables differ?

	ORDER	GROUP
Rows are ordered?	YES	YES
Repetitious printing of values is suppressed?	YES	YES
Rows with same values are collapsed?	NO	YES
Type of report produced?	DETAIL	SUMMARY

Defining a GROUP Variable

EXAMPLE

Use the REPORT.MAY1998 data set to generate a summary report.

REPORT.MAY1998 (obs = 5)

CLNT_ID	S_CTRY	S_CITY	S_TYPE	P_GRP	P_SUB	…
B-A-000674	B	B-A	R	Beverages	Soft Drinks	…
B-A-000674	B	B-A	R	Beverages	Soft Drinks	…
B-A-000674	B	B-A	R	Beverages	Beers	…
B-A-000674	B	B-A	R	Vegetables	Deep-freeze Vegetables	…
B-A-000674	B	B-A	R	Vegetables	Canned Vegetables	…

(continued)

CLNT_ID	…	P_ID	P_TYPE	QUANTITY	SALES
B-A-000674	…	Coca-Cola	Other Brand	24	1152
B-A-000674	…	Fanta	Other Brand	2	90
B-A-000674	…	Stella Artois	Other Brand	1	278
B-A-000674	…	Soup Vegetables	Own Brand	1	65
B-A-000674	…	Tomatoes	Own Brand	12	180

Steps

1. Use PROC FORMAT to create a custom format named $CITYFMT for the variable S_CITY. Permanently store the format in the REPORT library in the default FORMATS catalog.

```
proc format lib = report;
   value $ctryfmt "B"  = "Belgium"
                  "NL" = "The Netherlands";
run;
```

2. From the PROGRAM EDITOR window, invoke PROC REPORT with prompting to produce a summary report of the REPORT.MAY1998 SAS data set. Subset the report by only selecting wholesale trade sales figures. Specify appropriate subtitles for the report.

```
title4 "Wholesale Trade Sales Figures (May 1998)";
title5 "---------------------------------------";
title6 " ";

proc report data = report.may1998 prompt;
   where s_type = "W";
run;
```

3. Use the DATA COLUMNS window to select the variables to include in the report. The order in which you select the variables determines their order in your initial report. Select these variables in this order:

- Country
- City
- Sales (BEF)

Select **File - Accept Selection** to close the DATA COLUMNS window. The initial report is displayed in the REPORT window and the DEFINITION window is opened.

4. Make appropriate selections in prompting mode to define the variables S_CTRY, S_CITY and SALES as follows:

VARIABLE	DEFINITION	VALUE
S_CTRY	**Usage**	GROUP
	Attributes	
	Format =	$CTRYFMT.
	Width =	16
	Header =	Shop/Country
S_CITY	**Usage**	GROUP
	Attributes	
	Format =	$CITYFMT.
	Width =	9
	Header =	Shop/City
SALES	**Usage**	ANALYSIS
	Attributes	
	Format =	COMMAX10.
	Width =	10
	Header =	Sales/(BEF)
	Statistic	SUM

5. Open the ROPTIONS window by selecting **Tools - Options - Report ...** Use the HEADLINE and HEADSKIP report options to add a line under the column headings and skip a line between the headings and the text of the report.

PROC REPORT Output

```
                        SOLID Stores
                        ============

                Wholesale Trade Sales Figures (May 1998)
                ----------------------------------------


                Shop              Shop            Sales
                Country           City            (BEF)
                ffffffffffffffffffffffffffffffffffffff

                Belgium           Brussels     2.898.328
                                  Liège        2.806.218
                The Netherlands   Amsterdam    4.091.511
                                  Rotterdam    2.472.215
```

Open the SOURCE window to display the PROC REPORT language statements.

```
proc report data = report.may1998 ls = 96 ps = 54 split = "/"
            headline headskip center;
   column s_ctry s_city sales;
   define s_ctry / group format = $ctryfmt. width = 16
                   spacing = 2 left "Shop/Country";
   define s_city / group format = $cityfmt. width = 9
                   spacing = 2 left "Shop/City";
   define sales / sum format = commax10. width = 10
                  spacing = 2 right "Sales/(BEF)";
run;
```

Store the report definition in the following REPT catalog entry:

FIELD	VALUE
LIBNAME	REPORT
CATALOG	REP_DEF
REPORT NAME	SUMMARY1
DESCRIPTION	Simple Summary Report

Breaking on GROUP Variables

EXAMPLE

Further summarize the previously generated report by inserting break lines containing a grand total as well as subtotals after each shop country.

Steps

1. Display a grand total at the bottom of the previously generated report. Select **Edit - Summarize Information - At the Bottom**. The BREAK window opens. Within the BREAK window, select the following options:

   - Double overline summary
   - Double underline summary
   - Summarize analysis columns

2. Also include subtotals for each shop country. First select the variable **Shop Country** and then select **Edit - Summarize Information - After Item**. The BREAK window opens. Within the BREAK window, select the following options:

   - Overline summary
   - Skip line after break
   - Summarize analysis columns
   - Suppress break value

PROC REPORT Output

```
                    SOLID Stores
                    ============

              Wholesale Trade Sales Figures (May 1998)
              ----------------------------------------


              Shop              Shop             Sales
              Country           City             (BEF)
              ffffffffffffffffffffffffffffffffffffff

              Belgium           Brussels      2.898.328
                                Liège         2.806.218
                                             fffffffffff
                                              5.704.546

              The Netherlands   Amsterdam     4.091.511
                                Rotterdam     2.472.215
                                             fffffffffff
                                              6.563.726

                                            ===========
                                            12.268.271
                                            ===========
```

Open the SOURCE window to display the PROC REPORT language statements.

```
break after s_ctry / ol skip summarize suppress;
rbreak after / dol dul summarize;
```

Store the report definition in the following REPT catalog entry:

FIELD	VALUE
LIBNAME	REPORT
CATALOG	REP_DEF
REPORT NAME	SUMMARY2
DESCRIPTION	Summary Report with Totals and Subtotals

Creating Summarized Output Data

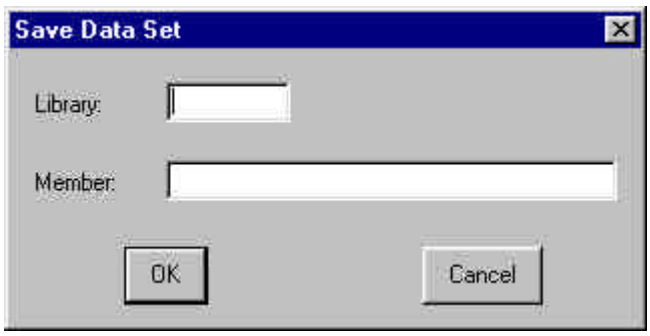
You can create a new SAS data set from the data shown on a report. You can use the new SAS data set

- to create another report in the REPORT window
- as input to another SAS tool
- as output to another software package or database.

EXAMPLE

Save the previously generated report as a SAS data set. Select **File - Save Data Set ...** to open the SAVE DATA SET window. Type a library of REPORT and a member of SUM_MAY1998.

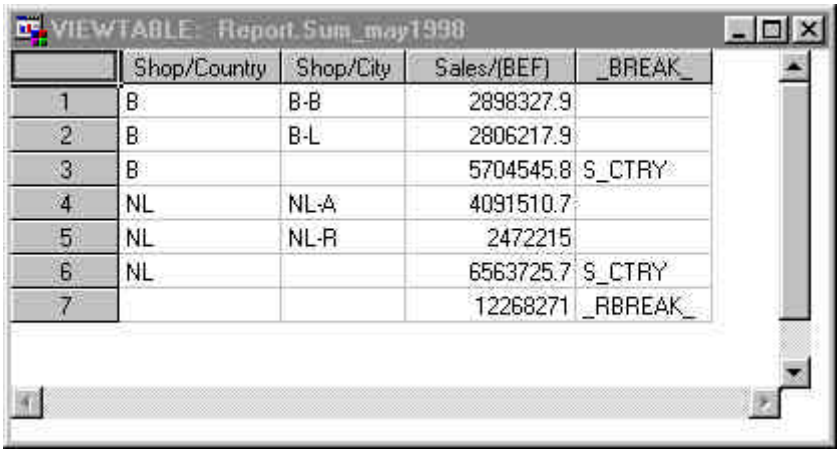
SAVE DATA SET Window



A note appears confirming that the new SAS data set was created.

You can view the new SAS data set by opening the VIEWTABLE window.

VIEWTABLE Window



	Shop/Country	Shop/City	Sales/(BEF)	_BREAK_
1	B	B-B	2898327.9	
2	B	B-L	2806217.9	
3	B		5704545.8	S_CTRY
4	NL	NL-A	4091510.7	
5	NL	NL-R	2472215	
6	NL		6563725.7	S_CTRY
7			12268271	_RBREAK_

Notice that the SAS data set REPORT.SUM_MAY1998 contains all display variables as well as a variable _BREAK_. This _BREAK_ variable contains all BREAK and RBREAK information. In addition to these variables, any non-display variables (NOPRINT) would be included if they existed.

Calculating Percentages

You can use an alias to produce a report that presents one variable in two different ways, in this case, both the sales figure in BEF and the sales figure expressed as a percentage.

You assign an alias in the COLUMN statement by replacing the name of an item with an alias.

SYNTAX

```
item-name = alias
```

item-name is the name of the item.

alias is the alias used in the DEFINE statement.

An alias can only be used in the DEFINE statement.

You can use an alias to present one variable in two different ways:

- Make an entry for each use of the variable in the COLUMN statement, specifying an alias for the variable you want to present in two different ways.
- Write a DEFINE statement for each alias, specifying a different definition for each one.

EXAMPLE

Add a percentage column to the previously generated report.

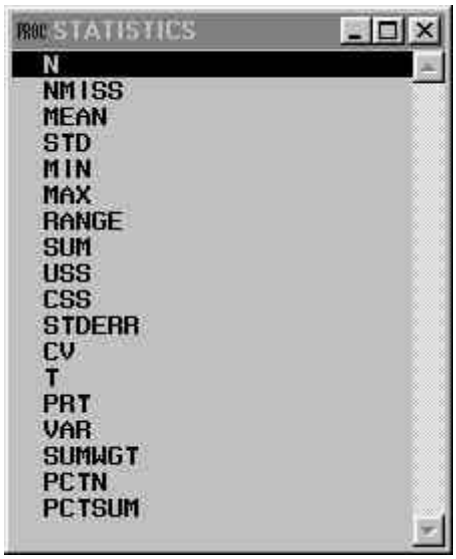
Steps

1. First select the variable **Sales (BEF)** and then select **Edit - Add Item - Data Column - Right**. The DATA COLUMNS window opens.
2. In the DATA COLUMNS window, select the variable **Sales (BEF)** to include in the report. Select **File - Accept Selection** to close the DATA COLUMNS window. The new column is added as the right-most column in the report.
3. In the DEFINITION window, make the following selections for the new column:

VARIABLE	DEFINITION	VALUE
SALES	**Usage**	ANALYSIS
	Attributes	
	Format =	PERCENT6.
	Width =	6
	Statistic =	PCTSUM
	Alias =	SALESPCT
	Header =	Sales/(%)

You could also type **?** in the **Statistic =** field to open the STATISTICS window. In the STATISTICS window, select PCTSUM from the list of available statistics.

STATISTICS Window



PROC REPORT Output

SOLID Stores

Wholesale Trade Sales Figures (May 1998)

Shop Country	Shop City	Sales (BEF)	Sales (%)
Belgium	Brussels	2.898.328	24%
	Liège	2.806.218	23%
		5.704.546	46%
The Netherlands	Amsterdam	4.091.511	33%
	Rotterdam	2.472.215	20%
		6.563.726	54%
		12.268.271	100%

Open the SOURCE window to display the PROC REPORT language statements.

```
column s_ctry s_city sales sales = salespct;
…
define salespct / pctsum format = percent6. width = 6 spacing = 2
                  right "Sales/(%)";
```

Store the report definition in the following REPT catalog entry:

FIELD	VALUE
LIBNAME	REPORT
CATALOG	REP_DEF
REPORT NAME	SUMMARY3
DESCRIPTION	Summary Report with Percentages

Traffic Lighting

The CALL DEFINE Statement

The CALL DEFINE statement sets the value of an attribute for a particular column in the current row. The statement is valid only in the COMPUTE window.

SYNTAX

```
CALL DEFINE (column-id, 'attribute-name', value);
```

column-id specifies a column name or a column number. A column ID can be one of the following:

- a character literal (in quotation marks)
- a character expression
- a numeric literal
- a numeric expression
- a name of the form _C*n*_, where *n* is the column number
- the automatic variable _COL_. This variable identifies the column containing the report item that the COMPUTE block is attached to.

attribute-name is the attribute to define.

value sets the value for the attribute.

Selected attributes include:

ATTRIBUTE	DESCRIPTION
BLINK	Controls blinking of current value.
COLOR	Controls the color of the current value in the REPORT window.
COMMAND	Specifies that a series of commands follows.
FORMAT	Specifies a format for the column.
HIGHLIGHT	Controls highlighting of the current value.
RVSVIDEO	Controls display of the current value.

Selected values for the attributes include:

ATTRIBUTE	VALUE
BLINK	1 = blinking on 0 = blinking off
COLOR	'BLUE' 'RED' 'PINK' 'GREEN' 'CYAN' 'YELLOW' 'WHITE' 'ORANGE' 'BLACK' 'MAGENTA' 'GRAY' 'BROWN'
COMMAND	a quoted string of SAS commands to submit to the command line
FORMAT	a SAS format or a user-defined format
HIGHLIGHT	1 = highlighting on 0 = highlighting off
RVSVIDEO	1 = reverse video on 0 = reverse video off

The attributes BLINK, HIGHLIGHT, and RVSVIDEO do not work on all devices.

EXAMPLE

Use traffic lighting for the percentage column in the previously generated report.

Steps

1. First select the variable **Sales (%)** and then select **Edit - Define ...**. The DEFINITION window opens.
2. In the DEFINITION window, select **Edit Program** to open the COMPUTE window. Enter the following SAS statements in this window to define traffic lighting:

```
if 0.1 < sales.pctsum <= 0.225
   then call define (_col_, "color", "red");
else if 0.225 < sales.pctsum <= 0.275
   then call define (_col_, "color", "blue");
else if 0.275 < sales.pctsum <= 0.4
   then call define (_col_, "color", "green");
```

PROC REPORT Output

SOLID Stores

Wholesale Trade Sales Figures (May 1998)

Shop Country	Shop City	Sales (BEF)	Sales (%)
Belgium	Brussels	2.898.328	24%
	Liège	2.806.218	23%
		5.704.546	46%
The Netherlands	Amsterdam	4.091.511	33%
	Rotterdam	2.472.215	20%
		6.563.726	54%
		12.268.271	100%

Open the SOURCE window to display the PROC REPORT language statements.

```
compute salespct;
   if 0.1 < sales.pctsum <= 0.225
      then call define (_col_, "color", "red");
   else if 0.225 < sales.pctsum <= 0.275
      then call define (_col_, "color", "blue");
   else if 0.275 < sales.pctsum <= 0.4
      then call define (_col_, "color", "green");
endcomp;
```

Store the report definition in the following REPT catalog entry:

FIELD	VALUE
LIBNAME	REPORT
CATALOG	REP_DEF
REPORT NAME	SUMMARY4
DESCRIPTION	Summary Report with Traffic Lighting

Customizing Break Lines

The LINE Statement

LINE statements

- are used in the COMPUTE window to customize the information presented at break points
- use a subset of the syntax of the PUT statement.

SYNTAX

```
LINE specification …;
```

LINE statement specifications can contain

- variable names
- compound names of analysis variables
- formats
- character literals
- *@n* and *+n* pointer controls.

The $VARYING. Format

You can use the $VARYING. format to write a character value with a length that differs from observation to observation. The $VARYING. format must be followed by a variable containing the length of the individual value.

SYNTAX

```
$VARYING length-variable
```

length-variable names the variable that defines the length of the current value.

The PUT Function

You can use the PUT function to write the value of a variable using a particular SAS or user-defined format.

SYNTAX

```
PUT (variable, format)
```

The LENGTH Function

The LENGTH function returns the length of each individual value.

SYNTAX

```
LENGTH (variable)
```

EXAMPLE

Use the REPORT.MAY1998 data set to generate a summary report with customized break lines.

REPORT.MAY1998 (obs = 5)

CLNT_ID	S_CTRY	S_CITY	S_TYPE	P_GRP	P_SUB	…
B-A-000674	B	B-A	R	Beverages	Soft Drinks	…
B-A-000674	B	B-A	R	Beverages	Soft Drinks	…
B-A-000674	B	B-A	R	Beverages	Beers	…
B-A-000674	B	B-A	R	Vegetables	Deep-freeze Vegetables	…
B-A-000674	B	B-A	R	Vegetables	Canned Vegetables	…

(continued)

CLNT_ID	…	P_ID	P_TYPE	QUANTITY	SALES
B-A-000674	…	Coca-Cola	Other Brand	24	1152
B-A-000674	…	Fanta	Other Brand	2	90
B-A-000674	…	Stella Artois	Other Brand	1	278
B-A-000674	…	Soup Vegetables	Own Brand	1	65
B-A-000674	…	Tomatoes	Own Brand	12	180

Steps

1. From the PROGRAM EDITOR window, invoke PROC REPORT with prompting to produce a summary report of the REPORT.MAY1998 SAS data set. Subset the report by only selecting wholesale trade sales figures.

```
proc report data = report.may1998 prompt;
   where s_type = "W";
run;
```

2. Use the DATA COLUMNS window to select the variables to include in the report. The order in which you select the variables determines their order in your initial report. Select these variables in this order:

- City
- Product Group
- Sales (BEF)

Select **File - Accept Selection** to close the DATA COLUMNS window. The initial report is displayed in the REPORT window and the DEFINITION window is opened.

3. Make appropriate selections in prompting mode to define the variables S_CITY, P_GRP and SALES as follows:

VARIABLE	DEFINITION	VALUE
S_CITY	**Usage**	GROUP
P_GRP	**Usage**	GROUP
SALES	**Attributes** Format = Width =	 COMMAX10. 12

3. Display customized information at the top of the report. Select **Edit - Summarize Information - At the Top**. The BREAK window opens. Within the BREAK window, select RED from the Color list. Then select **Edit Program** to open the COMPUTE window. Enter the following SAS statements in this window:

```
line " ";
line @20 63 * "=";
line @20 "The total sales figure for all cities is "
         sales.sum commax10. " BEF.";
line @20 63 * "=";
```

4. Also include customized information before each shop city. First select the variable **City** and then select **Edit - Summarize Information - Before Item**. The BREAK window opens. Within the BREAK window, select GREEN from the Color list. Then select **Edit Program** to open the COMPUTE window. Enter the following SAS statements in this window:

```
fmt_city = put (s_city, $cityfmt.);
len_city = length (fmt_city);
line " ";
line " ";
line @20 63 * "-";
line @20 "The total sales figure for the city "
         fmt_city $varying. len_city
         " is " sales.sum commax9. " BEF.";
line @20 "Breakdown by product group is as follows:";
line @20 63 * "-";
line " ";
```

4. There is no need any more to display the City column. To hide the City column, first select the variable **City** and then select **Edit - Define ...** In the DEFINITION window, select NOPRINT in the Options list.

Partial PROC REPORT Output

```
                          SOLID Stores
                          ============

                 Wholesale Trade Sales Figures (May 1998)
                 ----------------------------------------


                 Product Group          Sales (BEF)

================================================================
The total sales figure for all cities is 12.268.271 BEF.
================================================================


----------------------------------------------------------------
The total sales figure for the city Brussels is 2.898.328 BEF.
Breakdown by product group is as follows:
----------------------------------------------------------------

                 Bakery                    41.910
                 Beverages              1.149.653
                 Butchery                 691.399
                 Dairy Products           593.116
                 Fruits                   129.670
                 Groceries                149.143
                 Vegetables               143.437

                                 ...

----------------------------------------------------------------
The total sales figure for the city Rotterdam is 2.472.215 BEF.
Breakdown by product group is as follows:
----------------------------------------------------------------

                 Bakery                    39.871
                 Beverages                954.290
                 Butchery                 582.786
                 Dairy Products           522.122
                 Fruits                   104.394
                 Groceries                135.811
                 Vegetables               132.941
```

Open the SOURCE window to display the PROC REPORT language statements.

```
proc report data = report.may1998 ls = 96 ps = 54
            split = "/" center;
   column s_city p_grp sales;
   define s_city / group format = $5. width = 5 spacing = 2
                   noprint left "City";
   define p_grp / group format = $20. width = 20 spacing = 2
                  left "Product Group";
   define sales / sum format = commax10. width = 12 spacing = 2
                  right "Sales (BEF)";

   break before s_city / color = green;
   compute before s_city;
      fmt_city = put (s_city, $cityfmt.);
      len_city = length (fmt_city);
      line " ";
      line " ";
      line @20 63 * "-";
      line @20 "The total sales figure for the city "
               fmt_city $varying. len_city
               " is " sales.sum commax9. " BEF.";
      line @20 "Breakdown by product group is as follows:";
      line @20 63 * "-";
      line " ";
   endcomp;

   rbreak before / color = red;
   compute before;
      line " ";
      line @20 63 * "=";
      line @20 "The total sales figure for all cities is"
               sales.sum commax10. " BEF.";
      line @20 63 * "=";
   endcomp;
run;
```

Store the report definition in the following REPT catalog entry:

FIELD	VALUE
LIBNAME	REPORT
CATALOG	REP_DEF
REPORT NAME	SUMMARY5
DESCRIPTION	Summary Report with Customized Breaks

Advanced Report Writing Techniques

Creating Cross-tabular Reports

Defining an ACROSS Variable

When you define an ACROSS variable, the REPORT procedure

- creates a column for each value of the ACROSS variable
- automatically computes frequency counts for each column.

EXAMPLE

Use the input SAS data set WORK.SAMPLE to illustrate the usage of ACROSS variables.

WORK.SAMPLE

S_CTRY	S_TYPE	SALES
B	R	100
NL	W	200
NL	R	300
B	W	400
NL	R	500
B	W	600
B	R	700
NL	W	800
NL	R	900
B	W	1000

Define S_CTRY as a DISPLAY variable.

```
column s_ctry;
define s_ctry / display;
```

PROC REPORT Output

```
S_CTRY
ffffff

B
NL
NL
B
NL
B
B
NL
NL
B
```

Define S_CTRY as an ACROSS variable.

```
column s_ctry;
define s_ctry / across;
```

PROC REPORT Output

```
S_CTRY

B    NL
ffffff

 5    5
```

EXAMPLE

Use the REPORT.CLIENT data set to generate a cross-tabular report that displays the distribution of males and females within the different languages the clients speak.

REPORT.CLIENT (obs = 10)

CLNT_ID	LNAME	FNAME	LANGUAGE	…	COUNTRY	SEX	…
B-N-384240	Duchêne	Bernardine	F	…	B	F	…
NL-R-770239	Pappaert	Jan	D	…	NL	M	…
NL-DH-734677	Didden	Jos	D	…	NL	M	…
NL-R-930610	Bosch	Valerie	D	…	NL	F	…
NL-R-930841	Bruggeman	Nellie	D	…	NL	F	…
NL-R-931208	Fauconnier	Femke	D	…	NL	F	…
B-N-407468	Berlier	Jean-Marie	F	…	B	M	…
B-N-408019	Charon	Hubert	F	…	B	M	…
NL-A-598678	Agneessens	Frank	D	…	NL	M	…
NL-A-598818	Melis	Guy	D	…	NL	M	…

Steps

1. Create a temporary SAS data set named CLIENT, which contains only the variables LANGUAGE and SEX. Also add the LENGTH statement immediately after the DATA statement to increase the number of bytes used for storing values of the variable LANGUAGE to 5. This longer length is necessary to be able to label the summary row appropriately in a later step.

```
data client;
   length language $ 5;
   set report.client (keep = language sex);
run;
```

2. Use PROC FORMAT to create a custom format named $LANGFMT for the variable LANGUAGE and a custom format named $SEXFMT for the variable SEX. Permanently store the formats in the REPORT library in the default FORMATS catalog.

```
proc format lib = report;
   value $langfmt "D" = "Dutch"
                  "E" = "English"
                  "F" = "French";
   value $sexfmt "F" = "Female"
                 "M" = "Male";
run
```

3. From the PROGRAM EDITOR window, invoke PROC REPORT with prompting to produce a cross-tabular report of the temporary CLIENT SAS data set. Specify appropriate subtitles for the report.

```
title4 "Client Distribution";
title5 "--------------------";
title6 " ";

proc report data = client prompt;
run;
```

4. Use the DATA COLUMNS window to select the variables to include in the report. The order in which you select the variables determines their order in your initial report. Select these variables in this order:

- Mother Tongue
- Gender

Select **File - Accept Selection** to close the DATA COLUMNS window. The initial report is displayed in the REPORT window and the DEFINITION window is opened.

5. Make appropriate selections in prompting mode to define the variables LANGUAGE and SEX as follows:

VARIABLE	DEFINITION	VALUE
LANGUAGE	**Usage** **Attributes** Format = Width =	GROUP $LANGFMT. 13
SEX	**Usage** **Attributes** Format =	ACROSS $SEXFMT.

6. Further enhance the report by adding a summary row. Select **Edit - Summarize Information - At the Bottom**. The BREAK window opens. Within the BREAK window, select **Summarize analysis columns** from the Options list.

7. Label the summary row with descriptive text. Assign the value TOTAL to the variable LANGUAGE for the break line. The length of the label cannot exceed the variable length. Select **Edit Program** to open the COMPUTE window. Enter the following SAS statement in this window:

```
language = "TOTAL";
```

8. Also add a column to the report showing the total number of clients speaking each language. First select the variable **Gender** and then select **Edit - Add Item - Statistic - Right**. The STATISTICS window opens. In the STATISTICS window, select N from the list of available statistics.

9. First select the column **N** and then select **Edit - Define ...** The DEFINITION window opens. Enhance the appearance of the variable N as follows:

VARIABLE	DEFINITION	VALUE
N	**Attributes**	
	Width =	5
	Header =	TOTAL

10. Open the ROPTIONS window by selecting **Tools - Options - Report ...** Use the BOX report option to box in the report.

PROC REPORT Output

SOLID Stores
============

Client Distribution

Mother Tongue	Gender Female	Male	TOTAL
Dutch	866	564	1430
English	27	33	60
French	343	167	510
TOTAL	1236	764	2000

Open the SOURCE window to display the PROC REPORT language statements.

```
proc report data = client ls = 96 ps = 54 split = "/" center box;
   column language sex n;
   define language / group format = $langfmt. width = 13
                     spacing = 2 left "Mother Tongue";
   define sex / across format = $sexfmt. width = 6 spacing = 2
                left "Gender";
   define n / width = 5 spacing = 2 right "TOTAL";

   rbreak after / summarize;
   compute after;
      language = "TOTAL";
   endcomp;
run;
```

You can use the name of a statistic such as N in your COLUMN statement to create a column containing that statistic for each row in your report. You can then use a DEFINE statement to specify the attributes of the column.

Store the report definition in the following REPT catalog entry:

FIELD	VALUE
LIBNAME	REPORT
CATALOG	REP_DEF
REPORT NAME	ADVANCED1
DESCRIPTION	Cross-tabular Report with Frequencies

EXAMPLE

Use the REPORT.MAYMISS data set to generate a cross-tabular report that displays the total sales figures for wholesale trade shops. The SAS data set REPORT.MAYMISS contains no data for the product groups "Butchery" and "Dairy Products" in the Belgian shops.

REPORT.MAYMISS (obs = 5)

CLNT_ID	S_CTRY	S_CITY	S_TYPE	P_GRP	P_SUB	…
B-A-000674	B	B-A	R	Beverages	Soft Drinks	…
B-A-000674	B	B-A	R	Beverages	Soft Drinks	…
B-A-000674	B	B-A	R	Beverages	Beers	…
B-A-000674	B	B-A	R	Vegetables	Deep-freeze Vegetables	…
B-A-000674	B	B-A	R	Vegetables	Canned Vegetables	…

(continued)

CLNT_ID	…	P_ID	P_TYPE	QUANTITY	SALES
B-A-000674	…	Coca-Cola	Other Brand	24	1152
B-A-000674	…	Fanta	Other Brand	2	90
B-A-000674	…	Stella Artois	Other Brand	1	278
B-A-000674	…	Soup Vegetables	Own Brand	1	65
B-A-000674	…	Tomatoes	Own Brand	12	180

Steps

1. From the PROGRAM EDITOR window, invoke PROC REPORT with prompting to produce a cross-tabular report of the REPORT.MAYMISS SAS data set. Subset the report by only selecting wholesale trade sales figures. Use the MISSING = system option to specify that all missing numeric values should be displayed as zeroes (0). Also specify appropriate subtitles for the report.

```
options missing = "0";

title4 "Wholesale Trade Sales Figures (May 1998)";
title5 "---------------------------------------";
title6 " ";

proc report data = report.maymiss prompt;
   where s_type = "W";
run;
```

2. Use the DATA COLUMNS window to select the variables to include in the report. The order in which you select the variables determines their order in your initial report. Select these variables in this order:

- Product Group
- City

Select **File - Accept Selection** to close the DATA COLUMNS window. The initial report is displayed in the REPORT window and the DEFINITION window is opened.

3. Make appropriate selections in prompting mode to define the variables P_GRP and S_CITY as follows:

VARIABLE	DEFINITION	VALUE
P_GRP	**Usage**	GROUP
	Attributes	
	Width =	15
	Header =	(blank)
S_CITY	**Usage**	ACROSS
	Attributes	
	Format =	$CITYFMT.
	Width =	9
	Header =	(blank)

4. Open the ROPTIONS window by selecting **Tools - Options - Report ...** Use the HEADLINE and HEADSKIP report options to add a line under the column headings and skip a line between the headings and the text of the report.

5. Add a summary row at the bottom of the report. Select **Edit - Summarize Information - At the Bottom**. The BREAK window opens. Within the BREAK window, select the following options:

- Overline summary
- Underline summary
- Summarize analysis columns
- Color = GREEN

6. Label the summary row with descriptive text. Assign the value TOTAL to the variable P_GRP for the break line. Select **Edit Program** to open the COMPUTE window. Enter the following SAS statement in this window:

```
p_grp = "TOTAL";
```

7. Instead of frequencies, we would like to display total sales figures in the report. First select the variable **S_CITY** and then select **Edit - Add Item - Data Column - Below**. The DATA COLUMNS window opens. In the DATA COLUMNS window, select the variable **Sales (BEF)** to include in the report. Select **File - Accept Selection** to close the DATA COLUMNS window. The report is displayed in the REPORT window and the DEFINITION window is opened.

8. Make appropriate selections in prompting mode to define the variable SALES as follows:

VARIABLE	DEFINITION	VALUE
SALES	**Usage**	ANALYSIS
	Statistic	SUM
	Attributes	
	Format =	COMMAX10.
	Width =	10
	Header =	(blank)

PROC REPORT Output

SOLID Stores
============

Wholesale Trade Sales Figures (May 1998)

	Amsterdam	Brussels	Liège	Rotterdam
Bakery	56.102	41.910	38.769	39.871
Beverages	1.561.140	1.149.653	1.125.454	954.290
Butchery	1.052.190	0	0	582.786
Dairy Products	835.397	0	0	522.122
Fruits	184.277	129.670	124.827	104.394
Groceries	197.730	149.143	149.731	135.811
Vegetables	204.675	143.437	134.768	132.941
TOTAL	4.091.511	1.613.813	1.573.549	2.472.215

Open the SOURCE window to display the PROC REPORT language statements.

```
proc report data = report.maymiss ls = 96 ps = 54 split = "/"
            headline headskip center;
   column p_grp s_city, sales;
   define p_grp / group format = $20. width = 15 spacing = 2
                  left " ";
   define s_city / across format = $cityfmt. width = 9 spacing = 2
                   left " ";
   define sales / sum format = commax10. width = 10 spacing = 2
                  right " ";

   rbreak after / ol ul summarize color = green;
   compute after;
      p_grp = "TOTAL";
   endcomp;
run;
```

In the COLUMN statement, you can specify two or more items separated by commas to stack the items on one another. You can use this method to specify different statistics for different columns.

Store the report definition in the following REPT catalog entry:

FIELD	VALUE
LIBNAME	REPORT
CATALOG	REP_DEF
REPORT NAME	ADVANCED2
DESCRIPTION	Cross-tabular Report with Total Sales

Presenting the Same Column in Different Ways

You can use an alias to produce a report that presents the same variable in different ways, in this case, both the code and the full name for each shop country and shop city.

EXAMPLE

Enhance the previously created summary report SUMMARY2 by showing both the code and the full name for each shop country and shop city.

Steps

1. Use the stored report definition REPORT.REP_DEF.SUMMARY2.REPT to recreate the summary report in the REPORT window. The stored report definition does not include the subsetting WHERE clause. So you need to invoke the report with the equivalent WHERE clause to reproduce the same report.

```
proc report data = report.may1998
            report = report.rep_def.summary2;
   where s_type = "W";
run;
```

2. First select the variable **Shop Country** and then select **Edit - Add Item - Data Column - Left**. The DATA COLUMNS window opens. In the DATA COLUMNS window, select the variable **Country** to include in the report. Select **File - Accept Selection** to close the DATA COLUMNS window. The new column is added to the report.

3. Then select the variable **Shop City** and then select **Edit - Add Item - Data Column - Left**. The DATA COLUMNS window opens. In the DATA COLUMNS window, select the variable **City** to include in the report. Select **File - Accept Selection** to close the DATA COLUMNS window. The new column is added to the report.

4. In the DEFINITION window of the first four columns, specify the following attributes:

VARIABLE	DEFINITION	VALUE
S_CTRY (1)	**Attributes** Header =	 Country/Code
S_CTRY (2)	**Attributes** Alias = Header =	 FULLCTRY Country/Name
S_CITY (1)	**Attributes** Header =	 City/Code
S_CITY (2)	**Attributes** Alias = Header =	 FULLCITY City/Name

PROC REPORT Output

```
                          SOLID Stores
                          ============

                 Wholesale Trade Sales Figures (May 1998)
                 ----------------------------------------


Country  Country          City   City           Sales
Code     Name             Code   Name           (BEF)
ffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff

B        Belgium          B-B    Brussels    2.898.328
                          B-L    Liège       2.806.218
                                            fffffffffff
                                             5.704.546

NL       The Netherlands  NL-A   Amsterdam   4.091.511
                          NL-R   Rotterdam   2.472.215
                                            fffffffffff
                                             6.563.726

                                            ===========
                                            12.268.271
                                            ===========
```

Open the SOURCE window to display the PROC REPORT language statements.

```
proc report data = report.may1998 ls = 96 ps = 54 split = "/"
            headline headskip center;
   column s_ctry s_ctry = fullctry s_city s_city = fullcity sales;
   define s_ctry / group format = $2. width = 7
                   spacing = 2 left "Country/Code";
   define fullctry / group format = $ctryfmt. width = 16
                     spacing = 2 left "Country/Name";
   define s_city / group format = $5. width = 5
                   spacing = 2 left "City/Code";
   define fullcity / group format = $cityfmt. width = 9
                     spacing = 2 left "City/Name";
   define sales / sum format = commax10. width = 10
                  spacing = 2 right "Sales/(BEF)";

   break after s_ctry / ol skip summarize suppress;
   rbreak after / dol dul summarize;
run;
```

Store the report definition in the following REPT catalog entry:

FIELD	VALUE
LIBNAME	REPORT
CATALOG	REP_DEF
REPORT NAME	ADVANCED3
DESCRIPTION	Same Column Presented in Different Ways

Complex Grouping

You can combine grouping items with item aliases to create more complex cross-tabular reports.

EXAMPLE

Use the REPORT.SEP1998 data set to generate a cross-tabular report that displays the total number of client visits, the total sales figure and the average sales figure for each purchase date for each shop country.

REPORT.SEP1998 (obs = 10)

S_CTRY	S_TYPE	CLNT_ID	S_CITY	DATE	SALES
B	R	B-A-000820	B-A	01SEP1998	110.80
B	R	B-A-004274	B-A	01SEP1998	1910.40
B	R	B-A-008564	B-A	01SEP1998	851.70
B	W	B-A-011578	B-L	01SEP1998	2241.00
B	R	B-A-021041	B-A	01SEP1998	49.00
B	R	B-A-023359	B-A	01SEP1998	917.40
B	R	B-A-023966	B-A	01SEP1998	127.50
B	R	B-A-030429	B-A	01SEP1998	886.10
B	R	B-A-037526	B-A	01SEP1998	2183.35
B	R	B-A-039188	B-A	01SEP1998	464.25

Steps

1. Create a temporary SAS data set named SEP1998, which contains only the variables DATE, S_CTRY and SALES. Use the PUT function to convert the numeric variable DATE into a character one. This numeric-to-character conversion is necessary to be able to label the summary row in a later step.

```
data sep1998 (drop = num);
   set report.sep1998 (keep = date s_ctry sales
                       rename = (date = num));
   date = put (num, ddmmyy10.);
   label date = "Purchase Date";
run;
```

2. From the PROGRAM EDITOR window, invoke PROC REPORT with prompting to produce a cross-tabular report of the temporary SEP1998 SAS data set. Specify appropriate subtitles for the report.

```
title4 "Sales Figures (September 1998)";
title5 "-----------------------------";
title6 " ";

proc report data = sep1998 prompt;
run;
```

3. Use the DATA COLUMNS window to select the variables to include in the report. The order in which you select the variables determines their order in your initial report. Select these variables in this order:

- Purchase Date
- Country

Select **File - Accept Selection** to close the DATA COLUMNS window. The initial report is displayed in the REPORT window and the DEFINITION window is opened.

4. Make appropriate selections in prompting mode to define the variables DATE and S_CTRY as follows:

VARIABLE	DEFINITION	VALUE
DATE	**Usage**	GROUP
	Attributes	
	Header =	Purchase/Date
S_CTRY	**Usage**	ACROSS
	Attributes	
	Format =	$CTRYFMT.
	Width =	15

5. Add a column to the report showing the total number of client visits for each purchase date for each shop country. First select the variable **Country** and then select **Edit - Add Item - Statistic - Below**. The STATISTICS window opens. In the STATISTICS window, select N from the list of available statistics. Select **File - Accept Selection** to close the STATISTICS window. The report is displayed in the REPORT window and the DEFINITION window is opened.

6. Enhance the appearance of the variable N as follows:

VARIABLE	DEFINITION	VALUE
N	**Attributes**	
	Format =	COMMAX6.
	Width =	6
	Header =	/Client/Count

The extra / in the header for N creates a blank line between the shop country names and the next level of column heading.

7. Add another column to the report showing the total sales figure for each purchase date for each shop country. First select the variable **Client Count** and then select **Edit - Add Item - Data Column - Right**. The DATA COLUMNS window opens. In the DATA COLUMNS window, select the variable **Sales (BEF)** to include in the report. Select **File - Accept Selection** to close the DATA COLUMNS window. The report is displayed in the REPORT window and the DEFINITION window is opened.

8. Enhance the appearance of the variable SALES as follows:

VARIABLE	DEFINITION	VALUE
SALES (1)	**Attributes**	
	Format =	COMMAX9.
	Width =	9
	Header =	Total/Sales

9. Add another column to the report showing the average sales figure for each purchase date for each shop country. First select the variable **Total Sales** and then select **Edit - Add Item - Data Column - Right**. The DATA COLUMNS window opens. In the DATA COLUMNS window, select the variable **Sales (BEF)** to include in the report. Select **File - Accept Selection** to close the DATA COLUMNS window. The report is displayed in the REPORT window and the DEFINITION window is opened.

10. Enhance the appearance of the variable SALES as follows:

VARIABLE	DEFINITION	VALUE
SALES (2)	**Statistic**	MEAN
	Attributes	
	Format =	COMMAX7.
	Width =	7
	Header =	Average/Sales

11. In the DEFINITION window of the following columns, specify these additional attributes:

VARIABLE	DEFINITION	VALUE
DATE	**Attributes**	
	Justify =	CENTER
N	**Attributes**	
	Spacing =	5
SALES (2)	**Attributes**	
	Alias =	AVGSALES

12. Open the ROPTIONS window by selecting **Tools - Options - Report ...** Use the HEADLINE and HEADSKIP report options to add a line under the column headings and skip a line between the headings and the text of the report.

13. Add a summary row at the bottom of the report. Select **Edit - Summarize Information - At the Bottom**. The BREAK window opens. Within the BREAK window, select the following options:

- Overline summary
- Underline summary
- Summarize analysis columns
- Color = GREEN

14. Label the summary row with descriptive text. Assign the value TOTAL to the variable DATE for the break line. Select **Edit Program** to open the COMPUTE window. Enter the following SAS statement in this window:

```
date = "TOTAL";
```

PROC REPORT Output

SOLID Stores

Sales Figures (September 1998)

Purchase Date	Country: Belgium Client Count	Belgium Total Sales	Belgium Average Sales	The Netherlands Client Count	The Netherlands Total Sales	The Netherlands Average Sales
01/09/1998	88	161.961	1.840	97	185.561	1.913
02/09/1998	105	172.415	1.642	86	173.399	2.016
03/09/1998	108	165.697	1.534	77	148.658	1.931
04/09/1998	218	359.724	1.650	138	319.299	2.314
05/09/1998	254	423.830	1.669	198	400.597	2.023
07/09/1998	238	416.000	1.748	159	338.750	2.131
08/09/1998	101	166.061	1.644	72	148.445	2.062
09/09/1998	111	165.983	1.495	86	172.464	2.005
10/09/1998	106	181.396	1.711	90	200.230	2.225
11/09/1998	212	369.199	1.742	167	371.604	2.225
12/09/1998	300	556.453	1.855	220	449.478	2.043
14/09/1998	227	371.993	1.639	152	295.402	1.943
15/09/1998	113	152.778	1.352	77	174.811	2.270
16/09/1998	104	167.445	1.610	83	212.580	2.561
17/09/1998	105	171.752	1.636	82	162.900	1.987
18/09/1998	222	364.288	1.641	155	384.701	2.482
19/09/1998	264	413.175	1.565	208	428.423	2.060
21/09/1998	209	347.507	1.663	156	342.120	2.193
22/09/1998	102	136.278	1.336	71	174.478	2.457
23/09/1998	95	122.633	1.291	95	166.731	1.755
24/09/1998	109	191.127	1.753	78	157.921	2.025
25/09/1998	204	317.113	1.554	165	326.868	1.981
26/09/1998	298	526.035	1.765	202	452.141	2.238
28/09/1998	217	386.321	1.780	159	336.884	2.119
29/09/1998	103	176.684	1.715	69	121.731	1.764
30/09/1998	100	147.377	1.474	68	158.148	2.326
TOTAL	4.313	7.131.226	1.653	3.210	6.804.322	2.120

Open the SOURCE window to display the PROC REPORT language statements.

```
proc report data = sep1998 ls = 96 ps = 54 split = "/"
            headline headskip center;
   column date s_ctry, (n sales sales = avgsales);
   define date / group format = $10. width = 10
                 spacing = 2 center "Purchase/Date";
   define s_ctry / across format = $ctryfmt. width = 15
                   spacing = 2 left "Country";
   define n / format = commax6. width = 6
              spacing = 5 right "/Client/Count";
   define sales / sum format = commax9. width = 9
                  spacing = 2 right "Total/Sales";
   define avgsales / mean format = commax7. width = 7
                     spacing = 2 right "Average/Sales";

   rbreak after / ol ul summarize color = green;
   compute after;
      date = "TOTAL";
   endcomp;
run;
```

You use parentheses to place several items under one item.

Store the report definition in the following REPT catalog entry:

FIELD	VALUE
LIBNAME	REPORT
CATALOG	REP_DEF
REPORT NAME	ADVANCED4
DESCRIPTION	Complex Grouping

Creating Multi-column Reports

You can develop multi-column reports by using the PANELS = and PSPACE = options in the PROC REPORT statement or the Panels = and Panelspace = fields in the ROPTIONS window.

The PANELS = option or the Panels = field specifies the number of panels on each page of the report. If the width of a report is less than half of the line size, you can display the data in multiple sets of columns so that rows that would otherwise appear on multiple pages appear on the same page. Each set of columns is a panel. A familiar example of this kind of report is a telephone book, which contains multiple panels of names and telephone numbers on a single page. When PROC REPORT writes a multi-panel report, it fills one panel before beginning the next.

The PSPACE = option or the Panelspace = field specifies the number of blank characters between panels. PROC REPORT separates all panels in the report by the same number of blank characters.

EXAMPLE

Use the REPORT.JULY1998 data set to generate a multi-column report.

REPORT.JULY1998 (obs = 10)

S_CTRY	S_CITY	S_TYPE	P_GRP	P_SUB	SALES
B	B-A	R	Beverages	Waters	26136
B	B-A	R	Beverages	Fruit Juices	18159
B	B-A	R	Beverages	Soft Drinks	143016
B	B-A	R	Beverages	Beers	123917
B	B-A	R	Beverages	Wines	10272
B	B-A	R	Beverages	Alcoholic Drinks	46593
B	B-A	R	Butchery	Pork	16530
B	B-A	R	Butchery	Veal	36989
B	B-A	R	Butchery	Lamb	32245
B	B-A	R	Butchery	Chicken	16327

Steps

1. From the PROGRAM EDITOR window, invoke PROC REPORT with prompting to produce a multi-column report of the REPORT.JULY1998 SAS data set. Specify appropriate subtitles for the report.

```
title4 "Sales Figures (July 1998)";
title5 "-------------------------";
title6 " ";

proc report data = report.july1998 prompt;
run;
```

2. Use the DATA COLUMNS window to select the variables to include in the report. The order in which you select the variables determines their order in your initial report. Select these variables in this order:

- City
- Product Group
- Sales (BEF)

Select **File - Accept Selection** to close the DATA COLUMNS window. The initial report is displayed in the REPORT window and the DEFINITION window is opened.

3. Make appropriate selections in prompting mode to define the variables S_CITY, P_GRP and SALES as follows:

VARIABLE	DEFINITION	VALUE
S_CITY	**Usage**	GROUP
	Attributes	
	Format =	$CITYFMT.
	Width =	9
	Header =	Shop City
P_GRP	**Usage**	GROUP
SALES	**Usage**	ANALYSIS
	Statistic	SUM
	Attributes	
	Format =	COMMAX12.
	Width =	12

4. In the DEFINITION window of the column S_CITY, specify this additional attribute:

VARIABLE	DEFINITION	VALUE
S_CITY	**Attributes**	
	Justify =	CENTER

5. Open the ROPTIONS window by selecting **Tools - Options - Report ...** Use the HEADLINE and HEADSKIP report options to add a line under the column headings and skip a line between the headings and the text of the report. Use the **Panels =** field to specify 2 panels on each page of the report. Also use the **Panelspace =** field to specify 5 blank characters between panels.

6. Display a grand total at the bottom of the report. Select **Edit - Summarize Information - At the Bottom**. The BREAK window opens. Within the BREAK window, select the following options:

- Double overline summary
- Double underline summary
- Summarize analysis columns
- Color = RED

7. Label the summary row with descriptive text. Assign the value TOTAL to the variable S_CITY for the break line. Select **Edit Program** to open the COMPUTE window. Enter the following SAS statement in this window:

```
s_city = "TOTAL";
```

8. Also include subtotals for each shop city. First select the variable **Shop City** and then select **Edit - Summarize Information - After Item.** The BREAK window opens. Within the BREAK window, select the following options:

- Overline summary
- Underline summary
- Skip line after break
- Summarize analysis columns
- Color = GREEN

PROC REPORT Output

SOLID Stores
============

Sales Figures (July 1998)

Shop City	Product Group	Sales (BEF)	Shop City	Product Group	Sales (BEF)
Amsterdam	Bakery	53.860	Liège	Bakery	37.607
	Beverages	1.485.503		Beverages	1.106.558
	Butchery	937.549		Butchery	711.982
	Dairy Products	804.432		Dairy Products	526.061
	Fruits	167.218		Fruits	120.863
	Groceries	198.889		Groceries	147.084
	Vegetables	191.182		Vegetables	125.300
Amsterdam		3.838.633	Liège		2.775.455
Antwerp	Bakery	48.404	Namur	Bakery	32.755
	Beverages	368.093		Beverages	245.727
	Butchery	186.501		Butchery	131.896
	Dairy Products	132.460		Dairy Products	88.623
	Fruits	34.736		Fruits	24.756
	Groceries	118.084		Groceries	72.668
	Vegetables	30.428		Vegetables	22.999
Antwerp		918.706	Namur		619.425
Brussels	Bakery	46.692	Rotterdam	Bakery	27.889
	Beverages	1.187.368		Beverages	1.000.380
	Butchery	848.122		Butchery	583.073
	Dairy Products	651.151		Dairy Products	495.548
	Fruits	140.513		Fruits	107.727
	Groceries	163.976		Groceries	121.826
	Vegetables	147.069		Vegetables	119.257
Brussels		3.184.891	Rotterdam		2.455.700
Den Haag	Bakery	32.929			
	Beverages	227.282	TOTAL		14.366.965
	Butchery	110.309			
	Dairy Products	80.345			
	Fruits	22.080			
	Groceries	80.901			
	Vegetables	20.310			
Den Haag		574.155			

Open the SOURCE window to display the PROC REPORT language statements.

```
proc report data = report.july1998 ls = 96 ps = 54 split = "/"
            panels = 2 pspace = 5 headline headskip center;
   column s_city p_grp sales;
   define s_city / group format = $cityfmt. width = 9 spacing = 2
                   center "Shop City";
   define p_grp / group format = $20. width = 15 spacing = 2
                  left "Product Group";
   define sales / sum format = commax12. width = 12 spacing = 2
                  right "Sales (BEF)";

   break after s_city / ol ul skip summarize color = green;
   rbreak after / dol dul summarize color = red;
   compute after;
      s_city = "TOTAL";
   endcomp;
run;
```

Store the report definition in the following REPT catalog entry:

FIELD	VALUE
LIBNAME	REPORT
CATALOG	REP_DEF
REPORT NAME	ADVANCED5
DESCRIPTION	Multi-column Report