P-STAT®

The SURVEY, BALANCE and SAMPLE Commands

P-STAT: The SURVEY, BALANCE and SAMPLE Commands

,

Fifth Edition: January 2013

This publication corresponds to **P-STAT Version 3.01, January 2013**. This publication is designed for those who are already familiar with the P-STAT system to supply complete documentation for the SURVEY, BALANCE and SAMPLE commands.

Please direct any questions to:

P-STAT, Inc.
230 Lambertville-Hopewell Rd.
Hopewell, New Jersey 08525-2809
U.S.A.

Telephone: 609-466-9200

Fax: 609-466-1688

Internet: support@pstat.com

Web Page URL: http://www.pstat.com

CONTENTS

SURVEY: Creating Simple Tables

SURVEY: More About the Rows

;;

SURVEY: More About the Columns

SURVEY: Subtotals and Nets

SURVEY: Special Features of the Layout Sections

SURVEY: Enhancing Table Appearance

SURVEY: PostScript Output

SURVEY:The Statistics

SURVEY: More Statistics

SURVEY: Other Features

TITLES and LABELS

BALANCE and SAMPLE

FIGURES

1
SURVEY:
Creating Simple Tables

This manual is largely devoted to the SURVEY command which is a major tool for market research reports. It also documents the BALANCE and SAMPLE commands. BALANCE is used to calculate appropriate weights so that a sample represents known population weights. The SAMPLE command is used to produce a random subset of a larger file which matches the input file in critical ways.

The SURVEY command displays responses to market studies, questionnaires and any other data that can be presented in a table. The studies typically include background variables such as Age, Sex and Income, and a number of analysis variables. The analysis variables are the items of interest in the study. They are often respondent preferences, opinions and information.

The table output typically displays the background variables in columns across the top against the analysis variables in rows down the page. The terms BANNER for columns and STUB for rows are used because those are terms common in the market research field. However, the SURVEY output is a table, appropriate for analysis of any variables in terms of other variables. The stub (row) variables may be displayed one per page or grouped together, and multiple response variables are possible. A variety of format and layout options permit the survey to be tailored to each user's specifications.

This chapter describes how to:

1. use the SURVEY command to create simple tables
2. control the contents of the table
3. enhance the table with titles and labels
4. print the tables on the printer or store them in a disk file
5. weight the tables and calculate means on other variables
6. build the SURVEY command interactively or for a job to be run in the background

Other chapters describe

1. More about the rows, including multiple response variables
2. More about the columns, and row and column interactions
3. Subtotals and Nets
4. Special features for each section of the table
5. Fine tuning the appearance including PostScript support
6. Statistics and significance tests
7. Other features include handling character data, BY variables and tables of contents

1.1 THE SURVEY COMMAND

The SURVEY command requires one or more P-STAT system files as input. The data values that define the stubs (rows) and banners (columns) are typically numbers, but they may also be character strings 1-24 characters in

length. (See the MAP command in the chapter “Titles, Labels and the MAP command” for information on how to handle longer character strings.)

If a file name is not supplied, the most recently referenced file is used. The SURVEY command supports multiple (1-4) P-STAT system files as input. Multiple (no fixed limit) label files are also supported. Labels are optional. If a labels file is not used or if it does not contain extended variable labels, SURVEY makes use of the variable names. There are controls so that:

1. the full (up to 64 character) variable names can be used
2. the FULL, TAG or TEXT portions of a label can be selectively chosen for banner and/or stub labelling.

If there is a labels file, the variable name in the labels file needs to match either the entire variable name or, if it is available, the TAG. The 64 character variable names do not replace the extended label fields in the labels file which can be much longer and can have a larger character set than that allowed for variable names.

If there are extended variable labels in a labels file, they are used in the printout in place of the variable name. If the variable is a character variable, the values themselves are used to label the rows and columns of the table. If the variable is a numeric variable and has value labels, those labels are used instead of the numbers.

SURVEY has both identifiers and subcommands. Identifiers and their arguments are supplied as part of the SURVEY command – that is, *before* the initial semicolon. Subcommands and their arguments (the table definitions) are supplied at the subcommand level – that is, *after* the initial semicolon:

```
SURVEY  Paceset,   LABELS  'Paceset.lab' ;
     BANNER  Age  SEX,
     STUB    Num.TV ;
$
SURVEY  Paceset;
     SET.LABELS BAN FULL,
     SET.LABELS STUB TAGS,
     BANNER  Age  Sex,  STUB  Num.TV;
$
```

Figure 1.1 shows the first few cases of Paceset, the P-STAT system file used in many of the examples in this manual. The MODIFY command adds two variables. Variable Income.groups is created from the Income variable to provide the information in a form that is more useful for defining the rows or columns of a table. Variable Own is a composite of the 4 variables Phone, Lines, Tablet and Ereader. It is a character variable with the values “yes” or “no” depending on whether the respondent owns any of the four electronic items. Figure 1.1 also shows a SAVE.LABELS command that is appropriate for the data in file Paceset and at the bottom the variable names for Pacenew which has the long variable names supported in P-STAT version 3. The labels file in Figure 1 is appropriate for use with either version 2 or version 3 P-STAT system files. Labels files can, of course, have full 64 character variable names which must match either the tag or the full variable names.

1.2 REQUIRED SUBCOMMANDS

SURVEY requires at least one STUB or one BANNER subcommand to define the rows and columns of the tables. Each of these subcommands is followed by the names of 1 or more variables. The variables may be given as single variable names or groups of adjacent variables. A group of variables is provided by citing the names of the first and last variables in the group separated by the keyword “TO”. The following are valid subcommands:

```
BANNER Age Sex Education ....
BANNER Age  B1 TO B4  Income Region ....
STUB Q1A TO Q53  Q54  Q56 TO Q73  Usage ....
```

There may be any number of STUB subcommands. The chapter “SURVEY: All About the Rows” has a complete discussion of the different types of stub variables. These are used for handling multiple response questions and for controlling the arrangement of the variables on a page.

Figure 1.1 Paceset: Partial Data, a Modification and its Labels

```
 FILE Paceset

     id     Age     Sex    Income    Num.TV     Phone   Lines    Tablet   Ereader

 872020       1       2     37000         3         1       0         0         2
 872023       3       2     63100         5         1       0         0         1
 872024       2       2     35800         4         0       1         0         1
 872025       3       2     38100         5         0       0         0         -
 872027       1       2     46200         3         1       0         1         1
 872028       2       2     34700         4         1       1         1         2
 872032       2       2     56100         4         1       0         0         1
```

```
  MODIFY PACESET [ GENERATE Income.Groups = NCOT ( Income, 24999, 59999 ) ;
                   GENERATE Own:C = 'Yes' ;
                   IF SUM ( Phone TO Ereader ) = 0, SET Own = 'No' ],
    OUT PACESET $

SAVE.LABELS 'Paceset.lab';
Age
  ( 1 ) 'Under 30'  ( 2 ) '30 to 50'  ( 3 ) 'Over 50' /
Sex
 ( 1 ) 'Male'      ( 2 ) 'Female'                    /
Num.TV 'Number of television sets in your home:'  ( 5 ) '5 or more' /
Phone  'Smart phone as primary telephone'
            'Which of the following do you own?'  /
Lines       'One or more landline based telephones' /
Tablet      'one or more tablet computers'         /
Ereader     'one or more Ereaders'                /
Store.1     'Type of store item purchased in'
 ( 1 ) 'Dis*count'   ( 2 ) 'Depart*ment'
 ( 3 ) 'Web based source' /
Income.Groups  'Income of respondent'
 ( 1 ) Under $30,000 ( 2 ) $30,000 to $60,000  ( 3 ) Over $60,000  /
Income 'Income of respondent' /
Own <<Respondent ownership>> <<& &>>
<<of electronic items>> /
$
```

```
  FILE PaceNew
                                                 Num                          Tablet::
                    Sex::                  TV::Number    Phone::  Lines::One    One or
          Age::   Gender                          of      Smart     or more      more
        Age of       of                  television   phone as    landline    tablet
         respo    respo                     sets in    primary       based      comp
     id   ndent    ndent  Income          your home  telephone  telephones     uters

$
```

If there is no STUB subcommand, all the variables in the file that are not used in other ways are treated as the STUB variable list. Given the variables in file Paceset, the following statements have exactly the same result:

```
STUB Phone TO Ereader

STUB Phone,  STUB Lines Tablet,  STUB Ereader,
```

There may be only one BANNER statement for each group of tables. A group of tables is composed of any number of STUB subcommands and at most one BANNER subcommand. All the tables in a group have the *same layout and options*. The only subcommands which may be provided for individual tables within a tables group are those used to provide a table title and definitions for nets or subtotals.

A group of tables ends with a ";". This semicolon means that the table definitions are complete. The data is read and the current group of tables is printed. Many groups of tables can be given in a single SURVEY command. This is an example of a SURVEY command with two groups of tables.

```
SURVEY Paceset, LABELS 'Paceset.lab' ;
    BANNER AGE SEX,
    STUB Phone TO Ereader ;
    BAN    Income.Groups,
    STUB   Age Sex Own ;
$
```

Notice the use of the semicolons ";" in this example. Because the SURVEY command always has subcommand information, the command itself ends with a ";" indicating the end of the command proper and the beginning of the SURVEY subcommands. The semicolons that appear in the subcommands are action requests. They tell the command that it now has all the information needed to produce the next group of tables. Each group of tables requires a separate pass through the data file.

There is no set limit to the number of banner variables. There *are* limits on the number of banner points that print on a single page and there *are* limits on the total number of cells that can be represented in a single table. It is only necessary to have the resources, given a set of banner variables, to handle a single stub variable. If the tables are so large that all the stub variables cannot be processed at once, the SURVEY command does as many stub variables as it can in a pass of the data and then continues to make more passes until all the stub variables are processed.

The program limits are very large and allow thousands of cells. The exact number depends on a combination of many factors including the size of P-STAT being used and on the SURVEY options that are selected. See the section on "Program Limitations" in a later chapter for details.

If the total number of banner points does not fit on a page (or the screen) it is continued on subsequent pages. As a rule the program will not break a banner variable across pages if it can fit on a page by itself. This behavior can be controlled and is discussed in the chapter "SURVEY: More About the Columns".

The SURVEY command assumes that the values appropriate for a table are more or less continuous integers. Space is allocated to hold all the cells that might be filled given the range of the stub and banner variables. A table of age by sex, when ages are actual years might require 200+ cells. A table of sex by three age groups requires only 6 cells. The number of cells depends on the observed range of the variables. Thus a variable such as Income coded in dollars might require many thousand cells if used as a stub or banner variable and cannot be handled without some special processing. See the section "Sparse Data Values" in a later chapter.

It is always a good idea to ***drop any character variables and any variables with a very large range that are not to be used in the current SURVEY command***. You may use DROP or KEEP to modify the file as you give it to the SURVEY command.

```
SURVEY Paceset [ KEEP Age Sex Num.TV Income.Groups ],
   LABELS 'Paceset.lab';
   BAN Age  Sex ;
```

```
SURVEY Paceset [ DROP Income .CHARACTER. ],  LABELS 'Paceset.lab';
   BAN Income.Groups;
$
```

Note the use of ".CHARACTER." in this example to drop all instances of character variables. Because character variables must be mapped into an appropriate numeric form before SURVEY can handle them, performance may suffer if variables such as "Name", which will not be used, remain in the file as it is passed to the SURVEY command.

Figure 1.2 A Simple Survey with Labels

```
SURVEY  Paceset,      LABELS 'Paceset.lab';
   BANNERS Age    Sex,
   STUB    Income.Groups ;
$
```

```
                              ======== Age ======== ==== Sex====

                        Total  Under  30 to   Over
                       Sample     30     50     50   Male Female

 Total Sample             80     28     27     23     39     40
                         100.0% 100.0% 100.0% 100.0% 100.0% 100.0%

 Income of respondent

 Under $30,000             2      -      2      -      -      2
                          2.5%           7.4%                 5.0%

 $30,000 to $60,000       50     18     19     13     21     28
                         62.5%  64.3%  70.4%  56.5%  53.8%  70.0%

 Over $60,000             20     10      3      6     10     10
                         25.0%  35.7%  11.1%  26.1%  25.6%  25.0%

 Missing                   8      -      3      4      8      -
                          10.0%         11.1%  17.4%  20.5%

 Base                     72     28     24     19     31     40
                         90.0% 100.0%  88.9%  82.6%  79.5% 100.0%
 Mean                    2.25   2.36   2.04   2.32   2.32   2.20
 S.D.                    0.50   0.49   0.46   0.48   0.48   0.52
```

1.3 The Basic Table

Figure 1.2 illustrates the use of the SURVEY command with the data and labels shown in Figure 1.1 to produce a single table. There are only two command elements and two subcommand elements that are needed to produce this basic table. The command elements are the name of the P-STAT system file which contains the data, and the LABELS identifier which points to the file containing the label text. The LABELS are not required but it is the labelling that makes a table easy to read and understand.

```
SURVEY  Paceset,   LABELS 'Paceset.lab' ;
```

The two subcommand elements are BANNERS to specify the column variables, and STUB to specify the row variables.

```
BANNERS Age Sex,   STUB Income.Groups.
```

If STUB is not supplied, all the variables in the file except the two BANNER variables are used as STUB variables. If BANNERS is omitted, the table is a 1-way table or frequency distribution of the variable Income.groups.

Note: Abbreviations for both SURVEY subcommands and variable names are permitted. There is little confusion with the abbreviations of survey subcommands such as BAN for BANNER. However, abbreviations of variable names, while legal is dangerous. It is permitted only to be compatible with previous P-STAT releases. The best practice is to use wildcards (?) rather than abbreviations for variable names.

The NO ABBREVIATION subcommand applies only to the abbreviation of variable names. Abbreviations of SURVEY keywords will continue to work.

```
SURVEY Paceset [ KEEP Age Sex Num.TV Income.Groups ],
   LABELS 'Paceset.lab';
   NO ABBREVIATIONS,
   BAN A?...;
```

Figure 1.3 Changing the Layout

```
SURVEY Paceset, LABELS 'Paceset.lab';
  SET.LABELS BAN TAGS,   BAN Sex,   STUB Own,
  LAYOUT QUESTION LABELS BODY TOTAL.AREA;
$

Respondent ownership of electronic items

                      ==== Sex ====

               Total
              Sample   Male Female

no               11      5      6
                1.8%  12.8%  15.0%

yes              69     34     34
                86.2%  87.2%  85.0%

Total            80     39     40
Sample         100.0% 100.0% 100.0%
```

1.4 CONTROLLING THE CONTENTS OF THE TABLE

The contents and the arrangement of the table sections are determined by a number of different subcommands. The LAYOUT subcommand controls the order in which the sections of a table are arranged. Each section in the layout has controls to specify what statistics are desired and the order in which they should appear. Then there are 5 basic subcommands which control the space that is available across the page.

1.5 Table Organization: The Sections

The “layout” refers to discrete sections of the table which can be moved or deleted. The organization of the table from top to bottom is divided into 8 areas. These areas can be rearranged or omitted with the LAYOUT subcommand. The areas are:

1. LABELS — contains the variable names or extended labels and the values or value labels for the banner (column) variables.
2. TOTALS.AREA — contains the totals for the columns. This is assumed to be the total count of cases for each of the columns. It usually includes all cases including those with missing values on the stub variable. The counts of the cases with good (non-missing values) on the stub variables appears as the BASE in the summary section.
3. QUESTION — contains the variable name or the extended labels for the STUB (row) variable.
4. BODY — contains the cells that are defined by the values of the STUB and BANNER variables. If the stub variable contains the values 1 to 5, the body will contain 5 groups of rows. The assumption is that both the frequencies and column percents are printed for each of the cells.
5. SUBTOTALS — contains subtotals or nets. This section is not printed unless subtotal or net definitions are supplied. SUBTOTALS are discussed in a later chapter.
6. MISSING — contains the counts of the missing values on the STUB variable for the row total and each of the banner values. It is assumed that the three types of missing are to be combined and reported as a single value.
7. SUMMARY — contains summary statistics of the STUB variables for the totals and for each banner column. This is usually composed of the base (the good or non-missing count), the mean and the standard deviation.
8. F.CHI — contains additional summary statistics, chi-square and F.

In these chapters the use of the words “assumed” or “usually” indicates a setting that may be changed. If the assumptions produce a table that has the information you want in a format that meets with your approval you need only supply the names of the BANNER and STUB variables. If you want additional (or less) information or if the way that the table in arranged is not appropriate there are subcommands that can make these changes.

1.6 The LAYOUT Subcommand

The LAYOUT subcommand is composed of the keyword LAYOUT followed by one or more of the 8 keywords denoting the sections of the table. If the LAYOUT subcommand is not used, the assumed order is:

```
LABELS TOTALS.AREA QUESTION BODY SUBTOTALS MISSING SUMMARY F.CHI
```

In Figure 1.3, the LAYOUT command:

```
LAYOUT  QUESTION  LABELS  BODY  TOTAL.AREA
```

has two effects. First it specifies the desired order of the 4 sections that are cited. Second, by omitting MISSING and SUMMARY, it assures that they will not appear in the printout. If a LAYOUT subcommand is used, any section that is not included will not print even if it seems indicated by other subcommands.

Figure 1.4 The LAYOUT Subcommande

```
SURVEY  Paceset,   LABELS  'Paceset.lab' ;
          BAN Age,  STUB Num.TV Income,
          LAYOUT  QUESTION  LABELS  SUMMARY,  NO S.D;
       $
```

```
Number of television sets in your home:

                      ======= Age ========

                Total  Under  30 to   Over
               Sample     30     50     50

Base              80     28     27     23
                 100.0% 100.0% 100.0% 100.0%
Mean              3.10   2.36   3.22   3.83


Income of respondent

                      ===    Age of    ===
                      ===  respondent  ===

                Total  Under  30 to   Over
               Sample     30     50     50

Base              72     28     24     19
                  90.0% 100.0%  88.9%  82.6%
Mean             44898  47388  39311  46080
```

Figure 1.4 illustrates a layout with only the QUESTION, the LABELS and the SUMMARY section. Because there is no body included in the layout, a variable such as Income that would cause a space problem can be used as a STUB. Any variable which has hundreds of possible values is not suitable in the body of a table. It cannot be displayed in a reasonable number of pages even when there is room enough in the P-STAT workspace to store the information. However, when there is no need to store the cells that comprise the body, these variables can be used to get summary information.

Figure 1.4 also illustrates the use of the keyword NO to omit an item from the SUMMARY section.

```
NO STANDARD.DEV      or
NO S.D.
```

causes the standard deviation, which is one of the assumed summary statistics, to be left out of the section. Similarly, NO MEAN or NO BASE is used when those statistics are not needed in the summary section.

The chapter “SURVEY: The Statistics” describes the features of the SUMMARY section, the statistics that are available and how to select the ones that you want. It also documents which features in the summary require the body information and which can be used with variables such as Income to produce a report of summary information.

Figure 1.5 contains the same information as the table in Figure 1.3. However, in Figure 1.5 the sections are in the assumed order. Instead of a LAYOUT subcommand the keyword “NO” is used with name of each layout section that is not needed in the table.

```
NO MISSING,  NO SUMMARY,
```

If you prefer a layout that is different from the assumed layout, you may supply it once at the start of a SURVEY command and then use the section names to delete or reinsert sections for individual tables.

```
LAYOUT QUESTION LABELS BODY SUBTOTALS MISSING TOTAL SUMMARY,
BAN Age, STUB Income.Groups,   NO SUMMARY ;
```

```
STUB Phone TO Ereader,  NO MISSING, SUMMARY ;
STUB Income, NO BODY;  $
```

Figure 1.5 Deleting a Table Section

```
SURVEY Paceset, LABELS 'Paceset.lab' ;
  BAN Sex,  STUB Own,
  NO MISSING, NO SUMMARY ;
$

                      ==== Sex ====

               Total
              Sample   Male Female

Total            80     39     40
Sample        100.0% 100.0% 100.0%

Respondent ownership of electronic items

no               11      5      6
                13.8%  12.8%  15.0%

yes              69     34     34
                86.2%  87.2%  85.0%
```

The following example does the same thing using the LAYOUT subcommand.

```
LAYOUT QUESTION LABELS BODY MISSING TOTAL,
BAN Age, STUB Income.Groups ;

LAYOUT QUESTION LABELS BODY TOTAL SUMMARY,
STUB  VCR TO Ans.Mach ;

LAYOUT QUESTION LABELS TOTAL SUMMARY,
STUB Income ;
```

Either way is acceptable and it is a matter of personal preference which you choose. While the second example requires more key strokes, the resulting clarity may be worth the effort.

1.7 The BODY: Counts and Percents

The body contains the rows that are designated by the values of the STUB variable. Each “row” usually consists of 2 lines, the count of cases that belong to a given cell and the percent that the cell is of its column. This assumption can be changed by using any of the following:

1. NO COUNTS to produce a table with only the column percents
2. NO PERCENTS to produce a table which has only the counts
3. The PERCENT subcommand to request different percentages
4. BODY CONTAINS to specify what should appear in each line of the body and in what order the contents should be printed

Figure 1.6 uses the NO COUNTS subcommand to produce a table with nothing but row and column percents in the body section. When there are no counts and row percents are requested, the row total contains not the per-

cent, which will always be 100, but the count of cases in that row. The percents appear in the table in the order that they are given. Percents are usually based on the total count (TOTAL.N) which includes any missing values. However, the PERCENTS subcommand can be used to specify that the good count (GOOD.N) be the base when calculating percents.

```
ROW COLUMN PERCENTS BASED GOOD.N       is the same as
ROW PERCENTS BASED GOOD.N,  COLUMN PERCENTS BASED GOOD.N,
```

In the next chapter there are some examples of multiple response variables which often have the percents based on the responses.

Percents can be the percent that a cell is of its column, or the percent that it is of its row, or the percent that it is of the total count. The base for calculating the percent is either the TOTAL.N, the GOOD.N or the RESPONSES.

```
COLUMN ROW TOTAL PERCENTS,
COLUMN PERCENTS BASED GOOD.N,  ROW PERCENTS BASED TOTAL.N,
ROW PERCENTS BASED GOOD.N,
COLUMN PERCENTS,
```

Figure 1.6 Table of Percents

```
SURVEY Paceset,  LABELS 'Paceset.lab';
    BANNER Age,   STUB Own,
    NO MISSING, NO SUMMARY, NO COUNTS,,
    ROW COLUMN PERCENTS;
```

```
                        ======== Age =======

                  Total  Under  30 to   Over
                 Sample     30     50     50

Total               80     28     27     23
Sample           100.0%  35.0%  33.8%  28.7%
                 100.0% 100.0% 100.0% 100.0%

Respondent ownership of electronic items

no                  11   18.2%  54.5%  27.3%
                  13.8%   7.1%  22.2%  13.0%

yes                 69   37.7%  30.4%  29.0%
                  86.2%  92.9%  77.8%  87.0%
```

These are all valid PERCENT subcommands. PERCENT may be preceded by any or all of:

```
ROW      COLUMN      TOTAL
```

and followed by BASED and one of:

```
TOTAL.n   GOOD.n     RESPONSES
```

If the base is the same, all the percent requests may be joined in a single subcommand. There may be up to 5 different percents calculated in a single table. It is possible to get a percent twice in a single cell, calculated once using the TOTAL.N as the base and again, using the GOOD.N as the base.

```
COLUMN PERCENTS BASED GOOD.N,   COLUMN PERCENTS BASED TOTAL.N,
```

Figure 1.7 illustrates the BODY subcommand. Each section of the LAYOUT is itself a subcommand which is used to specify the exact contents of that section. The usual contents of the BODY are the counts and the percents. As you can see by looking at the figure, the PERCENTS subcommands are needed if anything other than column percents is desired. The LAYOUT subcommand is used to select the desired sections of the table and the order of those sections. The BODY CONTAINS subcommand is needed because we want the counts to follow the percents, which is not the assumed order.

BODY CONTAINS can be used to include other information in the cells. If the tables are weighted (see the section later in this chapter), the cells contain the weighted count. Both weighted and unweighted counts can be requested.

```
BODY CONTAINS WEIGHTED.N UNWEIGHTED.N PERCENTS,
```

Other features for BODY CONTAINS are discussed in the chapter “SURVEY: Special Features of the Layout Sections”.

Figure 1.7 BODY CONTAINS Subcommand

```
AGAIN,  COLUMN PERCENTS,  ROW PERCENTS,
        LAYOUT LABELS TOTALS QUESTION BODY,
        BODY CONTAINS PERCENTS COUNTS;

                        ======== Age =======

                  Total  Under  30 to   Over
                 Sample     30     50     50

Total               80     28     27     23
Sample           100.0% 100.0% 100.0% 100.0%

Respondent ownership of electronic items

no                13.8%   7.1%  22.2%  13.0%
                 100.0%  18.2%  54.5%  27.3%
                    11      2      6      3

yes               86.2%  92.9%  77.8%  87.0%
                 100.0%  37.7%  30.4%  29.0%
                    69     26     21     20
```

1.8 Row Totals and Spacing

There is less that can be done with the left to right arrangement of the table layout. However, the row totals are a section that is separate from the body. The row totals are referenced by the subcommand ROW.TOTALS. They can be removed or moved to the right hand side of the table. If there are too many banner points to fit on a page, the assumption is that the row totals will print on every page. This can be changed so that the row totals print but once, either as the first column of the table or as the last column.

```
ROW.TOTALS ON LEFT,       (assumed)
ROW TOTALS ON RIGHT,
ROW.TOTALS ON LEFT ONLY,
ROW.TOTALS ON RIGHT ONLY,
NO ROW.TOTALS,
```

It is the addition of the keyword “ONLY” that causes the row totals to print but once. The word “ON” is not necessary but makes the subcommand read more smoothly.

There are several subcommands that control the spacing of the columns. When a column has more space, there is more room for label text and it is easier to format the labels in an attractive way. However, if more space is used for each column, fewer columns fit on the page. The five basic subcommands to control the spacing are OUTPUT.WIDTH, MARGIN, GAP, CELL.WIDTH and SKIP.

Figure 1.8 Spacing Options

```
SURVEY Paceset,  LABELS 'Paceset.lab';
  BAN  Sex  Age,      STUB Own,
  MARGIN 10,          CELL.WIDTH 9,
  GAP     4,          OUTPUT.WIDTH 72,
  NO ROW.TOTALS ;
$
```

	====== Sex ======		========== Age ==========		
	Male	Female	Under 30	30 to 50	Over 50
Total	39	40	28	27	23
Sample	100.0%	100.0%	100.0%	100.0%	100.0%
Respondent ownership of electronic items					
no	5	6	2	6	3
	12.8%	15.0%	7.1%	22.2%	13.0%
yes	34	34	26	21	20
	87.2%	85.0%	92.9%	77.8%	87.0%
missing	-	-	-	-	-
Base	39	40	28	27	23
	100.0%	100.0%	100.0%	100.0%	100.0%
Mean	1.87	1.85	1.93	1.78	1.87
S.D.	0.34	0.36	0.26	0.42	0.34

OUTPUT.WIDTH specifies the number of print positions or spaces that are available across the page. Usually the output width is defined once for all the tables in a single survey command. The output width when defined as a subcommand must be an integer between 50 and 1200. The output width can be defined as an attribute of the current print destination. (See the section on “Printing the Tables” later in this chapter.) However, if the width is to be greater than 132 spaces, it must be given as a SURVEY subcommand. For example

```
OUTPUT.WIDTH   220,     or
OW             220,
```

If the real output width of the printer or terminal is less than the defined output width, lines that overflow will either wrap on to the next line or be truncated depending on the type of print destination.

MARGIN is used to specify the width of the left edge area where the stub labels are formatted. If the MARGIN subcommand is not used, the program looks at the length of the labels for the stub variable, at the current output width and at the number of banner points and makes its own decision on the number of spaces that are need-

ed to hold the stub labels. The value for the MARGIN must be an integer between 8 and 80 with at least 12 spaces for the body of the table.

GAP controls the spacing between banner variables, and between the row totals and the adjacent banner variable. A gap of 0 is assumed. The maximum gap is 10.

CELL.WIDTH is used to change the assumed space for an individual column. In most circumstances that assumed cell width is 7 spaces. This allows 6 spaces for the counts or percents and 1 space between columns. When row totals are present they are allowed an extra space. The minimum allowable cell width is 5, the maximum is 16. If you are working with very large weights designed to bring the totals up to population figures (for example) you should check the section on "Weighted Tables" in a later chapter.

The cell width for each column will be the same in most, but not all, situations. If means, sums, or medians are specified for a variable other than the stub variable, the assumed default cell width may not be enough to print the number. In this situation, the command itself will try to format the number in a reasonable way. The SPREAD and COMPACT subcommands which are described later in the manual can be used to control this behavior.

SKIP is used to control the location of the blank lines in the body of the table. The default is a blank line between each of the stub values. If the body of the table contains frequencies and column percents, a blank line is left between each 2 lines of printing. IF NO SKIP or SKIP 0 is used, these blank lines are omitted. SKIP with a number other than 0 indicates the number of stub values to be processed before the next blank line.

```
SKIP 3,
```

NEVER SKIP can be used to remove all blank lines from the table. This option is seldom used unless the output is to be post-processed by another computer program. SKIP.RULES which is described in the chapter "SURVEY: Enhancing Table Appearance" provides complete control over the use of blank lines in a table.

Figure 1.8 shows a survey with no row totals and with four of the spacing subcommands. The effect of the cell width is that the value labels for variable age do not need to be stacked on multiple lines. A cell width of 10 might enhance this even more. The gap between the banner variables makes the table easier to read. However, there are times when it is more important to get as much information as possible on a single page. When this is the case, small cell widths and no gap are needed.

1.9 TITLES AND LABELS

Titles and labels make the SURVEY display more attractive, more informative and easier to understand. The use of titles and labels in SURVEY is similar to their use in other commands. However, SURVEY makes use of some features that are not available or needed in other P-STAT commands. Both titles and labels are discussed in full in the chapter "Titles, Labels, and the MAP Command". This section emphasizes the special features that are utilized by the SURVEY command.

The titles feature in SURVEY, that is not generally available in other commands, is the ability to define local titles and to temporarily suspend previous titles using the subcommand language. The labels features include both the length of the labels that can be used and some special controls you can place in the labels file to specify how the labels should be formatted. In addition to P-STAT system titles, TABLE.TITLES provides additional titles for individual tables.

1.10 Labels in the SURVEY command

Labels are contained in disk files created either by using the SAVE.LABELS command in P-STAT or by an external system editor or word processor. The LABELS identifier, used in the SURVEY command, has one or more quoted arguments that are the names of the disk files containing the labels. The last mentioned labels file has precedence when a variable has value labels in more than one of the labels files. Similarly, if a variable has extended labels in more than 1 of the labels files, the last one takes precedence. The extended labels for a variable and the value labels may be in different labels files.

```
File lab1                          File lab2
```

```
Age  'Age of Respondent'          Age (1)  1-10  (2) 11-20
  (1) Under 20  (2) 20 to 50          (3) 21-30  (4) 31-40
  (3) Over 50   /                     (5) 41-50  (6) Over 50 /

Survey Paceset,  Labels 'lab1'  'lab2';
```

The extended label in this example are taken from file lab1, the value labels from lab2.

The text of each extended label or value label must fit on a single 80 character record. Since extended labels must always be contained in quotes or angle brackets, this limits each label to 78 characters. The angle brackets "<<" and ">>" have the advantage that they are easier to see than either the single or double quotation marks. Unbalanced quotes cause error messages. The angle brackets may be used instead of quotes in labels files and in the subcommand language. They are not, however, supported in command text.

Value labels need not be in quotes or angle brackets unless they contain special characters such as a slash or a left parenthesis. An unquoted/bracketed value label may contain 80 characters. A quoted value label may contain no more than 78 characters.

```
File longlab

Q1
<<place the entire text on the single line with the final bracket here>>
<<it may be followed by yet more extended label text>>
( 1 )
The value label text can extend from the first column to the last one
/
```

A very long label may be too long to fit in its entirety in all print situations. There is more flexibility in labelling the stubs than there is in labelling the banners, which have fairly rigid column constraints. The extended label for the stub variable is printed in the QUESTION section of the layout. Because there are no other items in this section, the entire width of the page is available and as many lines as necessary to hold the label information may be used. If the output width available is much greater than 80 characters, it may even be desirable to join two or more of the extended labels to make a single very long text string.

```
Q1
<<This is the first line of text for question Q1  The ampersands >>
<<&&>>
<<indicate that the lines should be joined with no blank between>>

<<If used for a stub variable this will begin the second line and>>
<<& &>>
<<will have a blank inserted before it is joined with this line>> /
```

SURVEY recognizes the special extended label text:

```
<<&&>>     and   <<& &>>
```

as requests to join two pieces of an extended label together. If there is a blank between the ampersands, a blank is inserted between the two lines of text. This feature applies only to the labels when they are used to label stub variables. Only the first label is used for banner variables. If the first extended label is too long to fit in the banner situation, it is truncated. The ampersand feature makes it easy to construct extended labels that are appropriate for a variable in either a banner (column) or stub (row) situation. For example:

```
Q3a <<Preferences in soaps>>   <<& &>>
    <<used for laundry and household cleaning>> /
```

The formatting of value labels is also easier for the stubs than the banners. The stub value labels are printed in the margin area. If a label has 30 characters and the margin has room for 20, it is broken across 2 lines. If

possible such breaks are done between words. If a label has 80 characters and the margin is narrow, it may take 8 or 9 lines to print all of it.

The formatting of the banner value labels depends upon the cell width. The maximum number of lines that will be used is 6. If the cell width is 7, one space is used between variables leaving a 6 by 6 area for the label. The blanks in the label count in the total number of characters that are printed. The label formatter tries to break the lines between words whenever possible.

Figure 1.9 Formatting Value Labels with the Break Characters

```
      File lab1


      var2 <<This is the text for Question 2>>
      ( 1 ) This is a long label that should fit
      ( 2 ) a b c d e f g h i j
      ( 3 ) a^b c^d e^f g^h i^j
      ( 4 ) Togetherness
      ( 5 ) To*gether*ness
      /


  SURVEY t3, LABELS lab1;
         STUB var1, BAN var2;
```

```
                    ====== This is the text for ======
                    ======      Question 2      ======

                    This i
                    s a lo           a b
                    ng lab  a b c    c d
                    el tha  d e f    e f            To
              Total t shou  g h i    g h Togeth gether
             Sample ld fit      j    i j erness   ness

Total            12      4      2      2      2      2
Sample        100.0% 100.0% 100.0% 100.0% 100.0% 100.0%
```

Figure 1.9 illustrates the formatting that is done and the use of special characters to control the break points. There is very little, given the cell width of 7, that can help the formatting of the first label which uses all available spaces. The label that is composed of 10 separate letters is usually broken by the formatter after every third blank. When the special character that requests that the pieces *not* be broken at a particular spot is used to change the break points two characters per line are formatted across five lines. This character is usually the not sign or caret (^) The no break character in the label is replaced by a blank for printing.

Figure 1.10 shows the improvement in all the labels when the cell width is increased to 9 columns. This is particularly true for the long label of the first banner point.

Figure 1.10 More Space Improves the Appearance

```
 CELL WIDTH 9,   MARGIN 8,

                     ====== This is the text for Question 2 =====

                      This is
                       a long
                        label
                         that  a b c d  a b c d
                Total   should  e f g h  e f g h   Togeth Together
               Sample      fit      i j      i j   erness     ness

 Total            12        4        2        2        2        2
 Sample        100.0%   100.0%   100.0%   100.0%   100.0%   100.0%
```

The label formatter breaks a long word in the middle. A break character which is assumed to be the asterisk (*) is used to advise the formatter about reasonable places for breaking a character string. It is not an order, just a suggestion. If the formatter does not honor the suggestion, it removes the asterisk and squeezes the label pieces together. You can see the effects of the break character by examining the difference between the two columns labelled “togetherness” in Figures 1.9 and 1.10.

The BREAK and NO BREAK characters must be characters that do not appear in the label text. If you wish to use the * or ^ as part of your labels, you must request a different set of characters by using the CHARACTER subcommand.

```
CHARACTER for BREAK <<!>>
CHAR BREAK          <<+>> <<@>>
CHAR BREAK          <<*>> <<^>> <<*>>
```

Figure 1.11 Break Character in an Extended Label

```
The labels file, lab1, now has this text as the extended label for Var2

  Var2 <<This is the^text for^Question 2>>

  SURVEY t3, LABELS lab1;
         STUB var1, BAN var2,
         CHARACTER BREAK  <<*>>   <<^>>  <<*>>;


                      ======== This is the text ========
                      ========  for Question 2  ========
```

The CHARACTER (for) BREAK subcommand supports 1 to 3 arguments. Each argument is a single character in quotes or angle brackets. The first character is the BREAK character. The second character is the NO BREAK character. Usually only the value labels are affected by the break rules because they are the ones with the most restrictions and therefore the most difficult to format attractively. However, if there is a third character, the extended labels *for the banner* are also controlled by break characters. The third character is used to specify a break in the extended label and the second character is used as the hold character in both situations.

Figure 1.11 illustrates the formatting of an extended label when break characters are used. The third break character is needed to tell the variable labels formatter to use the break/hold features. This text would usually be broken with “This is the text for” on the first line. The hold character is used to keep the words “the” and “text” together.

1.11 Titles in the SURVEY command

The TITLES that are used in SURVEY can be defined either by using the TITLES command or by entering single titles as SURVEY subcommands. In either case the language is the same.

```
      TITLES T1 C 'text'
```

Figure 1.12 Titles

```
   TITLES T1 'title 1 center'  T3 L 'titles 3 left'
          T4 R 'titles 4 right' $

   SURVEY Paceset, LABELS 'Paceset.lab' ;
      TITLES T5 'Title 5:  Report by XYZ Inc.',
      BAN ....,   STUB .... ;


                        titles 1 center

titles 3 left
                                                        title 4 right
                  Title 5: Report by XYZ inc
```

```
   TITLES BLANK T4,
   TITLES SUSPEND  T5,

   BAN ....,   STUB ..... ;

                        titles 1 center

titles 3 left
```

```
   TITLES T6 'Title 6: Restore title 5.  Note title 4 is now blank',
   BAN ....,  STUB .... ;

                        titles 1 center

titles 3 left

                  Title 5: Report by XYZ inc
       Title 6: Restore title 5.  Note title 4 is now blank
```

The keyword “TITLES” is followed by T1 through T9 or B1 through B3 to designate which of the 9 top titles or 3 bottom titles is being defined. This is followed by LEFT CENTER or RIGHT (L/C/R) to designate which area of the title is being defined. Finally there is title text in quotes. If the first argument (T1-T9, B1-B3) is not present, T1 is assumed. If the second argument (L/C/R) is not used, CENTER is assumed.

TITLES that are defined using the TITLES command are available for use until they are reset or replaced. TITLES defined in a SURVEY subcommand are available only for the duration of that command.

If all of the titles are defined in the TITLES command, they will not print unless the TITLES identifier is included in the SURVEY command.

```
TITLES  'January 4,1994' $
SURVEY Paceset, LABELS 'Paceset.lab', TITLES ;
```

The TITLES identifier is not necessary if local SURVEY titles are defined in the subcommands. The title text in the quotes may contain P-STAT system variables or permanent scratch variables. This provides a way for the title text to be modified dynamically.

```
TITLES '.DATE.' $
SURVEY Paceset, LABELS 'Paceset.lab' ;
    TITLES B1 LEFT 'Page .PAGE.', ....
```

The top center title contains the current date and is available for all commands that use titles including the SURVEY command. The first bottom title has on the left the text “Page” followed by the current page number. This title is only available for the current SURVEY command.

Further control over TITLES is provided by the JOIN subcommand which is described in the chapter “SURVEY: Enhancing Table Appearance”.

The TITLES subcommand in SURVEY allows a single title line to be suspended or set to blank. When a title line is suspended, it and any title lines below it are temporarily forgotten. These titles will be remembered again if another lower level title is later defined. If the title that was suspended, and therefore forgotten, is a local title, it can be recalled only in the current command.

If BLANK is used without the L/R/C position indicator, all three of the titles for that titles line are set to blanks. If a position is indicated, just that position is affected.

```
TITLES T4 LEFT BLANK,    blanks out just the T4 LEFT title
TITLES T4 BLANK,         blanks out all 3 T4 titles.
```

If a title, defined before the SURVEY command, is changed or set to blank by a SURVEY subcommand, it becomes a local title and is not available after the current command. In general the TITLES command is used to define those titles that apply to the entire run. Local titles are usually placed below the general titles and are changed, reset, redefined or set to blank depending on the contents of the individual tables group. TITLES are printed whenever a page change is requested.

The text of the TABLE.TITLE subcommand is in addition to any TITLES that are supplied. Table titles always appears left justified after the page TITLES and before the first line of the table itself. There is no limit to the number of lines of text that can follow the TABLE.TITLE subcommand. The lines of text can be joined in the same way that lines of extended labels are joined. TABLE.TITLE is discussed in more detail in the chapter “SURVEY: Other Features” in the TABLE OF CONTENTS section.

TABLE.TITLES are not stored and are associated with the immediately following stub variable. The 2 tables in Figure 1.13 are processed in a single pass of the data. If the second TABLE.TITLE is omitted:

```
TABLE.TITLE 'Basic demographic information'
            'asked of all respondents',
STUB Sex,
STUB Income.groups;
```

the table titles is not used for the Income.groups table.

Figure 1.13 TABLE.TITLES in the SURVEY command

The command

```
SURVEY Paceset,  LABELS 'Paceset.lab';
       TITLES '.DATE.', BANNER Age,
       NO MISSING, NO SUMMARY,

       TABLE.TITLE 'Basic demographic information'
                   'asked of all respondents',
       STUB Sex,

       TABLE.TITLE 'Recoded from raw income values',
       STUB Income.groups;
```

The first table

```
              Mon May  8  1995

  Basic demographic information
  asked of all respondents

                       ======== Age =======

                  Total  Under  30 to   Over
                 Sample     30     50     50

  Total             80     28     27     23
  Sample         100.0% 100.0% 100.0% 100.0%

  Sex

  Male              39     20     11      7
                  48.8%  71.4%  40.7%  30.4%

  Female            40      8     16     15
                  50.0%  28.6%  59.3%  65.2
```

1.12 PRINTING THE TABLES

It is very useful when you are working interactively to view at least some of the tables on the screen. However, once the options and layouts have been selected, you will usually want your tables sent either to the printer or stored on a diskfile. A table that is stored in a disk file can be included in another document. It can be edited and printed later at a convenient time. Tables can also be “printed” in a SPREADSHEET format which can be read directly into spreadsheet programs or text processors such as Excel.

PR is the keyword that is used to indicate where the table printout should go. PR can be used as a command, an identifier, or a SURVEY subcommand. The attributes of the printout depend on the output width and lines settings. There are options to echo the command and the subcommands. If the table is to go directly to a printer or the system print queue, the characters which indicate a page change must be correct for that printer. The

P-STAT commands which allow you to specify printer attributes are DEFAULT.SETTINGS and PRINTER.SETTINGS. These are documented in the manual "P-STAT: Utility Commands".

The choice of attributes depends on what you wish to do with the tables you produce. If you are storing them in a disk file so that you can include them in a report, the PAGE.CHARACTER should be set to a blank.

```
PAGE.CHARACTER ' ',
```

If you are planning to send the stored files to a printer or a print queue, the ASCII number 10, which is a form feed, is the appropriate choice for Unix or PC/windows. The character `'1'` is used for mainframe computers or on machines which have a Fortran print filter. If you have a laser printer with PostScript, you can choose your own fonts and produce camera ready printout. This is described in the chapter “SURVEY: Enhancing Layout and Appearance”.

In the following example, attributes for a file named Prfile are set with a PRINTER.SETTINGS command. The PR command then causes Prfile to be the current output device. All printout goes to that file until another PR command changes the destination or, if the run is interactive, there is an error.

```
PRINTER.SETTINGS  'Prfile',   OUTPUT.WIDTH 100,
        PAGE.CHARACTER 12,  NO ECHO $
PR  'Prfile' $

SURVEY Paceset, .....

SURVEY Test, .....
$
```

When PR is used as an identifier, the printout from that command alone is sent to the designated destination.

```
SURVEY Paceset, LABELS 'Paceset.lab',  PR 'Prfile';
```

Within SURVEY, a given group of tables may be sent to a printer by using PR as a subcommand. This is most common when SURVEY is run interactively and tables are only sent to the printer after they have been examined on the screen and approved.

```
SURVEY Paceset,  LABELS 'Paceset.lab';
BAN Age Sex,  STUB Own;
AGAIN, PR Prfile;
Ban Income.Groups, STUB Num.Tv;

AGAIN, PR Prfile;
```

When PR is used as a subcommand, it applies only to the current tables group. There is an interaction of the PR subcommand and the OW (OUTPUT.WIDTH subcommand). PR sets an output width that is appropriate for the named print file (See PRINTER.SETTINGS in the manual "P-STAT: Utility Commands"). If you wish to change this width, the OUTPUT.WIDTH subcommand must follow the PR subcommand.

On PC Windows, you can print directly from the front end MENUS or use the PRINT command. On Unix, there is a shell script called “pstatlp” which can be tailored so that files can be sent to the print queue with the PRINT command.

```
SURVEY Paceset, LABELS 'Paceset.lab', PR 'Prfile';
BAN AGE Sex,  STUB Own;
$
PRINT 'Prfile' $
```

1.13 WEIGHTING THE TABLES

Tables are often weighted when the respondents do not accurately represent what is known about the total population. For example, it may be very difficult to locate adult working males. At the same time it may be easy to interview people who have retired. In many cases the percentages of different groups as determined by background information such as age, sex, education and region are known from census data. Weights may be applied to the SURVEY command to adjust for any under or over sampling.

Figure 1.14 illustrates the use of the BALANCE command to create a file that has weights generated automatically. The output file Pacesetw contains all of the variables in file Paceset plus a variable named Weight which contains weights appropriate for a complex combination of age, sex and income groups. The BALANCE command is discussed in the chapter "BALANCE and SAMPLE".

Figure 1.14 The WEIGHT identifier

```
BALANCE Paceset, CONTROLS Pace.wt,  OUT Pacesetw,  WEIGHT.NAME Weight $
SURVEY Pacesetw,  LABELS 'Paceset.lab', WEIGHT Weight;
   BAN Age  sex,    STUB Own,
   GAP 2,  MARGIN 18,  NO MISSING;

Weighted by: Weight

                              ======== Age =======   ==== Sex ====

                      Total    Under  30 to   Over
                     Sample       30     50     50     Male Female

Weighted Total          82       25     26     29       39     42
                     100.0%   100.0% 100.0% 100.0%   100.0% 100.0%

Respondent ownership of electronic items

no                      11        2      6      4        5      6
                     14.0%     7.0%  22.4%  13.2%    13.6%  14.6%

yes                     71       23     20     25       34     36
                     86.0%    93.0%  77.6%  86.8%    86.4%  85.4%

Base                    82       25     26     29       39     42
                     100.0%   100.0% 100.0% 100.0%   100.0% 100.0%
Unweighted Total        80       28     27     23       39     40
Mean                  1.86     1.93   1.78   1.87     1.86   1.85
S.D.                  0.35     0.26   0.43   0.34     0.35   0.36
```

The output when the SURVEY is weighted has slightly different components than an unweighted survey. The totals area contains the weighted totals and is labelled appropriately. The base (the good or non-missing count) is also weighted. An additional line containing the unweighted total count is automatically added to the summary section of the layout.

To use weights, the WEIGHT identifier must be used in the command. However, once the weight variable has been specified, the subcommands NO WEIGHT and WEIGHT can be used to selectively turn it on and off.

```
SURVEY Pacesetw,  LABELS 'Paceset.lab',  WEIGHT Weight;
```

```
BAN Age Sex, STUB Own;
BAN Sex, Own, MR.STUB VCR TO Ans.Mach,  NO WEIGHT ;
```

The body of the table contains the weighted count. If you wish to see both the weighted and unweighted counts in the cells, use the BODY CONTAINS subcommand discussed earlier in the chapter.

1.14 Means, Medians or Sums of an Outside Variable

An "outside" variable is one that is not one of the stub or banner variables and has no part in defining the rows and columns of the table. The most common use is for variables which have been recoded into ranges. A mean or median of the resulting recoded values is not very interesting.

In our Paceset file, the variables Income and Income.groups are an excellent example. If the stub variable is Income.groups we can not produce a table which contains a body because of the large number of discrete values that are possible. If we produce a table on Income, the means are going to be the mean of the values 1, 2, and 3. The solution is to combine the two variables in a table.

```
STUB Income.groups,  MEANS Income.
```

The means variable need not have any relation to the stub variable and that poses a problem. If the MEANS variable is missing, there is no natural place in the table for those cases. The missing section usually includes only cases that are missing on the stub variable. In file Paceset, variable Income has 8 cases with missing values. These cases are normally excluded from the table when Income is the means variable. However, when Income.Groups is the stub variable, the pattern of missing data is identical for both the stub and the means variables and there is no reason to exclude the missing cases. They can be included by using:

```
INCLUDE MISSING MEANS VALUES,
```

Figure 1.15 shows the SURVEY command and subcommands to produce a table with means on variable Income. Because the Income and Income.Groups are different manifestations of the same information, INCLUDE MISSING MEANS VALUES is appropriate and the totals reflect all 80 cases in the file. Without the INCLUDE subcommand, the totals used in the table would be just the 72 cases with no missing data on the MEANS variable.

The effect of using INCLUDE MISSING MEANS VALUES when the means variable does not have the same pattern of missing as the STUB variable is to change the value of the means variable from missing to zero and the value of the stub variable to missing. The missing count now reflects missing on both the stub and the means variable. The cells contain only those cases with non missing values on both the stub and the means variable. The summary means are the same in both situations and are always based on the sample which has good data for both the stub and the means variable. The chapter "SURVEY: Special Features of the Layout Sections" explains how you can differentiate the cases that are missing on the stub variable from those that are missing on the means variable.

If there is a list of stub variables, you may supply a list of corresponding means variables. The list that follows the MEANS subcommand must have either a single variable or the same number of variables as the STUB or GR.STUB subcommand. If the MEANS subcommand list has a single variable, that variable will be used with each of the stub variables in turn. If the MEANS subcommand has a list of variables, each variable in the list will be used with the corresponding stub variable.

```
STUB Q3 TO Q14, MEANS Income,
STUB Q3 to Q14,  MEANS Q3x TO Q14x,
```

A variable can be repeated in the means list if necessary to make the lists balance:

```
GR.STUB Sex Age Income.Groups,
MEANS Num.TV Num.TV Income,
```

Figure 1.15 Means on an Outside Variable

```
SURVEY Paceset,  LABELS 'Paceset.lab';
   BANNER Sex Own,  STUB Income.groups,
   MEANS Income,
   INCLUDE MISSING MEAN VALUES;
$
```

		==== Sex ====		Respondent ownership	
	Total Sample	Male	Female	no	yes
Total Sample	80	39	40	11	69
	100.0%	100.0%	100.0%	100.0%	100.0%
Income of respondent					
Under $20,000	2	-	2	1	1
	2.5%		5.0%	9.1%	1.4%
$20,000 to $40,000	50	21	28	9	41
	62.5%	53.8%	70.0%	81.8%	59.4%
Over $40,000	20	10	10	-	20
	25.0%	25.6%	25.0%		29.0%
Missing	8	8	-	1	7
	10.0%	20.5%		9.1%	10.1%
Base	72	31	40	10	62
	90.0%	79.5%	100.0%	90.9%	89.9%
Mean Income	34398	36759	32634	26785	35626
S.D.	9334	9340	9149	5410	9277

INCLUDE MISSING MEANS VALUES applies to all the tables in the current tables group and cannot be selectively applied. EXCLUDE MISSING MEANS VALUES is used to restore the default setting.

The language for the SUMS and MEDIAN subcommands is exactly the same as the language for the MEANS subcommand. If any of them are used, the other two statistics, if requested, are also based on that variable. (The use of MEDIANS is discussed in detail in the chapter “SURVEY: The Statistics”.) When either SUMS or MEDIANS is used, that statistic is automatically added to the list of items included in the summary section. The SUMMARY subcommand, which is described in “SURVEY: Special Features of the Layout Sections”, is used to further control the contents of this section.

When an outside variable is cited as the means, medians, or sums variable it will be used in all subsequent tables until it is explicitly changed. If you wish to reset it so that the means are calculated using the values of the current stub variable you may use the RESET subcommand. However, reset restores all the settings except banner and stub variables and subtotal definitions to their original settings. One alternative which affects only the variable used for means, medians and sums assumes that there is no variable in the file named “dummy”. For example:

```
STUB Q3 TO Q14, MEANS income;
STUB Q24 Q26, MEANS DUMMY;
```

Another alternative is to use NO MEANS to turn off means, followed by MEANS to turn them back on.

1.15 CREATING THE SURVEY COMMAND

The SURVEY command can be entered by using an external editor or word processor. It can be entered interactively in response to the prompts from P-STAT and the SURVEY command. It can be built by selecting items from the menus. The examples in this chapter have been written as they would appear in a batch job stream. The only difference between that and entering the commands interactively is the absence of prompt messages.

Figure 1.16 Interactive Prompts and Replies

```
>>    SURVEY Paceset, LABELS 'paceset.lab';

      Begin subcommand, or type  Q  or  H:
>>    BANNER Age,
      Continue subcommand (or Q/H):
>>    STUB Own;

                          ======== Age =======

                    Total  Under  30 to   Over
                   Sample     30     50     50

      Total             80     28     27     23
      Sample         100.0% 100.0% 100.0% 100.0%

      Respondent ownership of electronic items

      no                11      2      6      3
                     13.8%   7.1%  22.2%  13.0%

      yes               69     26     21     20
                     86.2%  92.9%  77.8%  87.0%

      missing            -      -      -      -

      Base              80     28     27     23
                     100.0% 100.0% 100.0% 100.0%
      Mean            1.86   1.93   1.78   1.87
      S.D.            0.35   0.26   0.42   0.34

      Begin subcommand, or type  Q  or  H:
>>    $
```

The following is an example of a batch run. The commands are stored in a file named “pstat.job”.

```
BATCH $
PR Test.prt $
SURVEY Paceset, LABELS 'Paceset.lab' ;
BAN Age Sex, STUB Own;
```

```
$
END $
```

The exact command for using this file in P-STAT depends on the machine being used. The following commands are appropriate for PC/windows and Unix:

```
PSTAT   pstat.job  pstat.out       on a PC/windows PC
p-stat <pstat.job >pstat.out &     on a Unix machine
```

The job can also be run on any machine in interactive mode by invoking P-STAT in the usual way and entering

```
TRANSFER 'pstat.job' $
```

When you are building a SURVEY command with many tables groups, each with different choices for layout and contents, the RESET command is often useful. RESET is used to return all the subcommand values, except the stub and banners, to their original settings. The LAYOUT is reset to the original 7 part arrangement. BODY contents are reset to counts and column percents. ROW.TOTALS are once again on the left side of every page, and so forth. RESET does **NOT** change the BANNER and STUB settings, These may be explicitly reset with:

```
RESET BANNERS,        and
RESET STUBS,
```

This is particularly useful when you have done a series of tables with BANNERS and now wish to do a 1-way table with STUBS. BANNERS can be changed but it is necessary to use RESET BANNERS to remove them.

1.16 Macro Usage

P-STAT macros are very useful when you are doing production runs. They are described in the chapter SURVEY: Other Features. Command macros can be used to encapsulate an entire run setting the parameters that change over time. A RUN command is then used to invoke the macro and supply new parameters such as a file name.

In stream macros are often used to streamline the process of building a complex SURVEY run. These macros can be used anywhere in the SURVEY command and contain snippets of PPL or SURVEY code. For example the preferred layout can be stored in a macro and later referenced by the macro name. Macro usage is described in greater detail in the Chapter SURVEY: Other Features.

SUMMARY

SURVEY

```
SURVEY   Product1   [KEEP  Age  Education  Pepsi  TO  Dr.Pepper]
         Product2   [DROP  V(16)  .ON.] ,
LABELS  'ProdLab1'  'ProdLab2'  ;

  BANNER    Age  Education,
  STUBS     Question1   TO  Question6   ;

$
```

The SURVEY command displays data as tables. The data are often from questionnaires or research studies, but it may be any categorical numeric or character information. The survey typically consists of one or more BANNER variables that define the columns of the display and one or more STUB variables that define the rows of the display. Up to four P-STAT system data files and many label files may be input.

Information required by the SURVEY command is supplied as identifiers and as subcommands. Identifiers (such as LABELS) come before the initial (command-ending) semicolon. Subcommands (such as BANNER or STUBS) come after the initial semicolon.

Required:

SURVEY fn fn

provides the name of one or more P-STAT files that contain the input data. Variable selection phrases (using the PPL instructions KEEP or DROP) may follow directly after each input file to select only the variables necessary for a particular survey. Extra variables reduce the available workspace.

Some of the Optional Identifiers:

LABELS 'fn' 'fn'

gives the names of disk files containing extended variable and value labels. The label file names are enclosed in single or double quotes.

Value labels for the STUBS may be up to 80 characters long. They are formatted to fit within the MARGIN area and they continue on as many lines as needed. (The MARGIN area may be from 8 to 60 characters wide.) Depending on the defined CELL.WIDTH, 24 to 80 characters of the value labels are used for labelling the individual BANNER points.

Extended *variable* labels for STUB variables may be composed of multiple strings, each of which prints on a separate line in the QUESTION area of the survey. Each label may contain an indefinite number of quoted strings of up to 78 characters each. Each string prints on a separate line in the QUESTION (extended variable label) area unless joined by the special “&&” extended labels. Only one string of extended variable labels is used for BANNER variables. The label for the banner variable prints on one or more lines depending on the cell width and the number of banner points (columns) for that variable.

TITLES

turns on any titles previously defined using the TITLES command. Titles may also be supplied within the SURVEY command at the subcommand level.

WEIGHT vn

specifies the name of the variable that is to be used to weight the cases. Weighting may be fractional.

ECHO

requests that the labels be echoed as they are processed.

Optional Subcommands: Defining Rows and Columns

BANNERS vn vn

defines one or more variables as the banners or columns of the display. These are typically the background variables. Any number of banner variables may be defined and the survey continues on as many pages as necessary. BANNERS are not required. If banner variables are not supplied, a one-way display is produced for each of the STUB variables.

STUBS vn vn

defines variables which are the stubs or rows of the survey. These are the analysis or questionnaire variables, and they print one per page

```
BAN  Var1  TO  Var5,  STUB  Q1  TO  Q6;
```

This subcommand defines 6 tables that will almost certainly be processed in a single pass of the data file, depending on the size of P-STAT in use, the data set and the options selected.

Optional Subcommands

ABBREVIATION

of variable names is assumed. NO ABBREVIATION of variable names can be used to require full variable names.. The use of the wildcard (?) is a better way to refer to long variable names.

BASE

requests that a count of non-missing cases be included in the summary section of the table. This is assumed. NO BASE removes the base from the summary.

BODY CONTAINS arg arg arg

is followed by a list of the things that are to be included in the cells of the body of the table. COUNTS, PERCENTS, and UNWEIGHTED.N are possible choices. The order of the list is the order in which they appear.

```
BODY CONTAINS PERCENTS COUNTS
```

CELL.WIDTH nn

specifies the number of columns that are allotted for each cell in the survey. Widths between 5 and 16 are permitted. Initially, the cell width is set to 7 (the column between adjacent cells, where the percent sign prints, is included in that count). The formatting of labels for the banner points depends on the defined CELL.WIDTH. When values are too large to print in the defined cell width, they are divided by 1000. That number is rounded and printed followed by the letter K.

CHARACTER BREAK 'c' 'c' 'c'

A break character other than the assumed one (the asterisk) may be supplied for value labels. In this example, labels are broken (if necessary) where a question mark is found in the labels file:

```
CHARACTER FOR BREAK '?' '=' ,
```

The question mark does not print in the label. A second *no-break* character (an equal-sign here) may be supplied. It is used in labels to ensure that they do not break at blanks and to center or justify the labels. The equal-sign prints as a blank in labels. (Note that if a label does not fit, it will break nonetheless.) CHARACTER may be abbreviated to CHAR. The word FOR is optional.

BREAK and NO BREAK are usually applied only to the value labels. However, if there is a third break character, the BREAK and NO BREAK rules are also applied to the extended variable labels for the banner variables using the second character for NO BREAK and the third character for BREAK.

Other characters which may be changed, such as the character that represents an empty cell and the character that represents a percent sign, are discussed in a later chapter.

ECHO

Echo subcommands. NO ECHO turns subcommand echo off.

EXCLUDE MISSING MEANS VALUES

is the default setting. When means, medians or sums are calculated for a variable that is not the stub variable, the case is not included in the table if that vairiable has a missing value.

INCLUDE MISSING MEANS VALUES

specifies that cases with missing on the means/medians/sums variable are to be included in the table. The counts of missing in such a table include those cases that are missing on either the stub or the means variable.

GAP nn

indicates the number of spaces to be placed between the banner *variables* (not between the banner points or values of a single variable). The number may range from 0 to 10.

LAYOUT arg arg

specifies the order of the parts in the survey. The character string arguments are keywords to include in the survey in the specified order. The assumed order is:

```
LAYOUT   LABELS      TOTALS.AREA   QUESTION    BODY
         SUBTOTALS   MISSINGS      SUMMARY     F.CHI,
```

Any order may be requested, and any part of the layout omitted from the list is omitted from the survey. Individual parts of the display may also be omitted by using NO followed by the part to be omitted:

MARGIN nn

specifies the width of the area at the left of the survey that contains the stub value labels. The labels fold across as many lines as necessary according to this setting. When MARGIN is not used, the width of the margin is determined by the length of the stub labels, and the margin may vary from one set of surveys to the next. MARGIN may be from 8 to 80 characters wide. There must be at least 12 spaces for the body of the table.

MEANS

requests that the means of the stub variables be included in the summary sections of the table. This is assumed. NO MEANS removes the means from the summary and resets the value of the means/median/sums variable to the current stub variable.

MEANS vn vn

requests that the means in the summary section of the table be based on a variable other than the stub variable. If a single variable name is supplied, that variable is used for the means in all the tables in the current tables group. If a list is supplied, there must be the same number of variables in the means list as there are in the STUB variable list. MEDIAN can be used in a similar fashion.

NO arg

turns off the subcommand given as its argument. All parts of the LAYOUT, as well as the BASE, MEANS, STANDARD.DEV and ROW.TOTALS may be specifically turned off by preceding them with NO. NO can also be used with most of the keywords covered in the following chapters.

OUTPUT.WIDTH nn

specifies the output width of the terminal or page. Widths between 60 and 1200 are permitted. OW may be used as an abbreviation. The PR subcommand sets an output width according to PRINT.PARAMETER settings. Therefore, when PR and OUTPUT.WIDTH are both used as subcommands, the OUTPUT.WIDTH subcommand must follow the PR subcommand to have any effect.

PERCENTS

PERCENTS is preceded by one or more of:

```
ROW    COLUMN   TOTAL
```

and may be followed by "BASED" and one of the following:

```
GOOD.N    TOTAL.N   RESPONSES
```

Additional options for PERCENTS are described in the chapters "SURVEY: The Statistics".

PR fn

requests that the current survey print on some device other than the current output device. For example, this subcommand prints a single table on an attached PC/windows printer:

```
STUB  Question5,  BAN  Age,  PR 'LPT1';
```

(The filename following PR must be associated with a specific printer or else the survey is printed in a diskfile of that name.) PR is an abbreviation for PRINTER. PR sets an appropriate output width for the particular printer.

ROW.TOTALS arg

controls the position of the row totals in the survey. Possible arguments are ON RIGHT or ON LEFT. ON LEFT is the assumed setting. Row totals print on each page of the display unless the word ONLY is also used. ONLY causes the row totals to print only once, as either the first banner point on the first page or the last banner point on the last page. This causes row totals to appear on the left side of the display for the first page only:

```
ROW.TOTALS ON LEFT ONLY,
```

NO ROW.TOTALS is used to remove the column of row totals from the table. ROW.TOTALS ON can be used to turn the row totals on. This will also force the row totals to print when there is only one column in the table.

SHOW VARIABLES

lists the variables in the file. HELP may also be used.

SKIP nn

controls the blank lines in the body of the table. NO SKIP is the same as SKIP 0. SKIP 2 requests a blank line after every second stub value. NEVER SKIP can be used to prevent any blank lines from printing. This is seldom useful unless the table is to be post-processed by another program.

STANDARD.DEV

requests that the standard deviation of the stub values print in the SUMMARY. This is assumed. NO STANDARD.DEV removes it from the summary section.

SUMS

NO SUMS removes the sums from the summary.

SUMS vn vn

requests that the sums in the summary section of the table be based on a variable other than the stub variable. If a single variable name is supplied, that variable is used for the sums in all the tables in the current tables group. If a list is supplied, there must be the same number of variables in the sums list as there are in the STUB variable list. SUMS and MEANS cannot be done on different variables. The difference between the use of MEANS and SUMS is that SUMS automatically adds the total to the list of statistics in the summary section. NO MEANS is used to reset the SUMS as well as the MEANS variable.

TABLE.TITLES 'cs' 'cs'

supplies one or more lines of text as a title for the next table. The title is used for a single table only. Any two lines of titling can be joined by using either of the special join strings "&&" and "& &" as separators. Up to 9 strings can be supplied in each TABLE.TITLE instruction.. Each TABLE.TITLE precedes the associated STUB command.

```
TABLE.TITLES 'This is the first part of the titles' '& &'
             'It is joined to the second line by a single blank',
STUB nextvar,
```

TITLES 'cs'

defines titles directly within the SURVEY command. Up to nine top titles and three bottom titles may be supplied. Each line of titling must be preceded by a TITLES subcommand. The top center title is assumed if the title text is supplied without arguments:

```
TITLES 'This is the first title' ;
```

Left and right titles may also be specified:

```
TITLE  T3  RIGHT  'Third Right Title',
TITLE  B1  LEFT   'Bottom Left Title' ;
```

The TITLES *command* may be used prior to the SURVEY command to define a number of titles that do not change between surveys. Such titles remain in effect after the SURVEY command has completed. Use the TITLES *identifier* in the SURVEY command to turn on titles defined in the TITLES command.

Titles may be changed or added for specific displays within SURVEY using the TITLES *subcommand.* Titles entered within SURVEY are in effect only for the duration of the SURVEY command. The maximum length for a title entered as a subcommand is 78 characters.

TITLES may also be SUSPENDed or set to BLANK in the SURVEY subcommands.

WEIGHT

WEIGHT and NO WEIGHT can be used to turn weighting on and off if the WEIGHT identifier has been used to specify the weight variable.

2
SURVEY: More About the Rows

The previous chapter discusses simple column and row arrangements and provides the basic information needed to use the SURVEY command successfully. This chapter describes more complex row arrangements and provides a complete description of multiple response variables as they are handled by the SURVEY command.

2.1 STUB VARIABLES DEFINE THE ROWS

The variables describing the rows of the table are referred to as the "STUB" variables. The simple stubs described in the previous chapter are printed one variable per page. However, it is possible to:

1. provide a range for stub variables to get a subset or to ensure consistent formatting across tables
2. rank or order the rows: low to high or high to low
3. use the labels file to controls the order of the rows.
4. nest stubs one and two levels deep
5. print several tables on a page or group them together so they share labels and totals
6. process a multiple response variable
7. accumulate totals rather than the frequencies
8. calculate subtotals and nets
9. automatically combine rows with frequencies below a threshold.

This chapter covers the first 7 topics. Subtotals, nets and dynamic combines have a chapter of their own.

2.2 Supplying Ranges

The rows of the table that are usually printed are those rows that represent at least one case in the data. Any row that has no good values is omitted. If a variable has an observed low value of 2, an observed high of 5, and there are no cases with the value 3, the table has 3 rows for the values 2, 4, and 5. Thus the empty rows are "squeezed out". The use of FILL or FILL ROWS causes the empty rows to be included in the table.

Sometimes you may want all values from the low to the high to print even if a row total is zero. Other times you may want a range that is larger than the observed range so that the table can be more easily compared with other tables. In addition, there may be times when you want a table that represents a subset of the data.

The observed range can be over-ruled by adding the desired low and high values in parentheses after the variable name.

```
STUB Age ( 1 5 )
```

Supplying a range is not enough to ensure that all possible rows are displayed in the table. You must also use FILL. FILL, by itself without ranges, requests that all values between the *observed* low and the *observed* high be included in the table. When FILL is used *and* ranges are provided all values (and only the values) between the *supplied* low and high are included in the table.

The supplied range applies not to a particular table but to the variable and it stays in effect until there is another range provided for that variable. A variable can have only one supplied range within a single tables group.

```
SURVEY Paceset,  LABELS 'Paceset.lab';
```

```
BANNER  Own Sex,  STUB Num.Tv ( 1 3 );
STUB Num.TV ( 4 5 );
```

The example above produces two tables: One for those respondents with fewer television sets and one for those with more television sets. The following example also produces two tables but they are identical with a low value of 4 and a high value of 5 because there is only a comma between the subcommands. A comma ends a given subcommand but unlike a semi-colon does not complete the definitions for the current group of tables. Thus the values in the second range replace the values in the first range before the first table prints.

```
STUB Num.TV ( 1 3 ),  STUB Num.TV ( 4 5 );
```

If there are many tables to be produced using the same subset, the P-STAT programming language may be used. This is more efficient because the cases that are not part of the subset are not processed by the SURVEY command.

```
SURVEY Paceset [ IF Num.TV LE 3, RETAIN ],
```

Figure 2.1 Limit Ranges but Retain the Total Counts

```
SURVEY Paceset, LABELS 'paceset.lab';
BAN Age Sex, STUB Num.tv ( 1 3 ),
INCLUDE ALL STUB VALUES;
```

		Age			Sex	
	Total Sample	Under 30	30 to 50	Over 50	Male	Female
Total	80	28	27	23	39	40
Sample	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Number of television sets in your home:						
1	6	4	1	1	4	2
	7.5%	14.3%	3.7%	4.3%	10.3%	5.0%
2	22	14	7	1	18	4
	27.5%	50.0%	25.9%	4.3%	46.2%	10.0%
3	18	8	5	4	11	7
	22.5%	28.6%	18.5%	17.4%	28.2%	17.5%
Missing	-	-	-	-	-	-
Base	80	28	27	23	39	40
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Mean	3.09	2.32	3.22	3.83	2.54	3.60
S.D.	1.13	0.90	1.01	0.98	1.00	1.01

When a range is given, all cases outside of that range are usually dropped from the table. The subcommand INCLUDE ALL STUB VALUES can be used to maintain the total counts and the count of non-missing cases even though the body of the table reflects only those rows within the range. This is illustrated in Figure 2.1. EXCLUDE SOME STUB VALUES can be used to restore the setting.

If TO is used to provide a list of stub variables there is no direct way to apply the range to all of them.

```
STUB Q33 ( 1 5 ) TO Q55 ( 1 5 ),
```

provides a range for just the two variables Q33 and Q55. Adding the subcommand “SHARED RANGES” immediately after the list has the desired effect

```
STUB Q33 ( 1 5 ) TO Q55, SHARED RANGES,
```

When SHARED RANGES is used, the rows that actually appear depend on the current FILL and SQUEEZE settings. SHARED RANGES applies only to a single list of stub variables and may be used more than once in a tables group. It should immediately follow the STUB subcommand to which it applies.

```
STUB Q33 ( 1 5 ) TO Q55, SHARED RANGES,
STUB Q82 TO Q94 Q103 ( 1 3 ), SHARED RANGES,
STUB Q101 Q115,  FILL ;
```

2.3 Rank and Order

Usually the rows of the table are presented in order starting with the row that has the lowest value and ending with the row that has the highest value. The order can be reversed by using the ORDER DOWN subcommand. Figure 2.2 shows both the command and the output when ORDER DOWN is used with the Paceset file. ORDER UP is assumed if neither UP nor DOWN is supplied.

Figure 2.2 Reversing the Order of the Rows

```
SURVEY Paceset,  LABELS 'Paceset.lab';
   STUB AGE,   BANNER Own,   NO SKIP,   NO SUMMARY,
   ORDER DOWN ;
$
                        Respondent
                        ownership

               Total
              Sample     no    yes

Total            80     11     69
Sample        100.0% 100.0% 100.0%

Age

Over 50          23      3     20
                28.7%  27.3%  29.0%
30 to 50         27      6     21
                33.8%  54.5%  30.4%
Under 30         28      2     26
                35.0%  18.2%  37.7%

Missing           2      -      2
                 2.5%           2.9%
```

RANK is used to arrange the rows in frequency order. If the ranking is downwards, the row with the largest value on the row totals is first. The row with the smallest value on the row totals is usually last. If the ranking is upwards, this ordering is reversed.

```
RANK,                  is the same as
RANK ROWS DOWN,
RANK UP,               is the same as
RANK ROWS UP,
```

DOWN is the assumed setting for RANK if neither UP nor DOWN is supplied. Figure 2.3 illustrates the SURVEY command and the output when RANK is used.

Figure 2.3 RANK the Rows

```
SURVEY Paceset,   LABELS 'Paceset.lab';
   BAN Own,  STUB Income.Groups,
   NO MISSING,  NO SUMMARY,
   RANK ROWS ;
   $
```

		Respondent ownership	
	Total Sample	no	yes
Total Sample	80	11	69
	100.0%	100.0%	100.0%
Income of respondent			
$30,000 to $60,000	50	9	41
	62.5%	81.8%	59.4%
Over $60,000	20	-	20
	25.0%		29.0%
Under $30,000	2	1	1
	2.5%	9.1%	1.4%

RANK is always based on the values of the row totals unless the RANK.CONTROL subcommand is used to select one of the columns of the table as an alternate. The RANK.CONTROL subcommand requires the name of the banner variable and the value that designates the column which controls the ranking. If the rank control value is not one of the banner variables, the row totals continue to determine the ranks. The value must be within the currently known range for that variable or an error message is printed.

```
BAN  Age  Sex  Income.Groups,  RANK ROWS DOWN,  RANK.CONTROL Age 1,
```

Both the RANK.CONTROL and the RANK subcommands are needed. The RANK subcommand indicates how the ranking is to be done. The RANK.CONTROL subcommand determines the column which provides the basis for the ranking. NO RANK.CONTROL is used to restore the row totals as the rank criteria.

Figure 2.4 contains a table in which the rows are ranked down. The ranking is determined not by the row totals but by the frequencies in the last column, the column for those cases with a value of 2 on the variable Sex.

```
RANK ROWS DOWN,  RANK.CONTROL Sex 2
```

Figure 2.4 RANK.CONTROL: Changing the Basis for Rank

```
SURVEY Paceset,  LABELS 'Paceset.lab';
  BANNER Own Sex,  STUB  Age, NO MISSING, NO SUMMARY, NO SKIP,
  RANK ROWS DOWN,   RANK.CONTROL Sex 2;
$
                          Respondent
                          ownership   ==== Sex ====

                  Total
                 Sample     no    yes   Male Female

   Total            80     11     69     39     40
   Sample        100.0% 100.0% 100.0% 100.0% 100.0%

   Age

   30 to 60         27      6     21     11     16
                  33.8%  54.5%  30.4%  28.2%  40.0%
   Over 60          23      3     20      7     15
                  28.7%  27.3%  29.0%  17.9%  37.5%
   Under 30         28      2     26     20      8
                  35.0%  18.2%  37.7%  51.3%  20.0%
```

2.4 Partial Ranks

When RANK is used, it can be qualified so that only some of the rows are included in the rank process. This is done by following the RANK subcommand with a number (n). If the number is positive, the rank is done on just the first n rows (where n is the positive number). Any rows beyond that number are left in their natural order. If the number is negative, the ranking is done on all but the last n rows.

```
RANK ROWS  7 UP,     Rank the first 7 rows from low to high
RANK ROWS -2 DOWN    Rank all but the last 2 rows from high to low
```

If neither UP nor DOWN is specified, DOWN is assumed.

Consider this hypothetical question about usage of personal computers. With these data, which have representation in all possible rows, there is no difference in the results of the RANK subcommands.

```
Value  Label        Count       RANK 5 DOWN        RANK -1 DOWN

1      PC Brand A       1       Brand C   63       Brand C   63
2      PC Brand B      32       Brand E   45       Brand E   45
3      PC Brand C      63       Brand B   32       Brand B   32
4      PC Brand D       7       Brand D    7       Brand D    7
5      PC Brand E      45       Brand A    1       Brand A    1
6      Macintosh       58       Macintosh 59       Macintosh 59
```

However, if there are no respondents using Brand A, the results are quite different:

```
Value  Label        Count       RANK 5 DOWN        RANK -1 DOWN

1      PC Brand A              Brand C   63       Brand C   63
```

```
2      PC Brand B      32       Macintosh 59       Brand E   45
3      PC Brand C      63       Brand E   45       Brand B   32
4      PC Brand D       7       Brand B   32       Brand D    7
5      PC Brand E      45       Brand D    7       Macintosh 59
6      Macintosh       58
```

In this second example the use of -1 ensures that the last value will not be included in the ranking. There will always be a last row, so RANK with a negative number should only be used when you know that the rows you wish omitted have some representation (in this case, that there is at least 1 Macintosh user). If you have no idea what sort of distribution a variable has, FILL and ranges can also be used.

```
STUB PC ( 1 6 ), FILL,  RANK 5 DOWN,
```

RANK when the stub is a simple structure has few complications. However, when there are subtotals, the decision on how to rank the rows is more complex. This is discussed in the chapter "SURVEY: Subtotals and Nets".

2.5 Using LABELS To Control Rank and Order

The order of the values in the labels file does not matter. Usually the values are entered from low to high as a matter of convenience. It is also easier to check that all the values have labels when they are entered in order. However, if you wish the rows to appear in alphabetical order, you must use either a character variable or the labels file.

The subcommand CONTROL ROW ORDER USING LABELS instructs the SURVEY command to print the rows in the order in which they occur in the labels file. This does not cause rows which have no values to be omitted from the table. They simply appear in their natural order after the selected rows. RANK is not appropriate when a specific order is requested.

The subcommand CONTROL ROW RANK AFTER nnnn, provides a value which is beyond the range of the stub variable. When this control is set, the labels file is examined for the specified value. All of the labels entries which precede the value are displayed in rank order. Any values beyond are placed in normal order (ascending by value). This is especially useful when you wish an "others" category, which may have a large frequency, to appear last. It is also easier to use than RANK -n or RANK DOWN n when you are not quite sure of the actual ranges for a given variable.

```
Pets (1) Cats  (2) dogs (5) horses
     (1000) dummy label
     (3) birds (4) fish /
```

The rows with values 1, 2 and 5 will print in rank order. Rows with values 3 and 4 will appear after the ranked rows in normal order. CONTROL ROW ORDER OFF turns off the row order controls.

The combination of RANK ROWS and CONTROL ROW ORDER USING LABELS appears to be a contradiction and if they are used together the ranking is done first and the labels are used only when there are ties.

2.6 Nested Stub Variables

The nesting of stub variables can be one or two levels deep. Nesting is done by using the keyword WITHIN in the STUB variable list

```
SURVEY Paceset,  LABELS 'Paceset.lab';
STUB Own WITHIN Sex,  BAN Income.Groups.
```

Figure 2.5 shows a nested STUB variable. The use of SKIP 3 causes a blank line to be printed between every third stub value in the body of the table. Since the inner nest level has two values this causes a visual boundary between the outer levels This works only as long as there are no completely empty rows in the inner level which would be squeezed out.

SKIP will have unexpected results when the inner nest levels do not contain the same number of values. In file Paceset there are no males in the lowest income group. Therefore, when the variable Income.groups is nested within Sex there are only 2 stub values for the males while there are 3 stub values for the females. SKIP 4 causes the blank to be placed inappropriately after the subtotals for the females. This can be corrected by providing ranges for variable Income.Groups and using the FILL subcommand.

The STUBS can be nested 2 deep

```
STUB Sex WITHIN Age WITHIN Own,
```

In a double nested table the middle level is indented 2 columns from the left and the inner level labels are indented 4 columns from the left. This indenting helps define the structure of the table.

When tables are nested and double nested they rapidly overflow to additional pages. As a rule it is easier to follow a table that is contained in a single page and the use of NO SKIP or SKIP n (where n is the number of levels in the innermost within plus the number of nest levels) is very useful. When the SKIP is other than 0 with a nested table, the use of FILL may be necessary to ensure that the blank lines are appropriately placed.

Figure 2.5 Nested Variables in the Rows of the Table

```
SURVEY Paceset, LABELS 'Paceset.lab';
   BANNER Income.Groups,  STUB Own WITHIN Sex,
   LAYOUT LABELS TOTAL QUESTION BODY,  CELL.WIDTH 9,
   SKIP 3;  $
```

		== Income of respondent ==		
	Total Sample	Under $30,000	$30,000 to $60,000	Over $60,000
Total Sample	80 100.0%	2 100.0%	50 100.0%	20 100.0%
Respondent ownership of electronic items				
Male	39 48.8%	-	21 42.0%	10 50.0%
no	5 6.2%	-	4 8.0%	-
yes	34 42.5%	-	17 34.0%	10 50.0%
Female	40 50.0%	2 100.0%	28 56.0%	10 50.0%
no	6 7.5%	1 50.0%	5 10.0%	-
yes	34 42.5%	1 50.0%	23 46.0%	10 50.0%

2.7 MULTIPLE TABLES PER PAGE

There are three ways to get multiple tables on a page. The first is to use the SVAR.PER.PAGE subcommand which specifies the maximum number of stub variables that may be printed on each page. The second way is to

use the GROUP.STUBS command which takes several stub variables and permits them to share the labels and totals sections. The third way is to use PostScript output which permits you to place several small tables at specific locations on a page. PostScript is described in the chapter “SURVEY: Enhancing Layout and Appearance”.

Figure 2.6 Two Tables on a Single Page

```
SURVEY Paceset, LABELS 'Paceset.lab';
   BAN Age Income.Groups,  STUB Own Sex,
   LAYOUT QUESTION LABELS TOTAL BODY,
   MARGIN 9,  GAP 2,  CELL.WIDTH 8, SVAR.PER.PAGE 2;
$

Respondent ownership of electronic items

                                                ====== Income of  =====
                        ========= Age =========   ====== respondent =====

                                                          $30,000
              Total     Under   30 to              Under      to    Over
             Sample        30      50 Over 50   $30,000 $60,000 $60,000

Total           80        28      27      23         2      50      20
Sample       100.0%    100.0%  100.0%  100.0%    100.0%  100.0%  100.0%

no              11         2       6       3         1       9       -
             13.8%      7.1%   22.2%   13.0%     50.0%   18.0%

yes             69        26      21      20         1      41      20
             86.2%     92.9%   77.8%   87.0%     50.0%   82.0%  100.0%


Sex
                                                ====== Income of  =====
                        ========= Age =========   ====== respondent =====

                                                          $30,000
              Total     Under   30 to              Under      to    Over
             Sample        30      50 Over 50   $30,000 $60,000 $60,000

Total           80        28      27      23         2      50      20
Sample       100.0%    100.0%  100.0%  100.0%    100.0%  100.0%  100.0%

Male            39        20      11       7         -      21      10
             48.8%     71.4%   40.7%   30.4%             42.0%   50.0%

Female          40         8      16      15         2      28      10
             50.0%     28.6%   59.3%   65.2%    100.0%   56.0%   50.0%
```

2.8 Placing Several Stub (Row) Variables on a page.

The assumption is that each stub variable defines a printed page.

```
      SVAR.PER.PAGE 3,
```

requests that the SURVEY command try to fit 3 tables on each page. Figure 2.6 shows the command and the resulting printout when two tables per pages are requested. Each table is complete with all the requested sections of the layout. The only shared elements are titles whether defined with the TITLES command or with subcommands in SURVEY.

If the tables do not all fit on a page, they are continued on following pages. The SURVEY command will not break up a low level print unit (the lines representing the frequencies and percents for a given row, for example). Other than that, however, the break across pages can occur at any point in the table. See the chapter "SURVEY: Special Features of the Layout Sections" for more information on controlling page changes.

Figure 2.7 Grouping the Variables Together on the Page

```
SURVEY Paceset, LABELS 'Paceset.lab';
   BAN Age Income.Groups,  GR.STUBS Sex Own,
   LAYOUT LABELS TOTAL QUESTION BODY,
   MARGIN 9,  GAP 2,  CELL.WIDTH 8;
$
```

		========= Age =========			====== Income of ===== ====== respondent =====		
	Total Sample	Under 30	30 to 50	Over 50	Under $20,000	$20,000 to $40,000	Over $40,000
Total	80	28	27	23	2	50	20
Sample	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Sex							
Male	39	20	11	7	-	21	10
	48.8%	71.4%	40.7%	30.4%		42.0%	50.0%
Female	40	8	16	15	2	28	10
	50.0%	28.6%	59.3%	65.2%	100.0%	56.0%	50.0%
Respondent ownership of electronic items							
no	11	2	6	3	1	9	-
	13.8%	7.1%	22.2%	13.0%	50.0%	18.0%	
yes	69	26	21	20	1	41	20
	86.2%	92.9%	77.8%	87.0%	50.0%	82.0%	100.0%

2.9 GROUP.STUBS: Grouping Variables Together

GROUP.STUBS (GR.STUBS) is a subcommand that operates much like the STUB subcommand. The arguments are the names of two or more variables, specified singly or with variable lists. STUBS and GR.STUBS can be used together in a single tables group as long as they have the same banner variables, layout and other options. The only difference is in the final presentation. The GROUP.STUBS variables have a single labels and totals area section for all the variables in the list.

Several GROUP.STUBS groups may be defined for a single set of banners. In this example, the variables Item1, Item2 and Item3 are displayed together:

```
BANNERS  Age  Region,  GROUP.STUBS  Item1  Item2  Item3,
                      STUB Item4 Item5,
                      GROUP.STUBS  Item6  Item7  Item8;
```

Item4 and Item5 are each printed separately and Item6, Item7 and Item8 are then displayed together. Because the subcommands are defined with no semicolon (;) until the very end, they are processed in the same pass of the data. Note: If all of the variables do not fit on a page as requested, they are continued on subsequent pages.

GROUP.STUBS has a special feature for labelling the rows when there is no body and no question. The question is combined with the string that identifies the first statistic in the summary section. This permits a table of means like the one illustrated in Figure 2.8. If either BODY or QUESTION is part of the LAYOUT, the statistics in the summary section are labelled only with the statistic identification.

Figure 2.8 GROUP.STUBS in a Table of Means

```
SURVEY Paceset [ GEN Number.Owned = SUM ( VCR to Ans.Mach )],
     LABELS Paceset.lab;

     BAN Sex Age, GR.STUBS  Income  Num.TV  Number.Owned,
     LAYOUT  LABELS TOTAL SUMMARY,  NO BASE,   NO S.D,
     MARGIN 25;
     $
```

		==== Sex ====		======== Age =======		
	Total Sample	Male	Female	Under 30	30 to 50	Over 50
Total Sample	80	39	40	28	27	23
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Mean Income of respondent	34398	36759	32634	37388	29311	35580
Mean Number of television sets in your home:	3.09	2.54	3.60	2.32	3.22	3.83
Mean Number.Owned	1.81	2.05	1.60	2.21	1.70	1.48

It is possible to replace these standard statistic names with names of your choice. This is described in the chapter “SURVEY: Enhancing Table Appearance”. There is one further control for the labelling of these GR.STUB summary tables. GR.IDENTIFY specifies the amount of identification that is desired.

```
GR.IDENTIFY 1      assumed, both the statistic and variable name
                   or extended labels are used
GR.IDENTIFY 2      just the variable name or extended label
GR.IDENTIFY 3      just the statistic.
```

Figure 2.8 also illustrates some of the automatic formatting that is done. The values for Income are so large that they cannot print with a decimal place and a fractional part without overlapping. The SURVEY command automatically recognizes this and eliminates the fractional part of the mean when necessary. In addition if the number is still too large to fit in the current cell width, it is divided by 1000. The resulting number is printed with a “K” on the right. For example:

```
3435K     represents 3,435,000
```

See the chapter “SURVEY: Enhancing Table Appearance” for formatting options such as COMMAS.

TABLE.TITLES are often used with GR.STUBS when the question/extended label is used to label the summary statistic. The variables that are grouped together in this way are considered to be a single table.

```
TABLE.TITLE “Questions about Political Preference”,
GROUP.STUBS  Item1  Item2  Item3,
```

Figure 2.9 Ranking a Table of Means

```
SURVEY Paceset [ GEN Number.Owned = SUM ( VCR to Ans.Mach )],
     LABELS Paceset.lab;

     BAN Sex Age, GR.STUBS  Income  Num.TV  Number.Owned,
     LAYOUT  LABELS TOTAL SUMMARY,  NO BASE,   NO S.D,
     RANK SUMMARY UP,
     MARGIN 25;
     $
```

		==== Sex ====		======== Age =======		
	Total Sample	Male	Female	Under 30	30 to 50	Over 50
Total Sample	80	39	40	28	27	23
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Mean Number.Owned	1.81	2.05	1.60	2.21	1.70	1.48
Mean Number of television sets in your home:	3.09	2.54	3.60	2.32	3.22	3.83
Mean Income	34398	36759	32634	37388	29311	35580

Tables of means are often done using many related variables. Usually the variables in the table are entered in the order that they occur in the input data file. This can be altered by using the RANK SUMMARY subcommand. This subcommand is only used when GR.STUBS is used and when there is a summary section and no body section. The summary section is not limited to means but can contain any of the summary section statistics. The summary section is described in detail in the chapter “SURVEY: Special Features of the Layout Sections:.

Usually the ranking of the rows in the summary section is done using the means. It can be either ascending or descending:

```
RANK SUMMARY DOWN,    or
RANK SUMMARY UP,
```

DOWN is the default if the direction is not specified.

The statistic that determines the ranking is usually the means but can be the good n. Thus

```
RANK SUMMARY,                      is the same as
RANK SUMMARY DOWN USING MEANS,     and
RANK SUMMARY USING GOOD.N,         is the same as
RANK SUMMARY DOWN USING GOOD.N
```

Figure 2.9 illustrates the input and output when RANK SUMMARY is used. NO RANK turns all ranking off.

Figure 2.10 Multiple Response STUB Variable

```
SURVEY Paceset,  LABELS 'Paceset.lab';
   MR.STUB  phone to Ereader,   BAN Age Sex,   GAP 2;
$
```

	Total Sample	Age: Under 30	Age: 30 to 50	Age: Over 50	Sex: Male	Sex: Female
Total Sample	80	28	27	23	39	40
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Which of the following do you own?						
Smart phone as primary telephone	45	19	13	12	27	18
	56.2%	67.9%	48.1%	52.2%	69.2%	45.0%
One or more landline telephones	28	10	13	5	12	16
	35.0%	35.7%	48.1%	21.7%	30.8%	40.0%
One or more tablet computers	35	15	8	10	21	14
	43.8%	53.6%	29.6%	43.5%	53.8%	35.0%
One or more Ereaders	37	18	12	7	20	16
	46.2%	64.3%	44.4%	30.4%	51.3%	40.0%
Missing	11	2	6	3	5	6
	13.8%	7.1%	22.2%	13.0%	12.8%	15.0%
Base	69	26	21	20	34	34
	86.2%	92.9%	77.8%	87.0%	87.2%	85.0%
Responses	145	62	46	34	80	64

2.10 Multiple Response Variables

Multiple response variables occur whenever a respondent is given the chance to select more than one item or make several replies to a question. For example: in the Paceset file there are many people who own more than one of the four electronic items. One approach is to ask 4 separate questions in the form:

```
Do you own an Ereader?
```

Another approach is the use of a single question allowing multiple responses. For example:

```
Which of the following do you own? Check all that apply:
__   Smart phone as primary telephone
__   One or more landline telephones
__   a tablet computer
__   an Ereader
```

In either case the responses are entered as four separate variables. The values are coded as 1 if the item is owned and 0 (or left blank) if the item is not owned.

When the multiple response variables are coded as 0 and 1 (or 1 and missing) they are often referred to as "dummy" variables. If the question covers a wide range of possible responses, individual variables for each response may not be practical. The question about ownership for the Paceset data might have included a very great number of household electronic items including electric tooth brushes, can openers, hair dryers and pencil sharpeners. In figure 2.11 the sub-command SET.LABELS is used to change the multiple response stub labels to the short TAGS format thus saving an output line. An alternate way would use "MARGIN 30" to widen the left margin.

Figure 2.11 Percents Based on Responses

```
SURVEY Paceset,  LABELS 'Paceset.lab';
  SET.LABELS MR.STUBS TAGS,
  MR.STUB  Phone to Ereader,   BAN Age Sex,   GAP 2,
  PERCENTS BASED RESPONSES;
$
```

		Age			Sex	
	Total Sample	Under 30	30 to 50	Over 50	Male	Female
Total Sample	80	28	27	23	39	40
Which of the following do you own?						
Phone	45	19	13	12	27	18
	31.0%	30.6%	28.3%	35.3%	33.8%	28.1%
Lines	28	10	13	5	12	16
	19.3%	16.1%	28.3%	14.7%	15.0%	25.0%
Tablet	35	15	8	10	21	14
	24.1%	24.2%	17.4%	29.4%	26.2%	21.9%
Ereader	37	18	12	7	20	16
	25.5%	29.0%	26.1%	20.6%	25.0%	25.0%
Missing	11	2	6	3	5	6
Base	69	26	21	20	34	34
Responses	145	62	46	34	80	64
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%

When there are a only a few possible responses it makes very little difference how these questions are asked or how the responses are handled when the data are entered into a P-STAT system file. When there is a large number of possible responses it often makes sense to determine the maximum number of responses for a given question and build the file with that number of variables. For example, if it is determined that no respondent has checked more than four items, the variables might be named Item.1 to Item.4. Instead of values of 0 and 1, the values will range from 1 to the number of possible items in the list. We refer to this type of coding as 1/n coding.

Dummy variable coding is currently used for the 4 ownership variables in file Paceset. The labels and the values for the first case in the file are:

```
Phone Lines Table Ereader

   1     0     0       1
```

If 1/n coding had been used to create 4 ownership variables, the resulting labels and data might have been:

```
 Own  Own   Own  Own
  .1   .2    .3   .4

   1    4     -    -
```

1/n coding is also used for questions such as:

```
Select the 5 most important items from the following list:
Which is the most important item in the following list?
Which is the second most important item in the list?
```

When the stub (row) variable is actually several variables designed to be handled as one, the STUB subcommand is replaced by the MR.STUB subcommand. This is true whether the variables are dummy variables or contain 1/n type codes. Like STUB and GROUP.STUB, the subcommand is followed by variable names or variable lists. If individual variable names are used, the variables need not be adjacent in the file. For example:

```
MR.STUB  Q3A Q3C Q3F TO Q3J,
```

There may be several MR.STUB subcommands in a single tables group and they may be intermixed with STUB and GR.STUB subcommands. However, all of the variables in a single MR.STUBS subcommand must be either all dummy variables or all 1/n coded variables.

```
BANNER Age,
STUB Income.Groups, MR.STUB Phone to Ereader, GR.STUB Own Sex;
```

The SURVEY command needs to know which type of coding is used in the multiple response variables. If the variables are dummy variables, there will be one row in the table for each variable. If 1/n coding is used, there will be 1 row in the table for each unique value that occurs in the list of variables.

Usually the SURVEY command can determine the coding type by examining the observed values. If there are more than 2 values, 1/n coding is indicated. If there is only 1 value, dummy variable coding is indicated. If there are 2 values, dummy coding is used and the higher of the values is assumed to be the one of interest. This can be controlled by following MR.STUBS with instructions:

```
MR.STUBS ( COUNT 0   )  Q2 TO Q10,    type: dummy  count the zeros
MR.STUBS ( COUNT ALL )  Q2 TO Q10,    type:  1/n   count all values
MR.STUBS ( COUNT 2   )  Q2 TO Q10,    type: dummy  count the twos
```

COUNT ALL is only required when there are just 2 values and this is not a dummy variable situation.

The labelling conditions for the two types of multiple response stubs are quite different. When there are dummy variables, each variable represents a value. Therefore, the variable label or its extended label is used as the value label. When the number of values bears no relation to the number of variables, the labels for the rows are the value labels of the first variable in the MR.STUBS list.

The labels in the previous chapter for Paceset.lab are appropriate for a dummy variable situation.

```
Phone       <<Smart phone as primary telephone>>
            <<Which of the following do you own?>> /
Lines       <<One or more landline telephones>> /
Tablet      <<One or more tablet computers>>  /
Ereader     <<One or more Ereaders>>   /
```

Each of the four variables has an extended label. These are the labels used for the rows in the table in Figure 2.10 which uses the entire label and Figure 2.11 which requests the short TAG format.

Since the extended labels are used to label rows, the convention is that the QUESTION section of the layout contains any extra extended labels for the first variable in the list. If there are no extra extended labels, the QUESTION section is omitted from the table.

If the Paceset data were coded for four variables with many different possible values, the value labels would take the following form:

```
Item1 <<Which of the following do you own?>>
    (1) Smart phones
    (2) Landline telephones
    (3) Tablet computer
    (4) Ereaders
    (5) Electric Tooth Brush   .....
```

Figure 2.12 Counting Another Variable

```
TITLES  T1 LEFT 'Count the number of television sets' $

SURVEY Paceset,  LABELS 'Paceset.lab';
   BANNER Age Sex,
   STUB Own,
   GAP 2,  MARGIN 16,
   NO MISSING,
   COUNT Num.TV,    TOTAL PERCENTS;
```

		======== Age =======			==== Sex ====	
	Total Sample	Under 30	30 to 50	Over 50	Male	Female
Total Sample	247	65	87	88	99	144
	100.0%	26.3%	35.2%	35.6%	40.1%	58.3%
Respondent ownership of electronic items						
no	33	6	14	13	13	20
	13.4%	2.4%	5.7%	5.3%	5.3%	8.1%
yes	214	59	73	75	86	124
	86.6%	23.9%	29.6%	30.4%	34.8%	50.2%
Base Number of	247	65	87	88	99	144
television sets	100.0%	26.3%	35.2%	35.6%	40.1%	58.3%
in your home:						
Mean	3.09	2.32	3.22	3.83	2.54	3.60
S.D.	1.13	0.90	1.01	0.98	1.00	1.01

When MR.STUBS is used, the assumed layout differs from the assumed layout of a simple stub. The summary section contains the base and the number of responses. The base is the total number of respondents who have at least 1 positive reply to any of the multiple response variables. The responses are the total of all the replies.

The missing section reports the number of respondents who had no replies on any of the questions. Of the 80 people who answered the Paceset questionnaire, 69 people own 145 items. The 11 people who are counted in the MISSING section of the table in Figure 2.11 own none of the 4 possible items.

2.11 Printing Totals Rather than Frequencies

Usually the numbers in the cells of a table are either the frequencies or the percents based on frequencies. Sometimes the desired information is not the number of cases which belong in a cell, but the totals of some other variable. There are three different ways that this information can be computed. The numbers in the body of the table are identical for all three. There are differences in the base for percents, the auxiliary statistics which may be requested and the computation of the statistics in the summary section.

1. WEIGHT identifier — Summary statistics are based on the weighted values of the stub variables. Weight can be used with either SUMS or COUNTS.
2. SUMS subcommand — Percents are based on the number of respondents. Both the sums and the means can be included in the cells of the table. SUMS and COUNT cannot be used together.
3. COUNT subcommand — Both percents and summary statistics are based on the counted variable rather than the stub variable. Means, medians and sums in the cell are not permitted.

In Figure 2.12 the COUNT subcommand requests a table in which the information is the count (total number) of television sets that are owned by each of the respondents rather than the number of respondents who fall in each cell. Percents are based on the total number of television sets requested rather than on the column totals usually produced:

```
TOTAL PERCENTS,
```

Thus, if we look at the males who own none of the 4 electronic items, we can see that they own 5.3% of the 247 television sets. Females who own at least 1 electronic item own 50.2% of the 247 television sets.

The COUNT subcommand must always be used with a STUB, GR.STUB or MR.STUB subcommand. Usually the COUNT subcommand is followed by the name of a single variable. However, it can be followed by a list of variables. If there is a list of COUNT variables, each COUNT variable is paired with a STUB variable. If there are fewer COUNT variables than STUB variables, the final COUNT variable is used for the remaining STUB variables.

```
STUB  Manufacturer  Item  Department,    COUNT  Sales  Units,
```

This example has three stub variables and two counts variables. The first table has Manufacturer as the stub with cells representing totals of variable Sales. The second table has Item as the stub variable with cells representing totals of variable Units. The third table, which has Department as the stub, also has cells representing Units.

Figure 2.13 shows the label and data files for a survey about dog food. These data are used in Figures 2.14 to 2.16 People were asked the brands of dog food that they fed their dog during the past month. They could respond with up to four brands of dog food. The answers are in the variables Food1 to Food4.

Figure 2.14 is the typical multiple response survey (MR.STUBS) showing counts of the values observed for the Food1 to Food4 variables. The values are the various brands of dog food. Using RANK positions the brands in order of the number of times they were cited. Basing the percentages on the number of responses,

```
PERCENTS BASED ON RESPONSES,
```

rather than on the number of respondents (the Total Sample or TOTAL.N) shows the percentage of all answers that cited a given brand. This often makes more sense in a multiple response situation. Thus, we can see that Rivalry and Gravy Wagon were the top two selections of the total sample and of the higher income group, although not of the lower income group. (These conclusions are “tongue-in-cheek” and given just for illustrative purposes – this is a very small sample of fictitious data.)

Figure 2.13 Labels, Data for MR.STUBS and MQ.STUBS

```
LABEL FILE  DogFood.Lab:

 Food1  <<What brands of dogfood did you feed your dog last month?>>
        (1) Mitey Dog      (2) Allpo
        (3) Gravy Wagon    (4) Trim N Neat
        (5) Hardy Boy      (6) Rivalry     /

 Cans1  <<Cans of 1st Choice>>   /
 Cans2  <<Cans of 2nd Choice>>   /
 Cans3  <<Cans of 3rd Choice>>   /
 Cans4  <<Cans of 4th Choice>>   /

 Income <<Family Income>>
        (1) Under $30,000  (2) $30,000 and Up  /



 FILE  DogFood:

 Food1  Food2  Food3  Food4  Cans1  Cans2  Cans3  Cans4  Income

     6      1      3      4     12      5      8      2       2
     2      1      5      6     20      6     11      2       1
     2      4      3      -     30      2      2      -       1
     1      6      3      4     15      5      5      1       2
     2      5      6      3     10     10      1      1       2
     2      3      -      -     25      5      -      -       1
     6      4      3      1     35     20      3      1       2
     5      6      1      2     45     20      5      5       1
```

The respondents were also asked to estimate the number of cans used of each brand that they mentioned. This information is in variables Cans1 to Cans4. Cans1 is the number of cans of the first-choice dog food mentioned in Food1, Cans2 is the number of cans of the second-choice dog food mentioned in Food2, and so on. Income is a background variable.

Figure 2.15 is the same multiple response survey showing sums of the values of corresponding variables (MR.STUBS with COUNT). This is part of the same SURVEY command as the previous table. The sums (in each cell) are the number of cans of each brand used by the respondents. As before, the sums are ranked (the preceding RANK subcommand is still in effect since it was not reset). Now, however, the percentages are based on the Total Sample (the TOTAL.N), which is the total number of cans used:

```
PERCENTS BASED ON TOTAL.N,
```

Figure 2.15 is the same multiple response survey showing sums of the values of corresponding variables (MR.STUBS with COUNT). This is part of the same SURVEY command as the previous table. The sums (in each cell) are the number of cans of each brand used by the respondents. As before, the sums are ranked (the preceding RANK subcommand is still in effect since it was not reset). Now, however, the percentages are based on the Total Sample (the TOTAL.N), which is the total number of cans used:

```
PERCENTS BASED ON TOTAL.N,
```

Thus, we can see that Allpo leads the dog food brands in actual cans used, even though it was not among the two brands mentioned most often by respondents. (Perhaps other factors such as price affect usage.)

Figure 2.14 MR.STUBS: Showing Frequencies

```
SURVEY  DogFood,  LABELS  DogFood.Lab ;

  BANNER  Income,  MR.STUBS  Food1  TO  Food4,  NO MISSING,
  PERCENTS  BASED  ON  RESPONSES,  RANK,  CELL.WIDTH  10,
  TITLE  '** Dog Foods CHOSEN Most Often **' ;
```

```
** Dog Foods CHOSEN Most Often **

                    == Family Income ==

                Total     Under   $30,000
               Sample   $30,000    and Up

Total
Sample              8         4         4

What brands of dogfood did you feed your dog last month?

Rivalry             6         2         4
                 20.7%     15.4%     25.0%

Gravy Wagon         6         2         4
                 20.7%     15.4%     25.0%

Allpo               5         4         1
                 17.2%     30.8%      6.3%

Mitey Dog           5         2         3
                 17.2%     15.4%     18.8%

Trim N Neat         4         1         3
                 13.8%      7.7%     18.8%
Hardy Boy           3         2         1
                 10.3%     15.4%      6.3%

Base                8         4         4
Responses          29        13        16
                100.0%    100.0%    100.0%
```

2.12 MQ.STUBS: Multiple Quantity Stub Variables

Figure 2.16 illustrates MQ.STUBS (multiple quantity stubs), a variant of the typical multiple response survey. The values of MQ.STUBS variables are quantities, rather than choices of responses. The values of the variables Cans1 to Cans4 are quantities and not selections (whereas the values of Food1 to Food4 are selections). Thus, MQ.STUBS may be used with Cans1 to Cans4:

```
MQ.STUBS  Cans1  TO  Cans4,
```

Figure 2.15 MR.STUB: Showing Counts

```
      MR.STUB   Food1  TO  Food4,  COUNT  Cans1  TO  Cans4,
      PERCENTS  BASED  ON  TOTAL.N,
      TITLE  '** Dog Foods USED the Most **'  ;

          ** Dog Foods USED the Most **

                           == Family Income ==

                    Total     Under   $30,000
                   Sample   $30,000    and Up

 Total               312       178       134
 Sample            100.0%    100.0%    100.0%

 What brands of dogfood did you feed your dog last month?

 Allpo                90        80        10
                    28.8%     44.9%      7.5%

 Rivalry              75        22        53
                    24.0%     12.4%     39.6%

 Hardy Boy            66        56        10
                    21.2%     31.5%      7.5%

 Mitey Dog            32        11        21
                    10.3%      6.2%     15.7%

 Trim N Neat          25         2        23
                     8.0%      1.1%     17.2%

 Gravy Wagon          24         7        17
                     7.7%      3.9%     12.7%

 Missing               -         -         -

 Base                  8         4         4
 Responses            29        13        16
```

2.13 MQ.STUBS: Multiple Quantity Stub Variables

Figure 2.16 illustrates MQ.STUBS (multiple quantity stubs), a variant of the typical multiple response survey. The values of MQ.STUBS variables are quantities, rather than choices of responses. The values of the variables Cans1 to Cans4 are quantities and not selections (whereas the values of Food1 to Food4 are selections). Thus, MQ.STUBS may be used with Cans1 to Cans4:

```
      MQ.STUBS  Cans1  TO  Cans4,
```

MQ.STUBS sums all the values of each variable. (Looking back at the data in Figure 2.12, the total of all the cans in variable Cans1 is 192, in Cans2 it is 73, and so on.) Thus, we can see that over 60 percent (61.5 exactly) of the cans of dog food used are the first mentioned choice. Equally interesting, almost 40 percent of the cans used

are second, third and fourth mentioned choices – why are people switching brands? (Additional factors such as cost, desire for variety, packaging convenience, and so on, may influence usage.)

MQ.STUBS is useful when the variables represent a choice and the values represent a quantity. This is the case in the dog food example and, for another example, in a survey that asks the number of cans of various brands of soda consumed in a week. The variables might be Pepsi, Coke, 7-Up, and so on, and the values would be the quantities consumed. MQ.STUBS sums the number of cans consumed for each brand of soda:

```
MQ.STUBS  Pepsi  TO  Dr.Pepper ;
```

The SUMMARY section of an MQ.STUBS survey consists only of a count of the number of Responses — the number of good responses. (See the following chapters for information on more advanced aspects of multiple response variables, including character data, recoding 0/1 to 1/n coding, creating indices, nets, paired variables and cross-comparison tables. Complete information about percents and other statistics and about the interactions of rank with nested tables and nets is also covered in the following chapters.)

Figure 2.16 MQ.STUB: Showing Sums

```
AGAIN,
MQ.STUBS  Cans1  TO  Cans4,
TITLE  '** Why are Owners Switching Brands? **' ;
```

```
** Why are Owners Switching Brands? **

                              == Family Income ==

                      Total       Under    $30,000
                     Sample     $30,000     and Up

Totals                  312         178        134
                      100.0%      100.0%     100.0%

Cans of 1st Choice      192         120         72
                       61.5%       67.4%      53.7%

Cans of 2nd Choice       73          33         40
                       23.4%       18.5%      29.9%

Cans of 3rd Choice       35          18         17
                       11.2%       10.1%      12.7%

Cans of 4th Choice       12           7          5
                        3.8%        3.9%       3.7%

Responses                29          13         16
```

SUMMARY

SURVEY

```
SURVEY Paceset,  LABELS 'Paceset.lab';
    RANK DOWN,
    BAN Age Own,  STUB Num.TV ( 1 5 ),
    MR.STUB Phone to Ereader,
    GR.STUB Store.1 TO Store.4;
```

The SURVEY command has many options for controlling the content and the arrangement of the rows of a table. These include: multiple response variables, variables grouped together as a single table, nesting variables and ranking or ordering the rows in various ways.

Optional Subcommands: Defining the Rows

STUBS vn vn (nn nn)

STUBS is one of the 4 subcommands that can be used to supply the names for the variables which define the rows of the table. If the low and high values are not provided, the observed low and high values are used. Only rows which represent at least 1 case are printed unless the FILL identifier is also used.

GROUP.STUBS vn vn vn

GR.STUBS vn (nn nn) vn vn (nn nn)

GROUP.STUBS provides a list of variables to be used as row variables for separate tables, which, however, share the same labels and the same totals area. Group stubs are printed, when possible, on a single page. Each variable in the list may have its low and high values provided in parentheses following the variable name.

MQ.STUBS vn vn vn

defines multiple response variables in which the desired result is not a count of each unique value (response), but rather a sum of the values. The SUMMARY section of the survey consists of a count of the number of responses.

MR.STUBS vn vn vn

defines a list of variables as a multiple response group. If all the variables in the list have the same two values, it is assumed that this is a dummy (0/1) variable situation and that the higher value is to be counted. If there are more than two unique values, each occurrence of each value is tabulated. This can be controlled by specifying either the value to be counted:

```
MR.STUBS ( COUNT 0 )     Quest.1   TO  Quest.10,
```

or that all values are to be counted:

```
MR.STUBS ( COUNT ALL ) TV.Prog1  TO  TV.Prog3,
```

The *subcommand* COUNT, followed by a list of variables, may be used to indicate variables which are the quantities to be totaled for the corresponding stub variables:

```
MR.STUBS  Magazines1  TO  Magazines4,
COUNT     Copies1     TO  Copies4   ;
```

OPTIONAL Subcommands: Controlling the Row Contents

AGAIN

Again, requests that the previous table be repeated. Usually this is followed by 1 or more subcommands with change requests.

CONTROL ROW ORDER USING LABELS

The rows in the body of the table are to appear in the same order as the corresponding labels in the labels file. CONTROL ROW ORDER OFF is used to turn off row order controls. If this is used in combination with RANK ROWS, the only use for the labels is to determine the row order when there are ties on the rank.

CONTROL ROW RANK AFTER nn

The number "nn" is one which is outside the range of the variable. Any rows with labels which are found before this value in the labels file are to be ranked. Other rows are to be left in natural order. Note: This *must follow* the rank subcommand. CONTROL ROW RANK OFF turns off the control.

COUNT vn vn

requests a summation of the values (responses) of the specified variable rather than a count of the frequency of occurrence of each unique value of the stub variable. COUNT is followed by the variables which contain the quantities to be totaled for the corresponding stub variables:

EXCLUDE SOME STUB VALUES

is the assumed setting when ranges are used for the stub variables. All cases outside the range are dropped from the table before any computation is done.

FILL ROWS

requests that empty rows in the STUBS be printed. The assumption is SQUEEZE.

GR.IDENTIFY nn

The allowed arguments are 1 (assumed), 2, and 3. A 1 requests that the labeling of the summary statistics include both the statistic and the variable or extended label. A 2 requests just the variable or extended label. A 3 requests just the statistic.

INCLUDE ALL STUB VALUES

specifies that all the stub values are to be used in the totals and statistics even when a subset has been specified by using ranges after the stub variable name.

ORDER ROWS DOWN / UP

specifies whether the rows in the table are to be ordered from low value to high (UP) or from high value to low (DOWN). UP is assumed.

RANK ROWS DOWN / UP

requests that the rows in the body of the survey print in descending order of the row totals. RANK ROWS UP may be used to request the opposite – that the rows print in ascending order of the row totals. When RANK is not used, stub variables typically print in ascending order of their values (codes). RANK may be followed by an integer to request that only that many rows of the survey be ranked:

```
RANK ROWS 5  UP,
```

Rows beyond the rank limit are printed in their normal positions. When RANK is followed by a negative integer, the final n rows are excluded from the ranking.

RANK interacts with subtotals and nets. These interactions are discussed in the chapter "SURVEY: Subtotals and Nets".

RANK.CONTROL vn nn

RANK.CONTROL identifies a column other than the row totals which is to be used as the basis for ranking the rows. The required arguments are the name of the variable and the value which is to control the ranking. NO RANK.CONTROL resets the row label as the basis for row ranking. RANK must also be used to direct how the ranking is to be done.

RANK SUMMARY UP / DOWN USING MEANS / GOOD.N

RANK SUMMARY may be used with GR.STUBS and a table that has a summary section but no body. The stub variables in the summary are ranked wither UP or DOWN using either the MEANS or the GOOD.N to determine the order. The default is to rank down using the means.

SHARED RANGES

When a range of variables is given with "TO", it is often desirable to specify that all the variables in the list take on the same low and high values.

```
STUBS Q3 ( 1 5 ) TO Q5 Q47, SHARED RANGES,
```

SHARED RANGES must immediately follow the variable list.

SQUEEZE ROWS

squeeze out empty rows is assumed. FILL ROWS may be used to cause rows with no cases to print.

SVAR.PER.PAGE nn

specifies the number of stub variables to print on one page. Separate totals print for each stub variable. When SVAR.PER.PAGE is not used, one stub variable prints on a page unless multiple stub variables have been defined using GROUP.STUBS.

USE POSITIONS / VALUES

Use specifies how subtotals are to be calculated. POSITIONS is the assumed setting for subtotals. Use VALUES changes this assumption. USE stays in effect until it is explicitly changed or the RESET subcommand is used. The SUBTOTALS subcommand can also be used to specify the basis for computing subtotals. The SUBTOTALS setting is temporary and applies just to that single usage.

SURVEY Labeling Subcommands

SET.LABELS FILE FULL TAGS TEXT

This specifies whether stub and banner labels should be taken from the labels file; use the full up to 64 character variable names; use just the tag portion or just the text portion of any long labels.

SET.LABELS BAN STUBS or other part of the table followed by FILE FULL TAGS or TEXT

SET.LABELS STUB FULL,

SET.LABELS BAN TAGS.

Requests different choices for various parts of the tables. There is less room for banner labels than for stub labels so the combination above may often be useful. For the most part, however, the labels file will continue to be the major source for labels in the SURVEY command

3;;
SURVEY: More About the Columns

BANNER variables define the columns of the table. In the previous chapter the banners all had a simple structure and few values so that they fit neatly on the page. The banner variables can also:

1. be printed 1 or more per page
2. have many values requiring several physical pages
3. be nested by using WITHIN
4. have empty columns suppressed
5. have additional columns representing the good n, the mean, the median and the percents for that variable
6. represent quantities rather than frequencies
7. represent multiple response variables.

Each banner variable defines a group of 1 or more columns. The term “banner point” is a synonym for “column”.

3.1 CONTROLLING THE COLUMNS ON THE PAGE

The SURVEY command tries to fit as many columns on a page as it can without leaving a “widow” (a single column) and without breaking up a banner variable that would fit on a single page if used by itself.

Consider a variable with 12 values. If, given the current margin, the page is wide enough to hold 11 columns, the first page has 10 columns and the second page has 2 columns. If there are 2 banner variables, one with 2 values and one with 12 values, the first page contains the first variable and the first 9 columns of the second variable. The second page has the remaining columns for the second variable. However, if the second variable only has 11 values (i.e., it could fit by itself), each variable will be on a page by itself.

The subcommand BVAR.PER.PAGE can be used to control this behavior. If you wish to maximize the columns per page, use:

```
BVAR.PER.PAGE 0,
```

Setting this control to zero instructs the SURVEY command to put the columns out one after the other without worrying about where a new banner variable starts.

Setting BVAR.PER.PAGE to 1 causes each banner variable to begin on its own page. The setting of the BVAR.PER.PAGE subcommand when it is other than zero provides a maximum value. If there are too many columns to fit, they are continued across pages. A variable may be used more than once in a banner subcommand. For example; if you wish to have variable Own in file Paceset displayed with each of 3 other banner variables, you might use:

```
BANNER Own Age  Own Sex  Own Income.groups,
BVAR.PER.PAGE 2,
```

This produces three pages of output for each stub variable. There are 2 banner variables on each page. Own is repeated three times so that it can be easily compared with each of the other variables. BVAR.PER.PAGE can be followed by a list of numbers:

Figure 3.1 Nested Banners and Squeeze

```
  SURVEY PACESET,  LABELS 'Paceset.lab';
     BANNER SEX  Sex WITHIN Income.groups,   STUB Age,
     NO ROW.TOTALS,  NO MISSING,  NO SUMMARY;

                            ======= Sex / Income of respondent ======

                                                    $30,000
                                      Under           to           Over
                                Sex  $30,000       $60,000      $60,000

                              Male Female   Male Female   Male Female   Male Female

Total                           39     40      -      2     21     28     10     10
Sample                       100.0% 100.0%         100.0% 100.0% 100.0% 100.0% 100.0%

Age

Under 30                        20      8      -      -     10      8     10      -
                             51.3%  20.0%                47.6%  28.6% 100.0%

30 to 50                        11     16      -      2      8     11      -      3
                             28.2%  40.0%         100.0%  38.1%  39.3%         30.0%

Over 50                          7     15      -      -      3      9      -      6
                             17.9%  37.5%                14.3%  32.1%         60.0%

  AGAIN,  SQUEEZE COLUMNS;

                            === Sex / Income of respondent ===

                                      Under    $30,000
                                      $30,       to           Over
                                Sex    000     $60,000      $60,000

                              Male Female Female   Male Female   Male Female

Total                           39     40      2     21     28     10     10
Sample                       100.0% 100.0% 100.0% 100.0% 100.0% 100.0% 100.0%

Age

Under 30                        20      8      -     10      8     10      -
                             51.3%  20.0%         47.6%  28.6% 100.0%

30 to 50                        11     16      2      8     11      -      3
                             28.2%  40.0% 100.0%  38.1%  39.3%         30.0%

Over 50                          7     15      -      3      9      -      6
                             17.9%  37.5%         14.3%  32.1%         60.0%
```

```
BANNER Own Age Education Sex Income.Groups,  BVAR.PER.PAGE 2 3,
```

places the first two variables on one page and the next three on the following page. The settings are not honored if there are too many columns to fit on the page. Extra variables are placed using the default spacing.

The subcommands MARGIN, CELL.WIDTH, GAP, and OUTPUT.WIDTH covered in the chapter "SURVEY:Creating Simple Tables" are all considered in determining the number of columns that are used. The maximum number of columns including row totals that can print on a page is 91.

Figure 3.1 illustrates a combination of a nested banner and SQUEEZE COLUMNS. SQUEEZE in this situation saves space but when it squeezes out a lower nest level, the table may be less attractive. FILL COLUMNS may be used to change this setting back to its original value.

3.2 Supplying Ranges and Nesting Banner Variables

Ranges for the banner variables are supplied in the same way that they are supplied for stub variables, by adding the low and high values in parentheses immediately following the name of the variable.

```
BANNER Num.TV ( 1 6 )  Age  Own,
```

Because the list of banner variables is usually short, there is no provision for applying a single range to a group of variables. The assumption is that empty columns are to be printed unless the SQUEEZE COLUMNS subcommand is used. FILL COLUMNS can be used to change the setting back to the assumed initial setting. While you may use a banner variable more than once in a single table definition, you cannot use it with different ranges. There can be only one low value and one high value for each variable in a group of tables.

```
Ban Num.TV ( 1 2 ) Own  Num.TV ( 3 5 ) Age,
```

causes the low value 3 and the high value 5 to be used twice for the variable Num.TV. The first setting (1 2) is replaced with the second before any tables are actually printed.

Figure 3.2 Reverse Order of Columns

```
ORDER COLUMNS DOWN,
```

		Age			Respondent ownership	
	Total Sample	Over 50	30 to 50	Under 30	yes	no
Total Sample	80	23	27	28	69	11
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Income of respondent						
Under $30,000	2	-	2	-	1	1
	2.5%		7.4%		1.4%	9.1%
$30,000 to $60,000	50	13	19	18	41	9
	62.5%	56.5%	70.4%	64.3%	59.4%	81.8%
Over $60,000	20	6	3	10	20	-
	25.0%	26.1%	11.1%	35.7%	29.0%	

The keyword WITHIN when used with banner variables controls the nesting of one variable within another. However, banner variables *may only be nested one level deep.* If deeper nesting is required, the P-STAT program-

ming language which permits new variables to be created from 2 or more other variables can often be used successfully. The list of variables may include both nested and unnested variables and a variable may be used by itself as well as in a nest situation.

```
BANNER Sex Age WITHIN Sex
```

3.3 Rank and Order

Banner variables can be rearranged so that they are in descending *order* on the column values or so that they are *ranked* in either ascending or descending sort sequence on the column totals. The rank or order is done for each banner variable separately. Figure 3.2 shows the table that is produced when ORDER COLUMNS DOWN is used with variables Age and Own in file Paceset.

The order of the columns can be controlled by the contents of the labels file in the same way that the order of the rows, described in the previous chapter, is controlled.

```
CONTROL COLUMN ORDER USING LABELS,
```

When this subcommand is used, other RANK or ORDER subcommands are inappropriate.

Figure 3.3 shows a table with both rows and columns ranked upwards (ascending). Rows can be ranked while columns are ordered. Columns may be ordered while rows are ranked. The direction may be up for row and down for columns or visa-versa. However, there is no feature for the columns comparable to the RANK.CONTROL subcommand which is available for rows. The following combination of subcommands is allowed:

```
RANK COLUMNS UP,  RANK ROWS DOWN,  RANK.CONTROL Age 1,
```

When either RANK or ORDER is used without specifying ROWS or COLUMNS, ROWS is assumed. This is for conformity with previous releases when these features were available only for the rows. It never hurts and may prevent surprises if you always use the full subcommand and supply the ROW or COLUMN information.

Figure 3.3 Ranking the Columns

```
RANK COLUMNS UP,  RANK ROWS UP,
```

		======== Age =======			Respondent ownership	
	Total Sample	Over 50	30 to 50	Under 30	no	yes
Total Sample	80 100.0%	23 100.0%	27 100.0%	28 100.0%	11 100.0%	69 100.0%
Income of respondent						
Under $30,000	2 2.5%	-	2 7.4%	-	1 9.1%	1 1.4%
Over $60,000	20 25.0%	6 26.1%	3 11.1%	10 35.7%	-	20 29.0%
$30,000 to $60,000	50 62.5%	13 56.5%	19 70.4%	18 64.3%	9 81.8%	41 59.4%

The feature which permits some of the values to be omitted from the rank is available for the columns as well as for the rows.

```
RANK COLUMNS -1 DOWN,
RANK COLUMNS  3 UP,
```

When the number is negative all but the last n columns are included in the ranking. When the number is positive, that many columns are included in the ranking. The ranking occurs before any empty columns are squeezed out and includes all columns for each banner variable from the lowest defined value to the highest defined value. These low and high values are taken from the data unless they are explicitly provided.

RANK for columns can also be controlled by the use of a special dummy label which provides a boundary between the values that should be ranked and those that should not be ranked. For example:

```
CONTROL COLUMN RANK AFTER 1000,
```

RANK UP and RANK DOWN may be used with the CONTROL subcommand to determine the direction of the ranking. The dummy value in the labels file must be a value that will never occur for that variable.

3.4 ADDING SUMMARY INFORMATION TO THE BANNERS

Each banner variable can optionally have means, medians, totals, percents, and (instead of the frequency) a total of the values or quantities. Usually a banner variable has a column for each of the values that are observed for that variable. Variables such as income and age in years are seldom appropriate as banner variables but they are very appropriate when quantities are desired rather than frequencies.

Figure 3.4 Means of a Banner Variable

```
SURVEY Paceset,  LABELS 'Paceset.lab' ;
   BANNER Own ( M )  Age,  STUB Income.group,
   NO MISSING,  NO SUMMARY ;
```

		===== Respondent =====			===== ownership =====	======== Age =======		
	Total Sample	no	yes	Mean	Under 30	30 to 50	Over 50
Total Sample	80	11	69	1.86	28	27	23
	100.0%	100.0%	100.0%		100.0%	100.0%	100.0%
Income of respondent							
Under $30,000	2	1	1	1.50	-	2	-
	2.5%	9.1%	1.4%			7.4%	
$30,000 to $60,000	50	9	41	1.82	18	19	13
	62.5%	81.8%	59.4%		64.3%	70.4%	56.5%
Over $60,000	20	-	20	2.00	10	3	6
	25.0%		29.0%		35.7%	11.1%	26.1%

The parentheses that follow a banner variable may contain, in addition to the low and high values, requests for means, medians, totals, percents. This is also where the declaration of a quantity variable, the request for no values and suggestions about the order of the columns are placed.

The chapter “SURVEY: More Statistics” describes the use of arithmetic operations on the banner variables. The ADD subcommand can be used to produce subtotals for the banner variables.

Figure 3.5 Means in the Banner on Another Variable

```
SURVEY Paceset,  LABELS 'Paceset.lab';
   BANNERS Own ( M )  Age ( M ),  STUB Income.groups,
   MEANS Income,   INCLUDE MISSING MEANS VALUES,  PLACES MEANS 0,
   NO MISSING,  MARGIN 12,  GAP 1;
```

		==== Respondent ==== ==== ownership ====			========== Age ==========			
	Total Sample	no	yes	Mean Income	Under 30	30 to 50	Over 50	Mean Income
Total Sample	80	11	69	30958	28	27	23	31107
	100.0%	100.0%	100.0%		100.0%	100.0%	100.0%	
Income of respondent								
Under $30,000	2	1	1	18400	-	2	-	18400
	2.5%	9.1%	1.4%			7.4%		
$30,000 to $60,000	50	9	41	30007	18	19	13	30007
	62.5%	81.8%	59.4%		64.3%	70.4%	56.5%	
Over $60,000	20	-	20	46976	10	3	6	46800
	25.0%		29.0%		35.7%	11.1%	26.1%	
Base	72	10	62	34398	28	24	19	34174
	90.0%	90.9%	89.9%		100.0%	88.9%	82.6%	
Mean Income	34398	26785	35626		37388	29311	35580	
S.D.	9334	5410	9277		9156	8038	8450	

3.5 Adding Means, Medians and Totals to the Columns

The mean, median, and total usually follow the columns for the banner variables. The single letter "M" is used to request means, "T" to request totals, “MED” for medians, and "B" to provide information on where they should be placed. "B" represents the banner points (columns) for a given variable. The “B” is used only to indicate placement. By default the individual banner points are always printed. “NO.V” or “NO.VALUES” is used when you do not wish to see the individual columns of frequencies. The syntax is:

```
Varname ( M )          means
Varname ( B M )        means -- the B is not necessary.
Varname ( M T )        means and totals
Varname ( M MED T )    means, medians and totals
Varname ( T M )        totals and means
```

```
Varname ( T M NO.V )    totals and means but no frequency columns
varname ( T M B )       totals and means precede the counts
varname ( 1 5 T B M )   ranges are supplied, totals precede
                        the counts and means follow them.
```

In Figure 3.4 the means for variable Own are placed on the right of the banner points (columns) for that variable. There are no means for variable Age. These requests must be made individually for each variable. The following while unlikely is legal:

```
BANNER Age ( M B T )  Income.Groups ( T M B )
```

Here variable Age has means to the left of the values for Age and totals to the right. Income.Groups also has both means and totals but they are both placed on the left of the values. Each banner variable may have a different configuration of columns. If NO.VALUES and B are both used, the B is ignored and the frequencies are not printed.

The means are usually based on the banner values unless the MEANS or MEDIAN subcommands provide alternate variables. When these alternate variables are given, they provide the data for all means (row or column) that are requested. Figure 3.5 shows the table that is produced when Income is the means variable and Income.groups defines the rows. The subcommand INCLUDE MISSING MEANS VALUES is appropriate because Income and Income.groups are different representations of the same data and have the same pattern of missing values.

The PLACES subcommand in Figure 3.5 causes the means to print without any fractional part. Means are usually printed to 2 decimal places. The various forms of the PLACES subcommand are discussed in the chapter "SURVEY: Controlling Layout and Appearance".

Figure 3.6 Means and Medians of the Banner Variable

```
SURVEY Paceset,   LABELS 'Paceset.lab';
  BAN Num.tv ( M MED ), STUB Sex;
```

```
                    ====== Number of television sets in your home: ======

              Total                                    5 or
             Sample      1      2      3      4       more     Mean    Median

Total            80      6     22     18     27        7     3.09      3.00
Sample        100.0% 100.0% 100.0% 100.0% 100.0%   100.0%

Sex

Male             39      4     18     11      4        2     2.54      2.00
                48.8%  66.7%  81.8%  61.1%  14.8%    28.6%

Female           40      2      4      7     22        5     3.60      4.00
                50.0%  33.3%  18.2%  38.9%  81.5%    71.4%

Missing           1      -      -      -      1        -
                 1.2%                         3.7%

Base             79      6     22     18     26        7     3.08      3.00
                98.8% 100.0% 100.0% 100.0%  96.3%   100.0%
Mean            1.51   1.33   1.18   1.39   1.85     1.71
S.D.            0.50   0.52   0.39   0.50   0.37     0.49
```

Figure 3.6 illustrates a mean and a median of the banner variable. Usually both the mean and the median are printed to 2 decimal places. They can be separately controlled.

```
PLACES MEDIAN 0,
```

causes the medians to print with zero decimal places. See the Chapter “SURVEY: The Statistics” for more information on the three ways to calculate the median and for information about producing medians on a variable other than the banner variables.

The labels for these new columns can be controlled by entries for SURVEY.LABELS in the labels file.

1. 102 replaces “Means”
2. 103 replaces “Good.N”
3. 104 replaces “Median”

3.6 Another Way to Display Percents

If you have few banner points but many stub values it is sometimes convenient to print the percents beside the counts rather than underneath. This is done by adding “PCT” to the parentheses that follow the banner variable. This has a second advantage because you can request percents for one variable and not for the next. However, the types of percent (row, column, total) that are printed will be the same for all variables that request them.

```
BANNER Age ( M PCT ), ROW COL PERCENTS,
```

Figure 3.7 Another Way to Display Percents

```
SURVEY Paceset,  LABELS 'Paceset.lab';
BAN SEX ( PCT ), STUB Age;
$
```

	Total Sample	Sex: Male Counts	Sex: Male Col %	Sex: Female Counts	Sex: Female Col %
Total Sample	80	39	100.0%	40	100.0%
Age					
Under 30	28	20	51.3%	8	20.0%
30 to 50	27	11	28.2%	16	40.0%
Over 50	23	7	17.9%	15	37.5%
Missing	2	1	2.6%	1	2.5%
Base	78	38	97.4%	39	97.5%
Mean	1.94	1.66		2.18	
S.D.	0.81	0.78		0.76	

Figure 3.7 contains the command, the subcommands, and the resulting display when PCT is used with a banner variable. The codes that can be used as values for SURVEY.LABELS to replace the labelling are:

1. 120 replaces “Counts”
2. 121 replaces “Row %”
3. 122 replaces “Col %”
4. 123 replaces “Tot %”

Because the sideways percent format is controlled by the banner variable it requires a special convention to use this format in the 1-way, stub only table. This convention can be used only if there is no variable in the P-STAT system file with the variable name “dummy”.

```
BAN DUMMY (PCT),
```

Figure 3.8 has the command and the printout for a 1-way table with the percents printed on the same line as the counts rather than beneath them. In a 1-way table the row percents are of no interest and the percent of the total n is the same as the column percents. Thus, only the column percents are of any interest. The other banner statistics such as the mean and median can be requested but will show only empty cells because the dummy banner variable has no values.

Figure 3.8 A Dummy Banner Variable

```
SURVEY Paceset,
    LABELS 'Paceset.lab';
    BAN DUMMY ( PCT ),
    TABLE.TITLE 'Column Percents with Different Bases',
    STUB Income.groups,
    LAYOUT TOTALS QUESTION BODY MISSING,
    MARGIN 20,
    COL PERCENTS BASED TOTAL.N, COL PERCENTS BASED GOOD.N;


    Column Percents with Different Bases

    Total Sample
                              80  100.0% 111.1%

    Income of respondent

    Under $30,000              2    2.5%   2.8%

    $30,000 to $60,000        50   62.5%  69.4%

    Over $60,000              20   25.0%  27.8%

    Missing                    8   10.0%  11.1%
```

3.7 Quantity Banner Variables

Figure 3.9 illustrates the subcommands and the printout when QUANTITY is used to describe a banner variable. When used by itself, the banner has but a single column containing the totals of all the values for the variable. It will often be used with M to get both totals and means. Regular banner variables can be freely mixed with variables described as quantity variables. A variable can be used as a quantity variable in one table and as a regular stub or banner variable (providing the range is appropriate) in another table.

When QUANTITY is used, M for means and PCT for percents may also be used. The only percent that is produced for such a variable is the column percent. In addition it is possible to cause the totals to be omitted and produce a table with just means or means and percents.

```
BAN Income ( QUAN, NO.VALUES )
```

When QUAN is used by itself

```
BAN Income ( QUAN )
```

the variable name or extended label is used to label the column. If it is used with M or PCT, the variable name or extended label encompasses all of the columns for that variable and each column is labelled with the appropriate column label. The value 105 when used with SURVEY.LABELS provides the text which replaces “Totals” for the quantity variable.

Figure 3.9 Quantity Banner variables

```
SURVEY Paceset,  LABELS 'Paceset.lab';
STUB Age, LAYOUT  LABELS TOTALS QUESTION BODY,
BAN Income (QUANTITY M ) Num.TV ( Q M );
$
```

					Number of television sets in your home:	
			===== Income =====			
	Total Sample		Totals	Mean	Totals	Mean
Total	80		2476660	34398.06	247	3.09
Sample	100.0%					
Age						
Under 30	28		1046870	37388.21	65	2.32
	35.0%					
30 to 50	27		703460	29310.83	87	3.22
	33.8%					
Over 50	23		676020	35580.00	88	3.83
	28.7%					

3.8 MULTIPLE RESPONSE BANNERS

There may be only one BANNER subcommand per tables group. If there is a single banner variable which is actually a multiple response variable, the BANNER subcommand is entered as MR.BANNER and is followed by the list of variables that comprise the MR.BANNER group.

```
MR.BANNER  Store.1 TO Store.4,
```

If you are using multiple response banners with multiple response stubs (MR.STUBS) read the section which follows this. It describes the PAIRED, UNPAIRED and CROSS.COMPARE subcommands which describe how the banners are related to the stubs.

Figure 3.10 shows the command and the table for a multiple response banner variable. The totals are the totals of the respondents in each column. The columns will usually total to more than the total sample size as each respondent may be represented several times.

The individual variables comprising a multiple response variable when that variable is to be used in the banner cannot be in a dummy variable format. They must be coded from 1 to n, where n is the total number of possible responses. The difference between these two methods of coding is described in detail in the previous chapter.

In file Paceset, the 4 variables store.1 to store.4 are in the 1/n format and each can have any of the 3 values, 1, 2, or 3. The 4 variables, Phone Lines Tablet and Ereader are carried in dummy variable form. If you wish to use these variables as a multiple response banner variable they can be recoded in the following way:

Figure 3.10 Multiple Response Banner Variables

```
SURVEY Paceset, LABELS 'Paceset.lab';
   MR.BAN  Store.1 to Store.4,
   STUB Income.groups,
   MEANS Income,  PLACES MEANS 0,
   NO MISSING,  NO ROW.TOTALS,
   CELL.WIDTH 9;
```

	=== Type of store item === === purchased in: ===		
	Discount	Depart ment	Web based source
Total Sample	34	50	46
	100.0%	100.0%	100.0%
Income of respondent			
Under $30,000	-	-	4
			8.7%
$30,000 to $60,000	22	26	31
	64.7%	52.0%	67.4%
Over $60,000	12	24	11
	35.3%	48.0%	23.9%
Base	34	50	46
	100.0%	100.0%	100.0%
Mean Income of respondent	35962	39640	33012
S.D.	7892	9582	9416

Figure 3.11 Recoding Dummy Variables to Use in a Multiple Response Banner

```
SURVEY Paceset
       [ DO #J #N USING phone to ereader;
           IF V(#J) = 1,   SET V(#J) = #N,
           FM.SET V(#J) = .M1. ;
         ENDDO ] ,
LABELS 'Paceset.lab'  'P2.lab';

CELL.WIDTH 10,  MARGIN 12,
MR.BANNER  Phone to Ereader,   STUB Age;
```

	========== Electronic Items ==========				
	Total Sample	Smart phone	Landline tele phones	Tablet computers	Ereaders
Total Sample	80	45	28	35	37
	100.0%	100.0%	100.0%	100.0%	100.0%
Age					
Under 30	28	19	10	15	18
	35.0%	42.2%	35.7%	42.9%	48.6%
30 to 50	27	13	13	8	12
	33.8%	28.9%	46.4%	22.9%	32.4%
Over 50	23	12	5	10	7
	28.7%	26.7%	17.9%	28.6%	18.9%
Missing	2	1	-	2	-
	2.5%	2.2%		5.7%	
Base	78	44	28	33	37
	97.5%	97.8%	100.0%	94.3%	100.0%
Mean	1.94	1.84	1.82	1.85	1.70
S.D.	0.81	0.83	0.72	0.87	0.78

```
SURVEY Paceset [DO #J #N USING phone to ereader;
                  IF V(#J) = 1,  SET V(#J) = #N,
                  FM.SET V(#J) =.M1. ;
                  ENDDO ],  LABELS 'Paceset.lab';
```

This bit of programming language takes each of the 4 variables in the list VCR to Ans.mach and recodes them so that a 1 on VCR remains a 1. A 1 on CD becomes a 2. A 1 on Port.phone becomes a 3. A 1 on Ans.mach becomes a 4, and all zeros are set to missing type 1. The scratch variable #J controls the loop referring to each of the 4 variables in turn. The second scratch variable #N, counts the loops. The values of the scratch variables as the loop progresses:

```
#N         #J                    V(#J)                          #N
1      position of phone       value of phone       -- if 1 -> 1
```

```
2      position of lines       value of lines      -- if 1 -> 2
3      position of tablet      value of tablet     -- if 1 -> 3
4      position of Ereader     value of Ereader    -- if 1 -> 4
```

When a variable is recoded from a dummy variable to the 1/n format, the labels must also be changed because dummy variables use the extended labels and 1/n coded variables use value labels. A new labels file, "P2.lab" contains appropriate labels for the variables we have just recoded.:

```
phone <<Electronic Items>>
(1) Smart phone
(2) Landline tele*phones
(3) Tablet computers
(4) Ereaders /
```

Figure 3.11 contains the complete command and the subcommands needed to use these 4 variables as a multiple response banner variable. Note the use of the second labels file to provide appropriate labels for the recoded variables.

Figure 3.12 Multiple Response Variable in a Nested Banner

```
CREATE MR.BAN.1  Store.1 TO Store.4,
BANNER MR.BAN.1 WITHIN Sex  Sex,   STUB Age;
```

Type of store item purchased in: / Sex

		Male			Female			Sex	
	Total Sample	Dis count	Depart ment	Web based source	Dis count	Depart ment	Web based source	Male	Female
Total	80	26	35	19	13	23	28	39	40
Sample	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Age									
Under 30	28	17	24	10	1	6	4	20	8
	35.0%	65.4%	68.6%	52.6%	7.7%	26.1%	14.3%	51.3%	20.0%
30 to 50	27	4	7	4	7	7	17	11	16
	33.8%	15.4%	20.0%	21.1%	53.8%	30.4%	60.7%	28.2%	40.0%
Over 50	23	5	4	3	5	9	7	7	15
	28.7%	19.2%	11.4%	15.8%	38.5%	39.1%	25.0%	17.9%	37.5%
Missing	2	-	-	2	-	1	-	1	1
	2.5%			10.5%		4.3%		2.6%	2.5%
Base	78	26	35	17	13	22	28	38	39
	97.5%	100.0%	100.0%	89.5%	100.0%	95.7%	100.0%	97.4%	97.5%
Mean	1.94	1.54	1.43	1.59	2.31	2.14	2.11	1.66	2.18
S.D.	0.81	0.81	0.70	0.80	0.63	0.83	0.63	0.78	0.76

The multiple response banner variables can be used with RANK, ORDER, FILL and SQUEEZE. However the MR.BANNER subcommand cannot be used with banner means, with other variables, or in a nested table. These problems are solved when a special variable is created from the individual multiple response variables in a way that can be used in a standard BANNER subcommand.

The CREATE subcommand creates 1 to 9 variables with the names MR.BAN.1 to MR.BAN.9. These variables can be referenced in a BANNER subcommand in the same way as any other variable, with one constraint: In a nested banner, the multiple response variable must be the inner level of the nest.

```
CREATE MR.BAN.1  Store.1 TO Store.4,
BANNER MR.BAN.1 WITHIN Sex ....
```

Figure 3.12 shows the subcommands and the table that is produced when a multiple response variable is nested in the banners of a table.

3.9 MULTIPLE RESPONSE: BANNERS AND STUBS

Multiple response variables in the banner and stub of a table may be paired to associate items such as a product with the store where it was purchased, a drug reaction with its measure of severity, and so on. Alternatively, the multiple response variables may be unpaired to show *all* possible pairings of responses — all pairings of purchases such as a skirt and blouse, a skirt and sweater and a skirt and pants, or all pairings of drug reactions such as fever and rash, fever and chills and fever and nausea. Additional corresponding variables such as costs of purchases or lengths of reactions may be counted instead of frequencies.

There are two variations on the unpaired situation. If the two sets of variables are not directly related, the totals are based on the number of respondents. These tables are referred to as “unpaired” multiple response tables. The subcommand that is used to indicate this is “UNPAIRED”.

When the two sets of variables are directly related and the total number of responses in both directions is of interest, the tables are referred to as “cross-comparison tables”. The CROSS.COMPARE subcommand is used in this situation.

If there is a multiple response stub variable and a multiple response banner and there are two or more banner variables, UNPAIRED is assumed unless PAIRED is specified. If there is only a single (multiple response) banner variable CROSS.COMPARE is assumed. This is done for consistency with earlier releases of the SURVEY command.

3.10 Paired Variables

The *paired* multiple response variables that are to be used in both the banner and stub of a table should correspond appropriately when they are paired. Figure 3.13 shows a paired multiple response table. The four variables VCR through Ans.Mach are electronic purchases; the four variables Store.1 through Store.4 are the stores in which the items were purchased. Store.1 is the store where the VCR was purchased, Store.2 is the store where the Compact Disc was purchased, and so on.

The subcommands that specify the variables and their pairing are:

```
CREATE MR.BAN.1   Store.1  TO  Store.4,
BANNER MR.BAN.1,
MR.STUB  phone TO ereader,  PAIRED,
```

If the multiple response variables are not contiguous in the input P-STAT system file, they may be named individually. The first of the multiple response stub variables is paired with the first of the multiple response banner variables, the second with the second, and so on. The subcommand PAIRED must be used. It applies only to the multiple response banner variable-multiple response stub variable pairs, and does not apply to the other single response variables cited in the banner subcommand. There may be as many as 5 such pairings in a single banner subcommand.

If yet another variable also corresponds appropriately to the paired variables, its values may be counted and displayed instead of the frequencies of each pairing. For example, if in addition to the electronic item and the store in which it was purchased, the price of the item was included in the data, price could be counted. Price would be stored in four variables, Price.1 through Price.4, with Price.1 being the price of the VCR purchased in Store.1, Price.2 the price of the Compact Disc purchased in Store.2, and so on. This subcommand would specify this pairing and counting:

```
CREATE MR.BAN.1   Store.1  TO  Store.4,
BANNER MR.BAN.1,
MR.STUB     Phone TO Ereaders,
COUNT       Price.1  TO  Price.4,
PAIRED,
```

Figure 3.13 Paired Multiple Response Banner and Stub Variables

```
SURVEY Paceset,  LABELS 'Paceset.lab';
  CREATE MR.BAN.1 Store.1 to Store.4, GAP 3,
  BANNER  MR.BAN.1  Sex,
  MR.STUB VCR TO Ans.MACH,
  PAIRED;
```

		== Type of store == item purchased in			== Sex:: == Gender_of_ respondent	
	Total Sample	Discount	Department	Web based source	Male	Female
Totals	80	39	58	48	39	40
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Which of the following do you own?						
Smart phone as primary telephone	45	11	20	14	27	18
	56.3%	28.2%	34.5%	29.2%	69.2%	45.0%
One or more landline based telephones	28	8	8	12	12	16
	35.0%	20.5%	13.8%	25.0%	30.8%	40.0%
One or more tablet computers	35	8	16	11	21	14
	43.8%	20.5%	27.6%	22.9%	53.8%	35.0%
One or more Ereaders	37	12	14	11	20	16
	46.3%	30.8%	24.1%	22.9%	51.3%	40.0%
Missing	11	-	-	-	5	6
	13.8%				12.8%	15.0%
Responses	145	39	58	48	80	64

The COUNT subcommand should follow the MR.STUB subcommand in the same way that COUNT follows STUB in a single response table.

When PAIRED is used, the row totals are calculated using the counts (respondents) because that permits paired tables and unpaired tables to share the same row totals. It is possible to have the row totals based on responses only if there is a single multiple response banner variable and the subcommand

```
ROW.TOTALS BASED RESPONSES
```

is added to the command. This is illustrated in Figure 3.14.

Figure 3.14 PAIRED: Row Totals Based on Responses

```
SURVEY Paceset,  LABELS 'Paceset.lab';
  CREATE MR.BAN.1 Store.1 to Store.4, GAP 3,
  BANNER  MR.BAN.1,
  MR.STUB Phone TO Ereader,
  PAIRED, ROW.TOTALS BASED RESPONSES;
```

		== Type of store == item purchased == in		
	Total Sample	Dis count	Depart ment	Web based source
Totals	320	39	58	48
	100.0%	100.0%	100.0%	100.0%
Which of the following do you own?				
Smart phone as primary telephone	45	11	20	14
	14.1%	28.2%	34.5%	29.2%
One or more landline based telephones	28	8	8	12
	8.8%	20.5%	13.8%	25.0%
One or more tablet computers	35	8	16	11
	10.9%	20.5%	27.6%	22.9%
One or more Ereaders	37	12	14	11
	11.6%	30.8%	24.1%	22.9%
Responses	145	39	58	48

3.11 Unpaired Multiple Response Banner and Stub Tables

Figure 3.15 illustrates the type of situation that calls for an unpaired multiple response table. In an expansion of the Paceset survey, the respondents were asked about their first and second preferences in type of music. The music questions are not related to the ownership questions in the way that the store questions were related. They are "UNPAIRED". The row and column totals are the number of respondents. The percents are based on the total counts unless the PERCENTS subcommand is used to change the base.

An UNPAIRED multiple response banner produces a table that is exactly the same as one that is composed of several single value banner variables. The one difference is in the labelling.

```
MR.STUB VCR TO Ans.Mach,
BANNER Classical Country Rock SEX,
```

If the group of question is in the form: “What are your two preferred type of music?” the multiple response banner format must be used. If the group of questions is in the form: “Do you like classical music?”, “Do your like country music?”, etc. you may construct the table as single variables or as a multiple response variable.

Figure 3.15 Unpaired Multiple Response Tables.

```
SURVEY Paceset, LABELS 'Paceplus.lab';
   GAP 2, MARGIN 12, CELL.WIDTH 8,
   CREATE MR.BAN.1 Music.first Music.second,
   BANNER MR.BAN.1 Sex,
   MR.STUB VCR TO Ans.Mach,  UNPAIRED;
$
```

		==== What types of ====			===== Sex =====	
		==== music do you ====				
		==== enjoy ====				
	Total Sample	Classical	Country	Rock	Male	Female
Total Sample	80	13	50	36	39	40
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Which of the following do you own?						
Video Cassette Recorder	45	9	27	21	27	18
	56.2%	69.2%	54.0%	58.3%	69.2%	45.0%
Compact Disc Player	28	4	18	8	12	16
	35.0%	30.8%	36.0%	22.2%	30.8%	40.0%
Portable Telephone	35	9	21	14	21	14
	43.8%	69.2%	42.0%	38.9%	53.8%	35.0%
Answering Machine	37	9	26	15	20	16
	46.2%	69.2%	52.0%	41.7%	51.3%	40.0%
Missing	11	-	7	7	5	6
	13.8%		14.0%	19.4%	12.8%	15.0%
Base	69	13	43	29	34	34
	86.2%	100.0%	86.0%	80.6%	87.2%	85.0%
Responses	145	31	92	58	80	64

If you have coded the replies so that the first variable in the list is set to either "1" or missing, and the second variable is set to either "2" or missing, and so forth, then you do not need to make any coding changes to use either banner format. If you wish to have the variables grouped together with a common heading, use the multiple response format and provide labels and extended variable labels for the first variable in the list.

```
labels file:    Classical <<What types of music do you enjoy>>
                 (1) Classical (2) Country (3) Rock /

subcommand:     CREATE MR.BAN.1  Classical Country Rock,
                MR.BANNER  MR.BAN.1,
```

If you do not wish to have an extended label across the group of variables, use the standard banner format. Each column will be labelled with either its variable name or its extended label.

Figure 3.16 Cross Comparison Table

```
SURVEY Paceset  [ KEEP Phone TO Ereader;
    DO #j #n USING Phone to Ereader;
    IF V(#j) = 1,   SET V(#j) = #n,
    FM.SET V(#j) = .M1. ;
    ENDDO ] ,  LABELS 'P2.lab';

 CREATE MR.BAN.1   VCR TO ANS.MACH,
 BANNER MR.BAN.1,  MR.STUB VCR TO Ans.mach,
 CROSS.COMPARE,
 NO QUESTION,   NO MISSING,   NO SUMMARY;
```

		========= Electronic Items ========			
	Total Sample	Smart phone	Landline tele phones	Tablet comp uters	Ereaders
Total Sample	375	111	75	93	96
	100.0%	100.0%	100.0%	100.0%	100.0%
Smart phone	111	45	17	25	24
	29.6%	40.5%	22.7%	26.9%	25.0%
Landline telephones	75	17	28	14	16
	20.0%	15.3%	37.3%	15.1%	16.7%
Tablet computers	93	25	14	35	19
	24.8%	22.5%	18.7%	37.6%	19.8%
Ereaders	96	24	16	19	37
	25.6%	21.6%	21.3%	20.4%	38.5%

3.12 Cross Comparison Tables

When CROSS.COMPARE is used with a multiple response banner-multiple response stub table, the corresponding multiple response banner and stub variables are not paired. Instead, *all* multiple response banner variables are paired with *all* multiple response stub variables. This typically makes sense when the same set of multiple response variables are used in both the banner and the stub. Figure 3.16 illustrates this type of table.

The multiple response variables VCR through Ans.Mach are used in both the banner and the stub of the table. Because multiple response variables that are used in a banner must have 1 through n coding and these have 0/1 coding, they are recoded as the file is input to the SURVEY command. An appropriate label file that reflects this new coding is written beforehand. Because there is only a single banner variable, the CROSS.COMPARE need not be specified.

In a cross comparison table, *all* possible pairings of the variables are included in the table — VCR is paired with VCR, then with Compact Disc, then with Portable Telephone and then with Answering Machine. Compact Disc is then paired with each of the other variables. Similarly, Portable Telephone and Answering Machine are paired with each of the other variables.

In Figure 3.16, the Total Sample of 375 is the total number of responses that were made. The diagonals of 45, 28, 35 and 37 are the total counts of each type of electronic purchase (compare with Figure 3.13). The off-diagonals are the counts of the various pairings — 17 is the number of respondents who purchased both a VCR and a Compact Disc; 25 is the number who purchased both a VCR and a Portable Telephone, and so on.

The Row Totals at the left of the table are all cross pairings of purchases in that row— for example, 111 is the number of pairings of VCR with itself and with the other purchases. The statistics in the summary section, the base and responses that are usually printed in a multiple response table are uninteresting as they are also the totals of the responses. There is no such thing as missing in this type of table.

Because the unpaired use of multiple response in both columns and rows is based entirely on the number of responses, while a single response variable with a multiple response stub has totals based on the number of respondents, the banner in the unpaired situation is limited to a single multiple response variable.

3.13 TRANSPOSED TABLE FORMAT

Surveys may be *transposed* — flipped or rotated 90 degrees — to change their appearance and to produce attractive tables of ratings, marginals and summary statistics. The BANNER variables define the rows and the STUB variables define the columns in a transposed table. After the tables are transposed, portions of the layout may be rearranged relative to each other in both their horizontal and vertical aspects.

GROUP.STUBS is used together with TRANSPOSE to produce variable-by-values tables without recoding or creating indices. This format is appropriate for ratings, marginals and summary statistics of related variables. NOTE: many of the features of the regular tables are not available in the transposed format.

The TRANSPOSE subcommand rotates a table. TRANSPOSE may be used as part of a SURVEY subcommand to rotate the table currently being defined, or after AGAIN to rotate the previous table. Figure 3.17 shows the default table layout in both the regular form and the transposed form. AGAIN and TRANSPOSE are used to rotate and print the previous table.

When the table is transposed, the stub variable becomes the column variable and the banner variable becomes the row variable. The column totals become the row totals and vice versa. The summary section of the table is no longer underneath the body of the table, but positioned to the right of it. The layout of the transposed table may be changed to further affect the table appearance.

3.14 LAYOUT of a Transposed Table

The typical appearance of a table is changed when TRANSPOSE and LAYOUT are used. The table appearance just may be more pleasing this way, or the table in its new orientation (often horizontal instead of vertical) may fit better on the page or in a report.

Figure 3.17 Transposing a Survey: Default Layout and Format

```
SURVEY  PaceSet,  LABELS  'PaceSet.Lab' ;
  BANNER  Sex,
  STUB  Age ;
```

	Total Sample	==== Sex ==== Male	Female
Total	80	39	40
Sample	100.0%	100.0%	100.0%
Age			
Under 30	28	20	8
	35.0%	51.3%	20.0%
30 to 50	27	11	16
	33.8%	28.2%	40.0%
Over 50	23	7	15
	28.7%	17.9%	37.5%
Missing	2	1	1
	2.5%	2.6%	2.5%
Base	78	38	39
	97.5%	97.4%	97.5%
Mean	1.94	1.66	2.18
S.D.	0.81	0.78	0.76

AGAIN, TRANSPOSE ;

		Age					
	Total Sample	Under 30	30 to 50	Over 50	Miss ing	Base	Mean	S.D.
Total	80	28	27	23	2	78	1.94	0.81
Sample	100.0%	35.0%	33.8%	28.7%	2.5%	97.5%		
Male	39	20	11	7	1	38	1.66	0.78
	100.0%	51.3%	28.2%	17.9%	2.6%	97.4%		
Female	40	8	16	15	1	39	2.18	0.76
	100.0%	20.0%	40.0%	37.5%	2.5%	97.5%		

Figure 3.18 Parts of the LAYOUT: Regular and Transformed Survey

```
LAYOUT: USUAL ARRANGEMENT

SURVEY  PaceSet,  LABELS  'PaceSet.Lab' ;
BANNER  Sex,  STUB  Age ;
```

	==== Sex ====		
	Total Sample	Male	Female
Total Sample	80 100.0%	39 100.0%	40 100.0%
Age			
Under 30	28 35.0%	20 51.3%	8 20.0%
30 to 50	27 33.8%	11 28.2%	16 40.0%
Over 50	23 28.7%	7 17.9%	15 37.5%
Missing	2 2.5%	1 2.6%	1 2.5%
Base	78 97.5%	38 97.4%	39 97.5%
Mean	1.94	1.66	2.18
S.D.	0.81	0.78	0.76

← **1. LABELS**
← **2. TOTALS**
← **3. QUESTION**
← **4. BODY**
← **5. MISSING**
← **6. SUMMARY**

```
AGAIN,  TRANSPOSE ;
```

	Age							
	Total Sample	Under 30	30 to 50	Over 50	Missing	Base	Mean	S.D.
Total Sample	80 100.0%	28 35.0%	27 33.8%	23 28.7%	2 2.5%	78 97.5%	1.94	0.81
Sex								
Male	39 100.0%	20 51.3%	11 28.2%	7 17.9%	1 2.6%	38 97.4%	1.66	0.78
Female	40 100.0%	8 20.0%	16 40.0%	15 37.5%	1 2.5%	39 97.5%	2.18	0.76

1. ←
2. ←
3. ←
4. ←

4. ↑ **5.** ↑ **6.** ↑

Figure 3.18 shows the tables from Figure 3.17 with the parts of their layouts clearly labeled. The LAYOUT subcommand is used to reorder these parts. When LAYOUT is used in a table definition that also includes TRANSPOSE, the LAYOUT arguments refer to the parts in the *transposed* table. The order of the subcommands in the table definition does not matter — TRANSPOSE may precede or follow LAYOUT (or any of the other subcommands).

The relationships between the parts of the layout in a transposed table are somewhat different from those in a regular table. The LABELS, TOTALS, QUESTION and BODY may be ordered *vertically* with respect to each other — that is, the *rows* comprising these parts may be reordered. The BODY, MISSING and SUMMARY may be ordered *horizontally* with respect to each other — the *columns* comprising these parts may be reordered. Notice that the BODY of the table is positioned relative to the other parts in both dimensions.

3.15 Summary Statistics

TRANSPOSE and GROUP.STUBS may also be used to produce tables of summary statistics. Because the various statistics define the columns of the tables, many variables may be included in a concise table. As is the case with tables of ratings and marginals, a banner variable should *not* be defined.

Figure 3.19 shows the transposed table from the prior two figures with a rearranged layout. The QUESTION is first (it is simply the variable name in this table, but it could be an extended label). The LABELS and TOTALS follow. Thus, these three parts come before (above) the body of the table. Next, the SUMMARY is positioned before (left of) the body and MISSING is positioned after (right of) the body. SUBTOTALS or NETS are not included currently in transposed tables, and thus they may not be part of the LAYOUT subcommand.

Also, TRANSPOSE may *not* be used with nesting (WITHIN), subtotals (DEFINE) or nets (DEFINE.NET). When there is more than one banner variable, TRANSPOSE produces a separate table for each of the banner variables. ROW.TOTALS may be used to position that column on the right or left of the table body. NO TRANSPOSE, TRANSPOSE OFF or RESET restores the default table orientation.

If ORDER or RANK are used, they apply to the stub variables only. Thus the rank or order changes are made to the columns of the table.

Figure 3.19 The Layout of a Transposed Table May Be Changed

```
AGAIN, LAYOUT  QUESTION LABELS TOTALS SUMMARY BODY MISSING ;
```

Sex

				Age				
	Base	Mean	S.D.	Total Sample	Under 30	30 to 50	Over 50	Missing
Total	78	1.94	0.81	80	28	27	23	2
Sample	97.5%			100.0%	35.0%	33.8%	28.7%	2.5%
Male	38	1.66	0.78	39	20	11	7	1
	97.4%			100.0%	51.3%	28.2%	17.9%	2.6%
Female	39	2.18	0.76	40	8	16	15	1
	97.5%			100.0%	20.0%	40.0%	37.5%	2.5%

Figure 3.20 TRANSPOSE and GROUP.STUBS Produce a Ratings Table

```
mFile Ratings:

   Q1    Q2    Q3    Q4    Q5

    3     4     3     -     5
    1     2     2     4     2
    3     5     3     3     4
    4     1     5     5     3
    1     2     -     -     -
    5     5     3     5     4
    4     5     5     4     4
    3     5     5     5     4
    4     4     5     4     5
    5     2     3     2     1
   Label File Ratings.Lab:
Q1 TO Q5
  (1) Poor (2) Mediocre (3) Okay (4) Good (5) Excel*lent /
Q1 <<How would you rate the flavor?>>  /
Q2 <<                   the taste?>>   /
Q3 <<                   the texture?>> /
Q4 <<                   the color?>>   /
Q5 <<                   the aroma?>>   /  $


    SURVEY  Ratings,  LABELS  'Ratings.Lab' ;

      TRANSPOSE,  GROUP.STUBS  Q1  TO  Q5,     CELL.WIDTH 6,
      LAYOUT   LABELS  TOTALS  BODY  SUMMARY,  MARGIN 20,
      SUMMARY  BASE  MEAN,     TITLE  'Soft Dessert Ratings' ;
```

Soft Dessert Ratings

	Total Sample	Poor	Medi ocre	Okay	Good	Excel lent	Base	Mean
How would you rate the flavor?	10 100.0%	2 20.0%	-	3 30.0%	3 30.0%	2 20.0%	10 100.0%	3.30
the taste?	10 100.0%	1 10.0%	3 30.0%	-	2 20.0%	4 40.0%	10 100.0%	3.50
the texture?	10 100.0%	-	1 10.0%	4 40.0%	-	4 40.0%	9 90.0%	3.78
the color?	10 100.0%	-	1 10.0%	1 10.0%	3 30.0%	3 30.0%	8 80.0%	4.00
the aroma?	10 100.0%	1 10.0%	1 10.0%	1 10.0%	4 40.0%	2 20.0%	9 90.0%	3.56

Figure 3.21 TRANSPOSE and GROUP.STUBS Produce a Table of Marginals

```
SURVEY  PaceSet ;

TRANSPOSE,  GROUP.STUBS  Age  Sex  Num.TV  Phone TO Income.Groups,
   LAYOUT   LABELS  TOTALS  BODY  ;
```

	Total Sample	0	1	2	3	4	5	6
Age	80	-	28	27	23	-	-	-
	100.0%		35.0%	33.8%	28.7%			
Sex	80	-	39	40	-	-	-	-
	100.0%		48.8%	50.0%				
Num.TV	80	-	6	22	18	27	6	1
	100.0%		7.5%	27.5%	22.5%	33.8%	7.5%	1.2%
Phone	80	35	45	-	-	-	-	-
	100.0%	43.8%	56.2%					
Lines	80	52	28	-	-	-	-	-
	100.0%	65.0%	35.0%					
Tablet	80	45	35	-	-	-	-	-
	100.0%	56.2%	43.8%					
Ereader	80	43	37	-	-	-	-	-
	100.0%	53.8%	46.2%					
Store.1	80	-	11	20	14	-	-	-
	100.0%		13.8%	25.0%	17.5%			
Store.2	80	-	8	8	12	-	-	-
	100.0%		10.0%	10.0%	15.0%			
Store.3	80	-	8	16	11	-	-	-
	100.0%		10.0%	20.0%	13.8%			
Store.4	80	-	12	14	11	-	-	-
	100.0%		15.0%	17.5%	13.8%			
Income.	80	-	2	50	20	-	-	-
Groups	100.0%		2.5%	62.5%	25.0%			

3.16 Tables of Ratings and Marginals

Tables of ratings and marginals — sometimes called *variables-by-values* tables — are easily produced using TRANSPOSE with GROUP.STUBS. Appropriate LAYOUT options select and position the parts of the table.

The variables should be either a multiple response grouping or other variables related only by a common range of values.

Figure 3.20 shows a data file of ratings of various aspects of soft desserts. The ratings go from 1 (poor) through 5 (excellent). The label file is also shown in Figure 3.20. The value labels, which are the same for all 5 variables, are provided first. Extended variable labels are then provided individually for each variable.

The essential parts of the SURVEY subcommand that define the ratings table are:

```
TRANSPOSE,  GROUP.STUBS  Q1  TO  Q5,
LAYOUT   LABELS  TOTALS  BODY  SUMMARY,
```

The additional options in the subcommand merely enhance the table appearance. Notice that BANNER is not included in the table definition.

When TRANSPOSE and GROUP.STUBS are used to request a ratings table, a banner variable should *not* be specified. (The combination of TRANSPOSE, GROUP.STUBS and no banner variable results in a ratings table, where the extended variable labels are used instead of "Total Sample" to label what are in effect rotated row totals.)

Figure 3.21 shows counts of each value for a group of variables with a common or overlapping range of values. This type of display is often called a table of *marginals*. It provides an overall look at the frequencies of responses to a group of questions (variables). The SUMMARY is left out of the LAYOUT subcommand in this table definition.

3.17 Summary Statistics

TRANSPOSE and GROUP.STUBS may also be used to produce tables of summary statistics. Because the various statistics define the columns of the tables, many variables may be included in a concise table. As is the case with tables of ratings and marginals, a banner variable should *not* be defined.

Figure 3.22 TRANSPOSE and GROUP.STUBS Make a Table of Summary Statistics

```
SURVEY  PaceSet,  LABELS  'PaceSet.Lab' ;

TRANSPOSE,  GROUP.STUBS  Age  Sex  Income.Groups,
SUMMARY  CONTAINS  BASE  MEAN  MEDIAN  MODE  S.D  S.E  LOW  HIGH,

LAYOUT   LABELS    TOTALS   SUMMARY,  NO PERCENTS,
CHAR FOR UNDERLINE  '=',    MARGIN 8 ;
```

	Total Sample	Base	Mean	Median	Mode	S.D.	Standar d error	Min imum	Max imum
Age	80	78	1.94	2	1	0.81	0.09	1	3
Sex	80	79	1.51	2	2	0.50	0.06	1	2
Income	80	72	2.25	2	2	0.50	0.06	1	3

Figure 3.22 shows a table of summary statistics. The SUMMARY subcommand specifies the statistics to be included in the table. Currently, only categorical variables or continuous variables with limited ranges should be

used in this type of table, unless the statistics in the summary are restricted to the mean, standard deviation, standard error, base and/or totals. The statistics that are available in the summary section are discussed in more detail in later chapters.

SUMMARY

SURVEY

```
SURVEY Paceset,  LABELS 'Paceset.lab';
  BANNERS Age ( T M B )   Q33 ( 1 3 )  Income ( 1 2 M )
  STUB Num.TV;
```

```
SURVEY Paceset,  LABELS 'Paceset.lab';
  CREATE MR.BAN.1 Store.1 to Store.4,
  BANNERS MR.BAN.1 Sex,
  STUB Income.group,  MEANS Income;
```

The SURVEY command has a number of options for controlling the content and the arrangement of the columns of the table. These include supplying ranges, requesting means or totals of the variables. Banners may be multiple response variables and multiple response variables may be mixed with single response variables when the CREATE subcommand is used.

Subcommands: Column definitions

BANNERS vn vn

Each of the banner variables may be followed by parentheses containing the low and high values to be used for that variable. When ranges are not supplied, the lowest and highest observed values for the variables are used.

In addition to the low and high, the parentheses may contain any of the following:

```
"M"              a request for means
"T"              a request for totals
"B"              a positioning indicator
"MED".           a request for medians
"Q" or "QUAN"    specifies a quantity variable
"PCT"            a request for percents
```

```
BANNER Age ( T B M )
```

This is a request for totals on age placed before the values and means on age placed after the values. Each variable in the list of banner variables may have its own individual requests.

When BANNER is used, the variables may include any of the 9 special variables MR.BAN.1 to MR.BAN.9 which are defined using the CREATE subcommand.

CONTROL COLUMN ORDER USING LABELS

The columns of each of the banner variables in the table are to appear in the same order as the corresponding labels in the labels file. CONTROL COLUMN ORDER OFF can be sued to turn this off.

CONTROL COLUMN RANK AFTER nn

The number "nn" is one which is outside the range of the variable. Any column with labels found before this value in the labels file is to be ranked. Other columns are to be left in natural order.

CREATE MR.BAN.nn vn vn

CREATE is used to create one of the 9 special variables MR.BAN.1 to MR.BAN.9. One of these 9 keywords is followed by the list of variables which together are to be considered a single variable. The 9 special variables can be used with other variables in a BANNER subcommand.

```
CREATE MR.BAN.1 Q10A TO Q10M,
BANNER MR.BAN.1 Q33,
```

MR.BANNERS vn vn

provides a list of variables which comprise a single multiple response variable. MR.BANNER cannot be used in conjunction with other multiple response variables or with single response variables. If you need this capability use CREATE to define the special variables MR.BAN.1 through MR.BAN.5 which can be included in the standard list of BANNER variables.

Options subcommands;

BVAR.PER.PAGE nn nn ...

suggests the desired number of banner variables to print on a page. If BVAR.PER.PAGE is set to 1, each new banner variable will begin on a new page. If BVAR.PER.PAGE is set to 0, columns are placed one after another as many as can fit on a page without considering “widows”. If several numbers are used, each one controls the spacing for a given page. This is honored only if the number of columns fits in the specified output width. If there are extra banner variables they are placed using the default spacing rules.

CROSS.COMPARE

specifies that the values of the multiple response stub variables and the multiple response banner variables are to be displayed in cross-comparison tables. Regular single response variables cannot be used in the banner when CROSS.COMPARE is specified. See PAIRED and UNPAIRED for alternate ways of handling a table when both the stubs and banners have multiple response variables.

FILL COLUMNS

Fill is the assumed setting for columns and squeeze is the assumed setting for rows. When FILL (ROWS OR COLUMNS) is used, all rows or columns between the low and high values are printed even if they have no data. FILL or FILL ROWS turns on row fill. FILL COLUMNS turns on column fill.

GROUP.STUBS vn vn

can be used with the TRANSPOSED format to create tables of summary statistics or of marginals. GROUP.STUBS are described in detail in “SURVEY: All About the Rows”.

ORDER COLUMNS UP / DOWN

The columns of the tables usually print in order from the lowest value through the highest value. This is ORDER COLUMNS UP. Use of ORDER COLUMNS DOWN reverses this presentation. ORDER without COLUMNS is equivalent to ORDER ROWS.

PAIRED

specifies that the values of the defined MR.STUB and MR.BANNER variables are paired together. There should be the same number of multiple response stub and banner variables:

```
MR.STUB  Car.1     TO  Car.4,
MR.BAN   Dealer.1  TO  Dealer.4,  PAIRED ;
```

In this example, the stub variables are models of cars and the banner variables are dealers where the cars were purchased. The value of Car.1 is paired with the value of Dealer.1, and an entry is put in the appropriate cell in the table. Car.2 is paired with Dealer.2, and so on.

See CROSS.COMPARE and UNPAIRED for alternate ways of handling a table when both the stubs and banners have multiple response variables.

RANK COLUMNS UP / DOWN

requests that within each banner variable, the columns be ranked either UP or DOWN by the frequencies of the column totals rather than ordered by the values of the columns. RANK COLUMNS can be used with a number to indicate how many columns are to be included in the ranking.

```
RANK COLUMNS -1 DOWN          Rank all but the last value
RANK COLUMNS  3 UP            Rank only the first 3 values
```

ROW.TOTALS BASED RESPONSES

used only with a single paired multiple response banner and multiple response setup to specify that the base for the row totals should be responses rather than respondents.

SQUEEZE COLUMNS

SQUEEZE is the opposite of FILL. SQUEEZE columns requests that any empty column be omitted from the printout.

UNPAIRED

specifies the that the multiple response banner and multiple response stubs are not to be paired but tabulated separately. This differs from CROSS.COMPARE because the totals are based on the number of respondents and not the number of responses. When UNPAIRED is used, multiple response banner variables and single response banner variables can be used in the same table.

TRANSPOSE

requests that the defined table be rotated 90 degrees. Tables are transposed primarily for two reasons: 1) to change their appearance, and 2) to produce tables of ratings, marginals and summary statistics.

The appearance of transposed tables tends to be wider and shorter than their untransposed versions because the mean and the summary statistics (the base, mean and standard deviation) are on the right of the table, instead of at the bottom. The other SURVEY subcommands, such as LAYOUT, CELL.WIDTH and CHAR, etc., may be used with TRANSPOSE to further alter the appearance of the table.

When TRANSPOSE is used with GROUP.STUBS and a grouping of stub variables, tables of ratings, marginals or summary statistics may be produced. The variable names or extended labels define the rows of the tables and the values of the variables define the columns. LAYOUT may be used to delete or order different portions of the tables. This subcommand produces a table of marginals:

```
TRANSPOSE,  GROUP.STUBS  Q1 TO Q22,  LAYOUT  LABELS  TOTALS  BODY ;
```

The variables in the grouping should have common values because the columns will go from the lowest value through the highest.

Many of the options such as FILL, SQUEEZE and nested tables that are available in regular tables are not available in the transposed format.

4
SURVEY: Subtotals and Nets

This chapter covers five topics:

1. subtotals which are associated with the DEFINE subcommand
2. nets which are associated with the DEFINE.NET subcommand
3. interactions with RANK
4. dynamic recodes of stub values using the COMBINE subcommand
5. exception to the recodes using the EXCEPT subcommand.

4.1 SUBTOTALS AND NETS

The difference between a subtotal and a net is best illustrated with an example. A survey of breakfast cereals includes the following question: “Which of the following describe your feelings about Cheezy Flakes? Check all that apply.” A subtotal provides the total of the positive responses. A net counts the number of people who checked at least one of the positive responses. A respondent who likes Cheezy Flakes because it is sweet and because it is crunchy is counted only once in a net but counted twice in a subtotal. For this particular type of question, a net is usually more appropriate than a subtotal.

The question “Which of the following sports have you participated in during the past week?” lends itself to either a subtotal or a net. A possible net would be the number of people who had exercised during the week. A possible subtotal would be the total golf and tennis games played during the week. In the subtotal a person who had played both golf and tennis would be represented twice. In the net, that same person would be represented only once.

Subtotals can be computed for any type of stub variable, STUB, GROUP.STUB or MR.STUB. Nets apply only to MR.STUB, multiple response stub groups. When you are processing a multiple response table, the choice depends on what you want to know about the multiple response item. Subtotals, which are nothing but the sum of 1 or more rows of the table, are computed after the body of the table has been tabulated. Because it is not possible to tell from looking at the table whether a respondent is represented in more than one of its rows, nets are computed as the data are being read. Thus nets cannot be used when you are ranking the rows of a table to determine the members of the highest ranking group. The highest ranking group is not known until after the nets have been processed.

A table (where table is defined by a single stub variable or a single MR.STUB or GR.STUB group) can have any number of subtotals or nets. A table may have nothing but subtotals or nets. However, any single table cannot have both subtotals and nets.

The location of subtotals and nets in the table depends on the LAYOUT and on the SUBTOTALS subcommands. All of the subtotals or nets can be positioned together as a distinct part of the table or they can be interleaved in the body of the table. If RANK is requested, the ranking can be done so that the subtotal with the largest frequency (and all of the rows which comprise that subtotal) are first in the table. Alternatively, ranking can be done so that rows are ranked within the subtotals but the position of the subtotal groups depends on the order in which the subtotals or nets are defined.

4.2 Defining Subtotals or Nets

Subtotals are requested by using the DEFINE subcommand. Nets are requested by using the DEFINE.NET subcommand. The format for the two subcommands is identical. The difference is in the keyword and in the way that the groupings are computed.

```
DEFINE     'Label for the subtotal row'  Var List  Values,
DEFINE.NET 'Label for the net'           Var List  Values,
```

To create subtotals for the variable Num.TV in file Paceset we use:

```
DEFINE '1 or 2 TV sets'      Num.TV   1 2,
DEFINE '3 or more TV sets'   Num.TV   3 to 5,
```

Figure 4.1 Defining Subtotals

```
SURVEY Paceset,  LABELS 'Paceset.lab';
BANNER Age Sex,  STUB Num.TV,   NO MISSING,   NO SKIP,
DEFINE  '1 or 2 TV sets'      Num.TV   1 2,
DEFINE  '3 or more TV sets'   Num.TV   3 TO 5 ;
$
```

```
                    ======== Age ======= ==== Sex ====

               Total  Under  30 to   Over
              Sample     30     50     50   Male Female

Total             80     28     27     23     39     40
Sample         100.0% 100.0% 100.0% 100.0% 100.0% 100.0%

Number of television sets in your home:

       1           6      4      1      1      4      2
                 7.5%  14.3%   3.7%   4.3%  10.3%   5.0%
       2          22     14      7      1     18      4
                27.5%  50.0%  25.9%   4.3%  46.2%  10.0%
       3          18      8      5      4     11      7
                22.5%  28.6%  18.5%  17.4%  28.2%  17.5%
       4          27      1     13     12      4     22
                33.8%   3.6%  48.1%  52.2%  10.3%  55.0%
5 or more           7      1      1      5      2      5
                 8.8%   3.6%   3.7%  21.7%   5.1%  12.5%

1 or 2 TV          28     18      8      2     22      6
sets            35.0%  64.3%  29.6%   8.7%  56.4%  15.0%
---------
3 or more          52     10     19     21     17     34
TV sets         65.0%  35.7%  70.4%  91.3%  43.6%  85.0%
---------

Base               80     28     27     23     39     40
                100.0% 100.0% 100.0% 100.0% 100.0% 100.0%
Mean             3.09   2.32   3.22   3.83   2.54   3.60
S.D.             1.13   0.90   1.01   0.98   1.00   1.01
```

The label, which follows DEFINE or DEFINE.NET, is required. It must be enclosed in quotes (single or double) or angle brackets. There must also be at least 1 variable name and at least 1 numeric argument.

Figure 4.1 illustrates the subcommands and output when DEFINE is used to create two subtotals. Figure 4.2 illustrates DEFINE with multiple response stubs. Figure 4.3 shows the same table with nets instead of subtotals. Notice that the values in the defines are overlapping and that it is not necessary to specify in the DEFINE all the variables in the multiple response stub group. It is the first variable in the group that is important. For example:

```
MR.STUB VCR to Ans.Mach,
DEFINE 'VCR or CD'   VCR   1 2,
```

Figure 4.2 Subtotals with Multiple Response Variables

```
SURVEY Paceset,  LABELS 'Paceset.lab';

    DEFINE <<Phone plus Lines>>                          Phone  1 2,
    DEFINE <<Tablet plus Ereader>>                       Phone  3 4,
    DEFINE <<Total Electronics>>                         Phone 1 TO 4,
    BAN Age Sex, MR.STUB Phone to Ereader, NO MISSING, NO SUMMARY;
    $
```

		======== Age =======			==== Sex ====	
	Total Sample	Under 30	30 to 50	Over 50	Male	Female
Total Sample	80	28	27	23	39	40
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Which of the following do you own?						
Smart phone as primary telephone	45	19	13	12	27	18
	56.2%	67.9%	48.1%	52.2%	69.2%	45.0%
One or more landline based telephones	28	10	13	5	12	16
	35.0%	35.7%	48.1%	21.7%	30.8%	40.0%
One or more tablet computers	35	15	8	10	21	14
	43.8%	53.6%	29.6%	43.5%	53.8%	35.0%
One or more Ereaders	37	18	12	7	20	16
	46.2%	64.3%	44.4%	30.4%	51.3%	40.0%
Phone plus Lines	73	29	26	17	39	34
----------	91.2%	103.6%	96.3%	73.9%	100.0%	85.0%
Tablet plus Ereader	72	33	20	17	41	30
----------	90.0%	117.9%	74.1%	73.9%	105.1%	75.0%
Total Electronics	145	62	46	34	80	64
----------	181.2%	221.4%	170.4%	147.8%	205.1%	160.0%

If there are 2 or more similar variables which require the same subtotal they can be provided in a single DEFINE subcommand. Both the list of variable names and the list of numbers can contain individual values and ranges with “TO”. For example:

```
DEFINE 'Approve'     Q1 Q4 Q18 TO Q27   1 2,
DEFINE 'Disapprove'  Q1 Q4 Q19          3 TO 5,
DEFINE 'Disapprove'  Q18 Q20 TO Q17     3 4,
```

Figure 4.3 Nets with Multiple Response Variables

```
SURVEY Paceset,  LABELS 'Paceset.lab';

    DEFINE.NET <<Portable or landline phones>>       Phone   1 2,
    DEFINE.NET <<Tablet or Ereader>>                 Phone   3 4,
    DEFINE.NET <<Own 1 or more electronics>>         Phone   1 TO 4,

    BAN Age Sex, MR.STUB Phone TO Ereader, NO MISSING,  NO SUMMARY;
    $
```

	Total Sample	Under 30	30 to 50	Over 50	Male	Female
	======== Age =======				==== Sex ====	
Total Sample	80	28	27	23	39	40
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Which of the following do you own?						
Phone	45	19	13	12	27	18
	56.2%	67.9%	48.1%	52.2%	69.2%	45.0%
Lines	28	10	13	5	12	16
	35.0%	35.7%	48.1%	21.7%	30.8%	40.0%
Tablet	35	15	8	10	21	14
	43.8%	53.6%	29.6%	43.5%	53.8%	35.0%
Ereader	37	18	12	7	20	16
	46.2%	64.3%	44.4%	30.4%	51.3%	40.0%
Portable or landline phones	56	22	19	14	31	25
	70.0%	78.6%	70.4%	60.9%	79.5%	62.5%
Tablet or Ereader	53	23	15	13	28	24
	66.2%	82.1%	55.6%	56.5%	71.8%	60.0%
Own 1 or more electronics	69	26	21	20	34	34
	86.2%	92.9%	77.8%	87.0%	87.2%	85.0%

There is a limit on the number of variables and values that can be given in a single subcommand. (See the section on program limitations in a later chapter for details.) However, on most systems, 960 arguments are allowed for a single DEFINE or DEFINE.NET subcommand. The keyword DEFINE and the label each count as an argument. Each individual variable and each individual number count as a single argument. Thus "1 TO 10" in the argument list is counted as 10 arguments.

The DEFINE and DEFINE.NET statements that are needed for a given run may either be grouped together or placed separately with the relevant stub values.

```
DEFINE 'Likes'      Q1 Q2 Q113      1 2,
DEFINE 'Dislikes'   Q1 Q2 q113      4 5,
DEFINE 'Agrees'     Q19 Q33 TO q58  1 3 TO 5,
DEFINE 'Disagrees'  Q19 Q33 TO Q58, 2 6,
STUB Q19,
GR.STUB Q2 Q33 Q113, .....
```

or

```
DEFINE 'Likes'    Q1  1 2,
DEFINE 'Dislikes' Q1  4 5,
STUB Q1,
DEFINE 'Agrees'    Q19 Q33   1 3 TO 5,
DEFINE 'Disagrees' Q19 Q33   2 6,
STUB Q19 Q33, ....
```

NOTE: the semi-colon ";" which ends a tables group also affects the storage of the defines. DEFINEs are collected as the subcommands are entered until the ";" which indicates the beginning of a new tables group. If new DEFINE subcommands are entered after the semi-colon, any previous definitions are overwritten and are no longer available. The RESET SUBTOTALS subcommand can also be used to remove the existing definitions.

As always there are trade-offs. If you have hundreds of DEFINE subcommands, placing them all together may have a small impact on performance. If you have many variables with the same definitions, placing the DEFINEs within each tables group may require repetition of the define values. Therefore, use the method which is easiest for you to maintain.

4.3 Position Versus Value

Because nets are computed as the data are read, they are always based on the observed values of the variables in the P-STAT system file. However, the numbers used to define the *subtotals* refer to the *row position* in the table rather than the observed values of the data. This makes it possible, when used with RANK, to determine the highest and lowest groupings. (See the first table in figure 4.4). When RANK is not involved and the lowest observed value of a stub variable is a "1", there is no difference between a subtotal based on position and one based on the actual values of the variables. If the lowest data value is not a "1", there is a difference between the position and the value even in a table which does not require ranking. Therefore, it is always best to specify how subtotals are to be defined, The assumed setting is:

```
USE POSITIONS,
```

To change this so that values are the basis for subtotals:

```
USE VALUES,          or
SUBTOTALS VALUES,
```

The difference between these two subcommands is that the USE subcommand setting remains in effect until it is explicitly changed or the RESET subcommand resets it to POSITION. When the basis for subtotals is changed in the SUBTOTALS subcommand, it is temporary and the setting reverts to the current value of USE when the next SUBTOTALS subcommand is encountered.

Figure 4.4 SUBTOTALS: Position Versus Value

```
SURVEY Paceset,   LABELS “Paceset.lab”;
 MR.STUB Store.1 TO Store.4, BANNER Age,
 NO MISSING,  NO SUMMARY,  NO SKIP,
 DEFINE 'Most Sales'  Store.1    1 2,
 RANK DOWN,  SUBTOTALS  POSITION;
 $
                              ======== Age =======

                       Total  Under  30 to   Over
                      Sample     30     50     50

 Total Sample             80     28     27     23
                       100.0% 100.0% 100.0% 100.0%

 Type of store item purchased in:

 Department               58     30     14     13
                        72.5% 107.1%  51.9%  56.5%
 Web based source         48     14     21     11
                        60.0%  50.0%  77.8%  47.8%
 Discount                 39     18     11     10
                        48.8%  64.3%  40.7%  43.5%

 Most Sales              106     44     35     24
 ----------             132.5% 157.1% 129.6% 104.3%


 AGAIN, SUBTOTALS VALUE;

                              ======== Age =======

                       Total  Under  30 to   Over
                      Sample     30     50     50

 Total Sample             80     28     27     23
                       100.0% 100.0% 100.0% 100.0%

 Type of store item purchased in:

 Department               58     30     14     13
                        72.5% 107.1%  51.9%  56.5%
 Web based source         48     14     21     11
                        60.0%  50.0%  77.8%  47.8%
 Discount                 39     18     11     10
                        48.8%  64.3%  40.7%  43.5%

 Most Sales               97     48     25     23
 ----------             121.2% 171.4%  92.6% 100.0%
```

106 = 58 + 48. These are the two largest frequencies for the store variables.

97 = 39 + 58.

39 is the frequency for a value of 1 on the store variables.

58 is the frequency for a value of 2 on the store variables.

In file Paceset, the labels for the 3 store variables are:

```
(1) Discount (2) Department (3) Web based source,
```

In Figure 4.4, the first table, has the subtotals based on position so that the define:

```
DEFINE 'Most Sales' Store.1 1 2,
SUBTOTALS POSITION,
```

correctly identifies the 2 types of stores with the most sales and produces totals based on the rows after the ranking occurs. In the second table the subtotals are based on value.

```
SUBTOTALS  VALUE,
```

The subtotal row with the label “Most Sales” contains the sums of the rows associated with the values 1 and 2, not the sum of the 2 highest ranking rows.

4.4 Positioning Subtotals and Nets

If the subtotals are to be placed together in a subtotals section, that section must be part of any LAYOUT subcommand or the subtotals will not appear.

Figure 4.5 Placing the Subtotals in the Body of the Table

```
  SURVEY Paceset,  LABELS 'Paceset.lab';
  BANNER Age Sex,  STUB Num.TV,   NO MISSING,   NO SKIP,  NO SUMMARY,
  DEFINE  '1 or 2 TV sets'      Num.TV   1 2,
  DEFINE  '3 or more TV sets'   Num.TV   3 TO 5,
  SUBTOTALS BEFORE LOW VALUE;
$
 Number of television sets in your home:

                      ======== Age ======= ==== Sex ====

                Total  Under  30 to   Over
               Sample     30     50     50   Male Female

 Total            80     28     27     23     39     40
 Sample        100.0% 100.0% 100.0% 100.0% 100.0% 100.0%

 1 or 2 TV        28     18      8      2     22      6
 sets          35.0%  64.3%  29.6%   8.7%  56.4%  15.0%
 ---------
       1           6      4      1      1      4      2
                 7.5%  14.3%   3.7%   4.3%  10.3%   5.0%
       2          22     14      7      1     18      4
                27.5%  50.0%  25.9%   4.3%  46.2%  10.0%
 3 or more        52     10     19     21     17     34
 TV sets       65.0%  35.7%  70.4%  91.3%  43.6%  85.0%
 ---------
       3          18      8      5      4     11      7
                22.5%  28.6%  18.5%  17.4%  28.2%  17.5%
       4          27      1     13     12      4     22
                33.8%   3.6%  48.1%  52.2%  10.3%  55.0%
 5 or more         7      1      1      5      2      5
                 8.8%   3.6%   3.7%  21.7%   5.1%  12.5%
```

```
DEFINE “1 or 2 TV Sets”  Num.TV  1  2,
LAYOUT QUESTION TOTALS LABELS BODY SUBTOTALS
```

When the subtotals are integrated in the body of the table, as they are in Figure 4.5, the summary section is not needed. When subtotals are positioned in the body, the LAYOUT subcommand must include BODY if the subtotals are to print. The SUBTOTALS subcommand not only specifies whether VALUE or POSITION should be the interpretation of the numbers in the DEFINE subcommand, but also positions those subtotals in the table.

Figure 4.6 Multiple Response Stub with Subtotals

```
SURVEY Pacenew,  LABELS 'Paceset.lab';

   BAN Age Sex, MR.STUB phone TO ereader, NO MISSING,
   MARGIN 24,   SUBTOTALS BEFORE LOW,
   DEFINE 'Phone plus lines'                       Phone   1 2,
   DEFINE 'Portable Phone plus Answering Machine' Phone   3 4,
   DEFINE 'Total Purchases'                        Phone   1 TO 4;
```

		======== Age ========			==== Sex ====	
	Total Sample	Under 30	30 to 50	over 50	Male	Female
Total Sample	80	28	27	23	39	40
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Which of the following do you own?						
Phone plus lines	73	29	26	17	39	34
----------	91.3%	103.6%	96.3%	73.9%	100.0%	85.0%
Total Purchases	145	62	46	34	80	64
----------	181.3%	221.4%	170.4%	147.8%	205.1%	160.0%
Smart phone as primary	45	19	13	12	27	18
telephone	56.3%	67.9%	48.1%	52.2%	69.2%	45.0%
One or more landline	28	10	13	5	12	16
based telephones	35.0%	35.7%	48.1%	21.7%	30.8%	40.0%
Portable Phone plus	72	33	20	17	41	30
Answering Machine	90.0%	117.9%	74.1%	73.9%	105.1%	75.0%

One or more tablet	35	15	8	10	21	14
computers	43.8%	53.6%	29.6%	43.5%	53.8%	35.0%
One or more Ereaders	37	18	12	7	20	16
	46.3%	64.3%	44.4%	30.4%	51.3%	40.0%
Base	69	26	21	20	34	34
	86.3%	92.9%	77.8%	87.0%	87.2%	85.0%
Responses	145	62	46	34	80	64

If you wish to have the subtotals interleaved within the body of the table, there are two options. Subtotals may be placed before the low value or position for a given define or after the high value or position. If the SUBTOTALS subcommand does not specify VALUE or POSITION, the current setting of the USE subcommand determines the basis for the positioning.

```
SUBTOTALS BEFORE LOW,
SUBTOTALS AFTER HIGH,
```

Figure 4.7 Multiple Response Stub with Nets

```
SURVEY Paceset,  LABELS 'Paceset.lab';

   BAN Age Sex, MR.STUB VCR TO Ans.mach, NO MISSING,  NO SUMMARY,
   MARGIN 24,   SUBTOTALS BEFORE LOW VALUE,

   DEFINE.NET 'Own 1 or more of Phones to Ereader'  phone   1 TO 4,
   DEFINE.NET 'Phones or landline based telephones' phone   1 2,
   DEFINE.NET 'Tablet or Ereader'                   phone   3 4;
$
```

		======== Age =======			==== Sex ====	
	Total Sample	Under 30	30 to 50	Over 50	Male	Female
Total Sample	80	28	27	23	39	40
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Which of the following do you own?						
Own 1 or more of Phones to Ereader	69	26	21	20	34	34
	86.3%	92.9%	77.8%	87.0%	87.2%	85.0%
Phones or landline	56	22	19	14	31	25
	70.0%	78.6%	70.4%	60.9%	79.5%	62.5%
Smart phone as primary telephone	45	19	13	12	27	18
	56.3%	67.9%	48.1%	52.2%	69.2%	45.0%
One or more landline based telephones	28	10	13	5	12	16
	35.0%	35.7%	48.1%	21.7%	30.8%	40.0%
Tablet or Ereader	53	23	15	13	28	24
	66.3%	82.1%	55.6%	56.5%	71.8%	60.0%
One or more tablet computers	35	15	8	10	21	14
	43.8%	53.6%	29.6%	43.5%	53.8%	35.0%
One or more Ereaders	37	18	12	7	20	16
	46.3%	64.3%	44.4%	30.4%	51.3%	40.0%

Figure 4.6, which shows DEFINE with a multiple response stub, is very much like Figure 4.2 except for the positioning of the subtotals in the body of the table. Figure 4.7, which shows DEFINE.NET with a multiple response stub, is very much like Figure 4.3 except for the positioning of the nets in the body of the table.

The DEFINE statements in Figure 4.6 and the DEFINE.NET statements in Figure 4.7 are for the same values but the order is different.

```
DEFINE 'Phone plus lines'                       Phone   1 2,
DEFINE 'Portable Phone plus Answering Machine' Phone   3 4,
DEFINE 'Total Purchases'                        Phone   1 TO 4;

DEFINE.NET 'Own 1 or more of Phones to Ereader'  phone   1 TO 4,
DEFINE.NET 'Phones or landline based telephones' phone   1 2,
DEFINE.NET 'Tablet or Ereader'                   phone   3 4;
```

This difference explains why the order of the subtotals in Figure 4.6 is not the same as the order of the nets in Figure 4.7. When more than one net or subtotal falls at a particular location, the *order* in which they occur in the definitions is used to resolve the order in the table.

The argument "TOGETHER" used with SUBTOTALS indicates that the subtotals are to be placed not in the body but in the subtotals section as indicated by the LAYOUT subcommand.

```
SUBTOTALS TOGETHER,
SUBTOTALS TOGETHER VALUES,
SUBTOTALS TOGETHER POSITION,
```

SUBTOTALS TOGETHER POSITION is the assumed initial setting when DEFINE is used. SUBTOTALS TOGETHER VALUES is the assumed initial setting when DEFINE.NET is used. These initial settings can be changed by the USE subcommand.

Restating the difference between a DEFINE and a DEFINE.NET: A DEFINE totals responses. A DEFINE.NET totals respondents. In a DEFINE.NET a person is only counted once even if he has several responses. A DEFINE.NET is only appropriate in the multiple response situation. Because the nets must be tallied at the time that the data are first read into the table, they are always based on values: The eventual positions and ranks for a given row are not known until all the data are read.

When the multiple response stub is a group of dummy variables, tabulation is done on a single value, usually the 1's. Since the value is always the same, the values in the DEFINE and DEFINE.NET subcommands refer not to the single number that is counted but to the position of the variables in the MR.STUB list. The first variable in the MR.STUB list is used as the basis for the define. The numbers that follow point to the variables in that list.

```
MR.STUB Q1 Q5 Q2 TO Q4,
DEFINE.NET 'Q1 or Q5'  Q1   1 2,
DEFINE.NET 'Q2 to Q4'  Q1   3 to 5,
```

In the example above the first DEFINE.NET counts the number of respondents with a "1" on either Q1 or Q5 (the first two variables in the MR.STUB list). The second DEFINE.NET does the same for the respondents with a "1" on any of Q2, Q3, or Q4 (the 3rd, 4th and 5th variables in the MR.STUB variable list).

The only linguistic difference between the DEFINE and DEFINE.NET is the keyword used. The placement of the resulting rows is controlled in both situations by the LAYOUT and SUBTOTAL subcommands. See the chapter "SURVEY: Enhancing Layout and Appearance" for more about formatting the subtotals and nets.

4.5 Rank and Subtotal Interactions

There are 3 different ways to control the ranking when RANK is combined with either subtotals or nets.

1. Rank the rows as usual and let the nets or subtotals fall where they may, given the instructions in the SUBTOTALS command. This is appropriate when the subtotals are based on position but may well produce a scrambled table when subtotals are based on values.
2. Enter the DEFINE or DEFINE.NET subcommands in the order in which you wish them to occur and then have the rows ranked within them. Rows that are not defined appear after all the defined groups. Use RANK **WITHIN** SUBTOTALS/NETS.

3. Let the values of the subtotal or net groups control the order in which the subtotals or nets are to appear and then rank the rows within them. Rows that are not included in a define are placed as if they were a subtotal group with a single row. Use RANK **CONTROLS** SUBTOTALS/NETS.

The first table in figure 4.8 shows the command and the resulting table when the subcommands specify that the order of the net definitions controls the ranking. Since the definition for "birds" is first, that subtotal and its associated rows are first. The definition for "mammals is second" so that group appears next with its rows also ranked. The final group "other animals" has values for the other possible rows in the table. If the "other animals" define is omitted, the final three rows are in the same rank order but there is no "other animals" subtotal row. INDENT 4 is used to indent the rows in the body of the table 4 columns thus emphasizing the structure of the table.

Figure 4.8 Rank Within Subtotals

```
SURVEY Pets,  LABELS 'Pets.lab';

   DEFINE 'birds'          pet  1 6,
   DEFINE 'mammals'        pet  2 3 5,
   DEFINE 'other animals'  pet  4 7 8,

   BANNER Sex Age,  STUB Pet,  NO PERCENTS,  NO MISSING,  NO SUMMARY,
   MARGIN 16,  GAP 2,  NO SKIP,  INDENT 4,
   SUBTOTALS BEFORE LOW VALUE,
   RANK WITHIN SUBTOTALS;
   $
```

		==== sex ====		======== age =======		
	Total Sample	male	female	Under 20	20 to 39	40 and over
Total Sample	925	478	447	261	319	345
What animal makes the best pet?						
birds	155	100	55	28	48	79
-----						
parakeet	80	25	55	16	25	39
canary	75	75	-	12	23	40
mammals	675	325	350	220	230	225
-------						
cat	350	150	200	100	110	140
dog	275	150	125	100	100	75
horse	50	25	25	20	20	10
other animals	95	53	42	13	41	41
-------------						
fish	90	49	41	9	40	41
turtle	4	3	1	3	1	-
snake	1	1	-	1	-	-

A partial rank can be used with subtotals to exclude 1 or more rows from the ranking. A positive number after the RANK subcommand indicates the number of rows to rank. A negative number requests that all but the last n (where n is the number) rows be included in the ranking.

```
RANK -2 DOWN WITHIN SUBTOTALS,
RANK  6 DOWN WITHIN SUBTOTALS,
```

When a partial rank is done, the rows that are to be excluded are excluded first before any decisions are made about ranking. The excluded rows always appear at the end of the body and they are always in their natural order.

The Figure 4.9 illustrates RANK CONTROLS SUBTOTALS, the third way to organize ranks within subtotals. Since there are more mammals than either of the other defined groups, that define group appears first in rank order. Any rows that are not included in a define groups appear not at the end but in the position that they would have if they were declared a define group by themselves.

Figure 4.9 Rank Controls Subtotals

```
AGAIN, RANK CONTROLS SUBTOTALS;
```

		sex		age		
	Total Sample	male	female	Under 20	20 to 39	40 and over
Total Sample	925	478	447	261	319	345
What animal makes the best pet?						
mammals	675	325	350	220	230	225
-------						
cat	350	150	200	100	110	140
dog	275	150	125	100	100	75
horse	50	25	25	20	20	10
birds	155	100	55	28	48	79
-----						
parakeet	80	25	55	16	25	39
canary	75	75	-	12	23	40
other animals	95	53	42	13	41	41
-------------						
fish	90	49	41	9	40	41
turtle	4	3	1	3	1	-
snake	1	1	-	1	-	-

Figure 4.10 shows how the table looks if “dog” is left as a category by itself. Because there are more votes for dog than for all the birds, but fewer than there are for cats and horses combined, dogs appear between the two groups.

When either RANK WITHIN or RANK CONTROLS is used the rows in the table should not be included in more than 1 DEFINE subcommand. The subtotals will be correct but the placing of the subtotals within the table will not be appropriate if any of the rows is in multiple defines.

Normally when INDENT is used all labels for the rows of the body are indented the specified amount. However, when RANK CONTROLS the subtotals or nets, it is often the case that the rows in the body that are undefined are better displayed aligned with the subtotals than with the other rows in the body.

```
INDENT DEFINED VALUES 4
```

limits the domain of INDENT to those rows that are represented in a define. In Figure 4.10, the row labelled "dog" would have been indented with the other body rows except that it is not included in any of the DEFINEs *and* the INDENT DEFINED VALUES subcommand is used.

If the subtotal or net has no cases it is usually squeezed out of the printout. This can be changed by using:

```
FILL SUBTOTALS,
```

SQUEEZE SUBTOTALS is the assumed setting. When this subcommand is used, the subtotal will be included only if it has at least a single row with good data.

Figure 4.10 Placement of Undefined Values

```
SURVEY Pets,  LABELS 'Pets.lab';
     DEFINE 'birds'          pet 1 6,
     DEFINE 'cats & horses'  pet 2 5,
     DEFINE 'other animals'  pet 4 7 8,

     BANNER Sex Age,  STUB Pet,
     NO PERCENTS,  NO MISSING,  NO SUMMARY,
     MARGIN 16,  GAP 2,  NO SKIP,

     INDENT DEFINED VALUES 4,
     SUBTOTALS BEFORE LOW VALUE,
     RANK CONTROLS SUBTOTALS;
     $
```

```
                          ==== sex ====   ======== age =======

                  Total                   Under  20 to 40 and
                 Sample     male female       20     39   over

Total Sample        925      478    447      261    319    345

What animal makes the best pet?

cats & horses       400      175    225      120    130    150
-------------
    cat             350      150    200      100    110    140
    horse            50       25     25       20     20     10
dog                 275      150    125      100    100     75
birds               155      100     55       28     48     79
-----
    parakeet         80       25     55       16     25     39
    canary           75       75      -       12     23     40
other animals        95       53     42       13     41     41
-------------
    fish             90       49     41        9     40     41
    turtle            4        3      1        3      1      -
    snake             1        1      -        1      -      -
```

Figure 4.11 COMBINE subcommand

```
SURVEY Pets,  LABELS 'Pets.lab';
  BANNER Sex Age,  STUB Pet,  NO PERCENTS,  NO MISSING,  NO SUMMARY,
  MARGIN 16,  GAP 2,  NO SKIP,
  COMBINE 'Others' .01;
```

	======== age =======				==== sex ====	
	Total Sample	Under 20	20 to 40 and 39	over	male	female
Total	925	261	319	345	478	447
Sample	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
What animal makes the best pet?						
cat	350	100	110	140	150	200
	37.8%	38.3%	34.5%	40.6%	31.4%	44.7%
dog	275	100	100	75	150	125
	29.7%	38.3%	31.3%	21.7%	31.4%	28.0%
fish	90	9	40	41	49	41
	9.7%	3.4%	12.5%	11.9%	10.3%	9.2%
parakeet	80	16	25	39	25	55
	8.6%	6.1%	7.8%	11.3%	5.2%	12.3%
canary	75	12	23	40	75	-
	8.1%	4.6%	7.2%	11.6%	15.7%	
horse	50	20	20	10	25	25
	5.4%	7.7%	6.3%	2.9%	5.2%	5.6%
Others	5	4	1	-	4	1
	0.5%	1.5%	0.3%		0.8%	0.2%

4.6 COMBINING SMALL FREQUENCIES DYNAMICALLY

When a stub variable or a MR.STUB group of variables has a very large range, it is often necessary to print only those rows which have the larger frequencies and to combine into a single category those rows which are of less interest because they are smaller. This can be done by using the P-STAT programming language, examining the frequencies with the COUNTS command and then constructing the appropriate recode. However, this is cumbersome especially when the totals change from table to table with changes in the options and the banner variables.

The COMBINE subcommand provides a way to do this type of recoding dynamically as the tables are being printed. There are also provisions for some special features:

1. Individual rows can be excepted from the combinations either by value or value label

2. The threshold test can be done on the row totals or on all the values in the row
3. The threshold test can be done on a subtotal to ensure that large subtotals survive even if all of their rows have failed the threshold test.
4. A single row can be forced into the combined category even if it survives the threshold test. This permits a previously created “others” category to be forced into the combination whatever its size.
5. There can be combinations within subtotals as well as the overall combination for the table.

The format of the COMBINE subcommand is:

```
COMBINE  'Others'  .01,
COMBINE  'Others'   4,
```

The COMBINE subcommand is followed by the text *in quotes* which is to label the final combined row. The test number follows. It is either a fractional value to indicate a percentage or an integer to indicate an absolute count. Any row in the table with a row total that is less than or equal to the threshold test value does not appear as an individual row. The data from those rows are combined into a single row which is given the appropriate label.

Figure 4.11 illustrates the subcommand and printout when COMBINE is given a threshold value equal to 1 percent (.01). of the total count of the Pets file. Both “snake” and “turtle” have percentages that do not surpass the threshold. Therefore, they are combined to make the “Others” category.

COMBINE when used by itself combines all the rows in the table with values less than or equal to the threshold. Sometimes a row gets combined which you would like left alone. For example, if the Pets survey were conducted under the auspices of The Society for the Preservation of Snakes, the row with the information about snakes as pets should be included even when it does not survive the threshold test. This is done by using the EXCEPT subcommand which permits you to supply either values or labels for the stub variable which identify the rows that are always to be included. In figure 4.12 the EXCEPT subcommand is given the label “snake” rather than the number 7 which is its value in the table. The following are equivalent statements:

```
EXCEPT 7,
EXCEPT 'snake'
```

When the stub variable is a character variable, it is easier to use labels than values because the value is often unknown.

In Figure 4.12, the threshold value of 6 percent (.06) combines horse and turtle, but because of the exception, snake with the smallest percent of all remains in the table.

EXCEPT is followed by a list of the values or labels that are to be excepted. This list usually comprises only the values but it may include labels if they are in quotes. Both values and labels may be used in a single EXCEPT subcommand.

```
EXCEPT 3 'Some label' 7 TO 9 'canary',
```

A row is usually combined if the row totals do not exceed the threshold value even if some of the cells are above that threshold. It may be desirable to combine a row only when the row total *and* all of the cells in that row *fail* the threshold test. In the Pets file there are 2 cells for the row labelled “horse”, which are above the threshold given in Figure 4.12. Since the row totals are below the threshold of .06, the row will be combined unless the COMBINE request includes the keyword “ALL” as its second argument:

```
COMBINE ALL “other” .06,
```

The combine row is usually a total of all the rows which do not satisfy the test requirements. It is possible to expand this definition so that a single row which may pass the threshold test is added to the combined row. This is often done when you have previously combined cases (usually using P-STAT’s PPL) which are not of interest in this study but which may exceed the combine test threshold. You now wish to combine this single row of little interest with the dynamically combined “others” row. This is done by adding an argument immediately after the label which contains either the value or the value label for the row that is to be forced into the combined row.

Figure 4.12 COMBINE with EXCEPT

```
   AGAIN,  COMBINE 'others' .06,  EXCEPT 'snake';


                      ======== age ======= ==== sex ====

                Total  Under  20 to 40 and
               Sample     20     39   over   male female

Total            925    261    319    345    478    447
Sample         100.0% 100.0% 100.0% 100.0% 100.0% 100.0%

What animal makes the best pet?

cat              350    100    110    140    150    200
                37.8%  38.3%  34.5%  40.6%  31.4%  44.7%

dog              275    100    100     75    150    125
                29.7%  38.3%  31.3%  21.7%  31.4%  28.0%

fish              90      9     40     41     49     41
                 9.7%   3.4%  12.5%  11.9%  10.3%   9.2%

parakeet         80     16     25     39     25     55
                 8.6%   6.1%   7.8%  11.3%   5.2%  12.3%

canary           75     12     23     40     75      -
                 8.1%   4.6%   7.2%  11.6%  15.7%

snake             1      1      -      -      1      -
                 0.1%   0.4%                 0.2%

others           54     23     21     10     28     26
                 5.8%   8.8%   6.6%   2.9%   5.9%   5.8%
```

```
     COMBINE 'other' 99 .02,
     COMBINE 'other' 'Not Applicable' .02,
     COMBINE ALL 'other' 99 .04,
```

The row that is to be combined regardless of the number of cases or the percentage is specified either by the value or the label. However, if the label is used, it must be no more than 16 characters long. Only a single row may be combined in this way.

4.7 Combine, Except and Subtotals

When COMBINE is used with subtotals, any subtotal which has no rows that meet the combine criterion is automatically omitted from the table. There are two ways that this can be controlled. The first way tests the value for the subtotal as if it were a row. If the subtotal survives the combine threshold it is included in the table. The subcommand is:

```
     OPTION TEST SUBTOTALS
```

Figure 4.13 Some of the SUBTOTAL Options

```
DEFINE 'one and two '   Var1  1 2,
DEFINE 'three and four' Var1  3 4,

NO MISSING, NO SUMMARY,
STUB Var1 ( 1 4 ),

SUBTOTALS BEFORE LOW VALUES;


Total             2
Sample        100.0%

VAR1

one and two       2
-----------   100.0%

       1          1
                50.0%

       2          1
                50.0%
```

```
AGAIN, FILL SUBTOTALS;


Total             2
Sample        100.0%

VAR1

one and two       2
-----------   100.0%

       1          1
                50.0%

       2          1
                50.0%

three and         -
four
---------
```

```
AGAIN,
RANK DOWN CONTROLS SUBTOTALS,
COMBINE 'Other' 1;

Total             2
Sample        100.0%

VAR1

Other             2
              100.0%
```

```
AGAIN, OPTION TEST SUBTOTALS;

Total             2
Sample        100.0%

VAR1

one and two       2
-----------   100.0%

      2           1
                50.0%

      1           1
                50.0%
```

```
AGAIN, EXCEPT 2;


Total             2
Sample        100.0%

VAR1

one and two       2
-----------   100.0%

       2          1
                50.0%

Other             1
                50.0%
```

The second way forces all subtotals to print even if they have no rows which survive the combine threshold. This is done by using the INCLUDE ALL SUBTOTALS subcommand.

```
DEFINE 'Turtles and Snakes'  pet 7 8,
COMBINE 'other' .01,
INCLUDE ALL SUBTOTALS,
```

INCLUDE ALL SUBTOTALS forces a subtotal to print that has some rows with non-missing data. This is not the same as FILL SUBTOTALS which forces the printing of a subtotal that has never had any rows.

In the Pet data set, neither snakes nor turtles accounted for the required 1 percent. Thus the subtotal “Turtles and Snakes” is eliminated from the printout unless the INCLUDE ALL SUBTOTALS subcommand is used. If any row is included in a subtotal because of an exception, the subtotal will always print.

```
DEFINE 'Turtles and Snakes'  pet 7 8,
COMBINE 'other' .01,
EXCEPT  'Snake',
```

The setting which includes all the subtotals can be turned off by using the default setting:

```
EXCLUDE SMALL SUBTOTALS;
```

The variety of subcommands which control the appearance of subtotals or nets and their interaction is often best explained by examples. Figure 4.13 contains 5 small examples with a data set of 2 values, a '1' and a '2'. There are two DEFINE subcommands, one for values that are not even in the data file.

1. SUBTOTALS BEFORE LOW VALUES produces a table with a single subtotal preceding the two data values.
2. FILL SUBTOTALS produces a table like the first except that the second define is now printed even though it has no cases.
3. RANK DOWN CONTROLS SUBTOTALS, COMBINE 'Other' 1, produces a table with a single row with the label 'Other'. Neither of the cases survives the combine threshold so neither row prints and the subtotal is omitted. The FILL SUBTOTALS setting has no effect except in the simple table without ranking so the empty subtotal is also ignored.
4. OPTION TEST SUBTOTALS is added. The subtotal passes the threshold so even though both of the rows have failed, it is treated as a regular subtotal with rows that pass the threshold. Both rows and subtotals print.
5. EXCEPT 2 is added. It requests that the second row be exempted from the COMBINE threshold test. The subtotal is now treated as if it were a normal subtotal with one row that passed the combine threshold and one row that failed. The failed row prints as part of the combined 'Others' row.

4.8 Dynamic Combines Within Subtotals

A further ramification of COMBINE and subtotals permits dynamic combination at the subtotal/net level. It requires the combination of two subcommands:

```
COMBINE 'label for overall other' .01,
OPTION LABEL COMBINED SUBTOTALS 'label for subtotal others',
```

The combinations for each of the subtotals follow the same rules, including exception values, that apply to the overall COMBINE. In addition, any rows that do not belong to a subtotal are subject to the overall combine group.

It is possible to specify a different label for each subtotal by adding the text as a second argument in the DEFINE statement.

```
DEFINE 'birds' 'other birds' pet 1 7 8
```

Figure 4.14 Labelling Combined Subtotals

```
DEFINE 'birds' 'other birds' pet 1 7 8 ,
DEFINE 'mammals' pet 2 3 5 6,
DEFINE 'other animals' 'yet other animals' pet 4 10 11,

EXCEPT 'snake',
COMBINE 'other breeds' 2,
OPTION LABEL COMBINED SUBTOTALS 'other',

BAN Sex, STUB Pet,
NO PERCENTS, NO SKIP, NO MISSING,
NO SUMMARY, SUBTOTALS BEFORE LOW VALUE,

RANK WITHIN SUBTOTALS,
INDENT DEFINED VALUES 2,
MARGIN 20;
$
```

		==== sex ====	
	Total Sample	male	female
Total Sample	28	16	12
What animal makes the best pet?			
birds	7	4	3
-----			
parakeet	3	1	2
canary	3	3	-
other birds	1	-	1
mammals	13	7	6
-------			
cat	4	2	2
horse	4	2	2
dog	4	2	2
other	1	1	-
other animals	6	4	2
-------------			
fish	3	2	1
snake	1	1	-
yet other animals	2	1	1
other breeds	2	1	1

Any rows in a subtotal that meet the combination criteria are combined into a single row which prints following all the other rows in the subtotal. These combined rows are labelled with the text that follows the OPTION LABEL subcommand unless the DEFINE statement provides a unique label for them.

In Figure 4.14, which uses an expanded version of the Pets file, there is no specific label in the DEFINE for the subtotal “mammals”. Since there are rows belonging to this subtotal which do not meet the combine threshold of 2, they are combined and given the label associated with the OPTION LABEL COMBINED SUBTOTALS subcommand. There are several rows which are not part of any DEFINE. They all fail the combine threshold and are given the label from the COMBINE command itself.

This subcommand is best used either with RANK WITHIN SUBTOTALS or RANK CONTROLS SUBTOTALS. If this feature is used without RANK and the values within the subtotals are not adjacent rows, the combined values may be placed in strange places and may include rows from other subtotals.

SUMMARY

SURVEY

```
SURVEY Paceset,  LABELS 'Paceset.lab';
    DEFINE <<Few Television sets>>  Num.TV 1 2,
    DEFINE <<More Television sets>> Num.TV 3 TO 5,
    BAN Age Own,  STUB Num.TV ( 1 5 ),
    MR.STUB Phone to Ereader;
```

The SURVEY command has variations in the way that rows can be subtotaled or netted and the results displayed.

Optional Subcommands

OPTIONAL Subcommands: Controlling the Row Contents

COMBINE 'cs' nn

COMBINE provides a label and a threshold test. Rows which do not exceed that threshold are combined into a single row. The first argument, a string in quotes, provides the label. The second argument is either a fraction, which indicates that the threshold is to be based on a percent, or an integer, which indicates that the threshold is to be based on the frequency itself.

```
COMBINE 'Others' .01,
```

Any single row which might not meet the test requirements can also be combined into the other category by providing either its value or its value label.

```
COMBINE 'Others' 99 .01,
COMBINE 'Others' 'extras', .01,
```

DEFINE 'cs' vn nn nn

specifies subtotal groupings of a stub variable. The general pattern of a DEFINE subcommand is:

```
DEFINE    'Subtotal Name'  Variables(s)  Positions,
```

The name or label for the subtotal must be enclosed in quotes. The *positions* (rows) of the specified stub variable are added together to get the value of the subtotal. In the following example, the frequencies or counts in rows 1 through 8 are summed to get the value of the subtotal “GM”,

```
STUB  Car.Model,  DEFINE  <<GM>>    Cars  1 TO 8,
                  DEFINE  <<Ford>>  Cars  9 11 TO 15,
```

and similarly, rows 9 and 11 through 15 are summed for the second subtotal “Ford”. When there are multiple stub variables, multiple DEFINE variables may be specified – the subtotal definition applies to all that are specified.

Subtotals are located after the BODY of the survey, unless the SUBTOTALS subcommand repositions them. If the subtotals are not repositioned, any LAYOUT subcommand must include the SUBTOTALS keyword or they will not print.

If the USE VALUES subcommand has been specified, the *values* of the DEFINE variable are totaled, instead of the positions.

```
USE VALUES,
DEFINE    'Subtotal Name'  Variables(s)  Values,
```

See also the SUBTOTALS subcommand. Supplying print levels for the subtotals is discussed in the chapter “SURVEY: Enhancing Table Appearance”.

```
DEFINE <<Define Label>> <<Subtotals Combine Label>>  vars values
```

A second character string label is used only if both COMBINE and OPTION LABEL subcommands are provided. It then becomes the label for any rows that are combined within that subtotal.

DEFINE.NET 'cs' 'cs' vn nn nn

is similar to DEFINE except that it may only be used with multiple response variables and a DEFINE.NET provides totals of the respondents rather than the responses. DEFINE.NET is always based on value while the assumption for a DEFINE is position. See the USE subcommand to change these defaults.

EXCLUDE SMALL SUBTOTALS

is the assumed setting for COMBINE. This means that the subtotals will not print if all of the rows in that subtotal or net are combined into the “other” category.

EXCEPT nn nn 'cs'

provides a list of values which are to be excepted from any COMBINE threshold. The list can have either values. If labels are used they must be in quotes. If a value is excepted, any subtotal or net to which it belongs will be printed.

FILL SUBTOTALS

requests that subtotals and nets be printed even when there are no possible rows for that subtotal or net.

INCLUDE ALL SUBTOTALS

specifies that all subtotals are to print even if there are no rows left because they have been combined into an “others” category. However, subtotals that never had any cases will not print. See FILL SUBTOTALS.

INDENT nn

requests that the stub value labels be indented the specified number of spaces within the margin area. When indent is not used the labels are flush left to the margin. This does not apply to nested tables which have their own indent structure.

```
INDENT DEFINED VALUES 6,
```

when used with subtotals or nets causes the labels for any values that have been included in a subtotal or net to be indented 6 spaces. Any value that is not defined has its label placed in the normal position.

NETS arg arg

The NETS and SUBTOTAL subcommands are identical and can be used interchangeably. Even though there is a real difference between them computationally, the only difference between them at subcommand level occurs in the DEFINE and DEFINE.NET subcommands. See SUBTOTALS for further details.

OPTION LABEL COMBINED SUBTOTAL 'cs'

requests that dynamic combinations be done within subtotal groups as well as for the table as a whole. The COMBINE subcommand provides the threshold for testing rows. EXCEPT may also be used with this option.

OPTION TEST SUBTOTALS

requests that the subtotals as well as the rows be tested for the combine threshold. Any subtotal which survives the threshold test is included in the table even if none of the individual rows passes the threshold test. If a subtotal is accepted on this basis, all its rows are also accepted.

RESET SUBTOTALS

resets the count of subtotals and nets to zero. Previous definitions are no longer available

RANK DOWN / UP

requests that the rows in the body of the survey print in descending order of the row totals. RANK UP may be used to request the opposite – that the rows print in ascending order of the row totals. When RANK is not used, stub variables typically print in ascending order of their values (codes). RANK may be followed by an integer to request that only that many rows of the survey be ranked:

```
RANK  5  UP,
```

Rows beyond the rank limit are printed in their normal positions. When RANK is followed by a negative integer, the final n rows are excluded from the ranking.

RANK interacts with DEFINE and SUBTOTALS – ranking occurs *before* subtotaling. Typically, when RANK is used, subtotals are computed by position (position after ranking). This permits subtotals of the largest rows, for example:

```
RANK,  DEFINE  <<Top Three>>  Stores  1 TO 3,  USE POSITIONS;
```

The order in which the ranks occur when there are subtotals can be controlled by using one of the following:

```
RANK  WITHIN    SUBTOTALS / NETS  or
RANK  CONTROLS  SUBTOTALS / NETS
```

In both situations, the ranking is done for each define or define.net. RANK WITHIN SUBTOTALS uses the order of the defines to determine the order of the rank groups. RANK CONTROLS SUBTOTALS uses the frequency of the subtotals to determine the order. The subtotal or net with the largest count and its rows precede the subtotal with the next largest count and its rows, etc. The order of the rank and partial rank controls may be specified at the same time.

```
RANK 5 DOWN WITHIN SUBTOTALS,
```

SQUEEZE SUBTOTALS

squeeze out empty subtotals is assumed. FILL SUBTOTALS may be used to change this setting.

SUBTOTALS arg arg

specifies two things: 1) the placement of the subtotals created by DEFINE subcommands, and 2) optionally, the method that the DEFINE uses to subtotal. (See the DEFINE subcommand.) The arguments that may follow SUBTOTALS are:

```
AFTER HIGH           BEFORE LOW           TOGETHER
AFTER HIGH POSITION  BEFORE LOW POSITION  TOGETHER POSITION
AFTER HIGH VALUE     BEFORE LOW VALUE     TOGETHER VALUE
```

The keywords AFTER HIGH, BEFORE LOW or TOGETHER specify the *location.* The keyword POSITION or VALUE specifies the *method.* When the method is not provided, the current USE setting determines the method.

When POSITION is specified as the method, the numeric list in the DEFINE specifies which *rows* to total; when VALUE is used, the numeric list in the DEFINE specifies the *values* to total.

When the SUBTOTALS subcommand is not used, SUBTOTALS TOGETHER is assumed. If the current USE value is position, the DEFINE subtotals by rows or positions, and the subtotals are placed together according to the LAYOUT specifications. If this is a net (DEFINE.NET) or if the current USE value is VALUES, the subtotal/net combinations are determined by the actual values of the stub variables. NO SUBTOTALS or SUBTOTALS OFF cancels any subtotal specifications.

USE VALUES/POSITIONS

Specifies the way that the numbers in the DEFINE subcommand are to be interpreted. POSITION is most frequently used with RANK to give subtotals which depend on the ranking of the rows. The USE setting remains in effect until another USE subcommand or a RESET command. RESET returns it to USE POSITIONS which is the assumed setting for subtotals.

5
SURVEY: Special Features of the Layout Sections

This chapter covers various subcommands which permit fine-tuning of the appearance and contents of the table. There are controls for pagination. Each of the table sections has controls that apply to just that section. Subcommands which apply to the table as a whole such as the SURVEY.LABELS feature and camera ready output using PostScript are covered in the next chapter.

5.1 PAGE CONTROLS

A physical page and a logical page are not necessarily the same. If the table has so many lines of information that it will not fit on a single piece of paper, it is usually continued without interruption on as many sheets as necessary and is considered logically a single "page". Within P-STAT the size of a physical page is defined by the LINES setting. This usually reflects the actual size of the paper. In the SURVEY command, the behavior when a "page" is full (as indicated by the LINES setting) depends on the BREAK and PAGE subcommands.

Usually a table prints continuously until it is complete. TITLES are printed once at the top of the first page and at the bottom of the last page. Page numbers, if they are used, increase with each physical page as determined by the LINES setting. When BREAK is used, a page change also occurs when the LINES setting is exceeded. This prevents the table from crossing the crease in continuous paper or printing so near the edges that information is lost. Each table, however, is still considered a single logical page and the titles (top and bottom) print only once.

The PAGE subcommand by itself causes each physical page (as defined by the LINES setting) to be treated as a separate logical page. When the LINES setting is exceeded, the page number increases, titles (if defined) are printed, and the LABELS, QUESTION and TOTALS.AREA sections are repeated in the same order that they are given in the LAYOUT. In addition, the PAGE subcommand can be used to selectively repeat table sections. The sections that can be repeated are the QUESTION, the LABELS, and the TOTALS.AREA. The format of the subcommand is:

```
PAGE,
PAGE  LABELS,
PAGE  QUESTION  LABELS,
PAGE  QUESTION  TOTALS  LABELS,
```

Any of the three arguments can be used in any order. The order indicates the order of these sections on the continuation pages, regardless of the order specified in the LAYOUT subcommand:

```
LAYOUT LABELS TOTALS QUESTION BODY SUMMARY,
PAGE QUESTION TOTALS,
```

causes the first page of the survey to have a different arrangement of the sections than any subsequent pages. Usually the table looks better if the order of the arguments for the PAGE subcommand is the same as the order of those sections in the LAYOUT subcommand.

```
PAGE QUESTION TOTALS 2,
```

requests that page change include titles, the question and the totals area. However, only the first 2 lines of the extended label which comprises the question are to be used on the continuation pages. This feature is discussed more fully later in this chapter.

Figure 5.1 Page Numbers and Page Break

```
  TITLE  B2 'Page .PAGE.' $
  SURVEY Paceset,  LABELS 'Paceset,lab',
    TITLES,  PAGE.NUMBER 22,  LINES 24 ;
    BANNER  Age Own,  STUB Num.TV,
    ROW COLUMN PERCENTS,  BREAK;
```

		======= Age =======			Respondent ownership	
	Total Sample	Under 30	30 to 50	Over 50	no	yes
Total	80	28	27	23	11	69
Sample	100.0%	35.0%	33.8%	28.7%	13.8%	86.2%
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Number of television sets in your home:						
1	6	4	1	1	1	5
	100.0%	66.7%	16.7%	16.7%	16.7%	83.3%
	7.5%	14.3%	3.7%	4.3%	9.1%	7.2%
2	22	14	7	1	3	19
	100.0%	63.6%	31.8%	4.5%	13.6%	86.4%
	27.5%	50.0%	25.9%	4.3%	27.3%	27.5%
3	18	8	5	4	3	15
	100.0%	44.4%	27.8%	22.2%	16.7%	83.3%
	22.5%	28.6%	18.5%	17.4%	27.3%	21.7%

..... (lines 24 causes a page break here)

4	27	1	13	12	3	24
	100.0%	3.7%	48.1%	44.4%	11.1%	88.9%
	33.8%	3.6%	48.1%	52.2%	27.3%	34.8%
5 or more	7	1	1	5	1	6
	100.0%	14.3%	14.3%	71.4%	14.3%	85.7%
	8.8%	3.6%	3.7%	21.7%	9.1%	8.7%
Missing	-	-	-	-	-	-
Base	80	28	27	23	11	69
	100.0%	35.0%	33.8%	28.7%	13.8%	86.2%
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Mean	3.09	2.32	3.22	3.83	3.00	3.10
S.D.	1.13	0.90	1.01	0.98	1.18	1.13

Page 23

Figure 5.1 shows the command and the printout when BREAK is used with PAGE.NUMBER. The lines setting of 24 causes the table to break in the middle. The line of dots indicates where the page change occurs. The bottom title prints only at the end of the table. A similar control for the lines of QUESTION text is covered later in "The QUESTION Section".

Any titles that are defined either in the titles command or with the TITLES subcommand print on every page of a table. If you do not want all of the titles on the continuation pages, a second PAGE command can be used to provide the number of top title lines that are to print on all pages except the first page.

```
PAGE QUESTION LABELS,
PAGE TITLES 1,
```

causes continuation pages to begin with just the first line of top titles. If you want page changes to include just the full or partial titles and to leave out the labels, totals, question and other heading information use:

```
PAGE TITLES ONLY,
PAGE TITLES ONLY 1,
```

If there are table titles, they usually print at every page change following any top titles. The PAGE subcommand can be used to specify how many of the table titles should print on continuation pages. The following example causes continuation pages to have top title 1 and the first line of the table titles.

```
PAGE TITLES ONLY 1,  PAGE TABLE.TITLES 1,
```

All three varieties of the PAGE subcommand can be used together:

```
PAGE QUESTION 1,
PAGE TITLES 1,
PAGE TABLE.TITLES 0,
```

causes each continuation page to begin with the first top title, followed by the first line of the question section.

The PAGE TITLES subcommand can be used in conjunction with a USE TITLES subcommand for yet greater control.

```
USE TITLES FIRST 1 3,
USE TITLES CONTINUED 3 4,
```

This subcommand instructs SURVEY which titles are to be printed on the FIRST page of a table and which titles are to be printed on any continuation pages. The numbers are a range. In this example the first page will have any defined titles from T1 through T3. All continuation pages will have titles T3 and T4 printed.

A second pair of ranges can be used for bottom titles.

```
USE TITLES FIRST 1 3 1 1,
USE TITLES CONTINUED 3 4 2,
```

With this control, the first page will also have the first bottom title. Continuation pages will have bottom title B2 printed and because the end of the range is not supplied, B3 (if defined) will also print. Thus

```
USE TITLES FIRST 3,
```

causes all defined titles starting with title T3 to be printed on the first page.

Titles can be turned off by using zero (0) as both the start and end of the range.

```
USE TITLES FIRST 3 5 0 0,
USE TITLES CONTINUED 0 0 2 3,
```

The use of the USE command is temporary. The titles are still defined and can be reinstated within the SURVEY by another USE subcommand.

5.2 THE SECTIONS OF A TABLE

Each of the seven sections of the layout can be controlled for contents or appearance. The seven names can be used as arguments following the LAYOUT subcommand or as subcommands in their own right. The sections are:

1. QUESTION
2. LABELS
3. TOTALS.AREA
4. BODY
5. SUBTOTALS
6. MISSING
7. SUMMARY

The basic properties of these sections and their use in the LAYOUT subcommand are covered in the chapter "SURVEY: Creating Simple Tables". This chapter covers each section in detail and describes additional ways to change the contents and to enhance the appearance of the tables.

5.3 The QUESTION Section

The QUESTION section of the table usually contains the variable name or the extended labels of the stub variable. The placement is determined by the LAYOUT command. If LAYOUT is not used, the QUESTION section is placed between the totals area and the body of the table; To place the question at the top of the page below any top titles use a LAYOUT such as:

```
LAYOUT QUESTION LABELS TOTALS BODY ... ,
```

The question is usually left justified. The QUESTION CENTER subcommand is used to center the question on the page:

```
QUESTION CENTER,
```

"QUESTION LEFT" is used to reset the location of the question text to the assumed location at the left edge of the printout.

When there is a multi-line extended label for a stub variable, the entire text prints, one line per label, on each "page" of the printout. There is no limit on the number of lines of extended labels that can appear, and, in fact, there is no reason why the question section could not fill an entire page. In such a situation, it may be useful to have a limit on the number of lines that appear on continuation pages. This number is supplied not in the QUESTION subcommand but in the PAGE subcommand.

The QUESTION section can be combined with the LABELS section by using the subcommand:

```
QUESTION IN LABELS.AREA
```

When this is used, the contents of the QUESTION area are placed in the margin of the LABELS section. The length of the question text depends on the width of the MARGIN and on the number of lines used to print the variable labels and the value labels.

```
                              ======== Age =======

                     Total  Under  30 to   Over
 Sex                Sample     30     50     50
```

The subcommands that produced this printout were:

```
BANNER Age, STUB Sex, QUESTION IN LABELS.AREA;
```

This subcommand may be used with PostScript or tabbed output but is not supported for transposed tables.

Figure 5.2 Extended Labels and the QUESTION Section

Labels File Xl.lab

```
Var1  'this is the first xl '  '&&'   'this is the second'
      'this is the third xl'   '& &'  'this is the fourth'
      'this is the final xl'
     (1) 'one' (2) two /
```

```
TITLES 'Page .PAGE.' $

SURVEY Small,  LABELS  'Xl.lab',  LINES 16,
               TITLES,  PAGE.NUMBER 39;
  LAYOUT QUESTION BODY SUMMARY,
  BAN Var2, PAGE QUESTION 1,
  ROW COL TOT PERCENTS;
```

Page 39

```
this is the first xl this is the second
this is the third xl this is the fourth
this is the final xl

one               6      1      1      1      -      -      -      1
              100.0%  16.7%  16.7%  16.7%                        16.7%
               66.7% 100.0% 100.0% 100.0%                       100.0%
               66.7%  11.1%  11.1%  11.1%                        11.1%

two               3      -      -      -      1      1      1      -
              100.0%                       33.3%  33.3%  33.3%
               33.3%                      100.0% 100.0% 100.0%
               33.3%                       11.1%  11.1%  11.1%
```

Page 40

```
this is the first xl this is the second

Base              9      1      1      1      1      1      1      1
              100.0%  11.1%  11.1%  11.1%  11.1%  11.1%  11.1%  11.1%
              100.0% 100.0% 100.0% 100.0% 100.0% 100.0% 100.0% 100.0%
              100.0%  11.1%  11.1%  11.1%  11.1%  11.1%  11.1%  11.1%
Mean           1.33   1.00   1.00   1.00   2.00   2.00   2.00   1.00
S.D.           0.50   0.00   0.00   0.00   0.00   0.00   0.00   0.00
```

AGAIN, QUESTION CENTER;

```
                                  Page 41

                          this is the first xl this is the second
                          this is the third xl this is the fourth
                                   this is the final xl

one                   6      1      1      1      -      -      -      1
                   100.0%  16.7%  16.7%  16.7%                        16.7%
                    66.7% 100.0% 100.0% 100.0%                       100.0%
                    66.7%  11.1%  11.1%  11.1%                        11.1%

two                   3      -      -      -      1      1      1      -
                   100.0%                       33.3%  33.3%  33.3%
                    33.3%                      100.0% 100.0% 100.0%
                    33.3%                       11.1%  11.1%  11.1%


                                  Page 42

                          this is the first xl this is the second

Base                  9      1      1      1      1      1      1      1
                   100.0%  11.1%  11.1%  11.1%  11.1%  11.1%  11.1%  11.1%
                   100.0% 100.0% 100.0% 100.0% 100.0% 100.0% 100.0% 100.0%
                   100.0%  11.1%  11.1%  11.1%  11.1%  11.1%  11.1%  11.1%
Mean                1.33   1.00   1.00   1.00   2.00   2.00   2.00   1.00
S.D.                0.50   0.00   0.00   0.00   0.00   0.00   0.00   0.00
```

When the PAGE subcommand contains QUESTION as an argument and a number as the final argument, that number is used to determine how many of the lines of question text are to be used on continuation pages. NOTE: the number is *always last* regardless of the order of the other arguments.

```
PAGE QUESTION LABELS 1,
```

The two PAGE subcommands cannot be combined. One controls the SURVEY sections involved in a page change, the other controls the TITLES. Both can be used in a single SURVEY command.exi

In Figure 5.2 the extended label for variable Var1 contains 7 strings:

```
Var1  <<this is the first x>>    <<&&>>   <<l this is the second>>
      <<this is the third xl>>   <<& &>>  <<this is the fourth>>
      <<this is the final xl>>
```

The second string, "<<&&>>" and the fifth string "<<& &>>" have a special role. They are never included in the printed text but instead they serve to paste the two strings on either side together. When the two ampersands are separated by a space, the strings are joined with a space between them. When the two ampersands are not separated by a space, the strings are joined without a space. The seven strings are treated as 3 strings at print time. Figure 5.2 shows the subcommands and the resulting output when the QUESTION and PAGE subcommands are used.

5.4 The LABELS Section

The appearance of the variable labels is largely controlled by the setting of the cell width and the use of break characters in the labels themselves. However there are some minor enhancements that can be made. One is to omit the variable labels and use only the value labels. This can be very effective when the value labels are self explanatory. The subcommand to turn off the printing of the variable labels is:

```
NO VARIABLE.LABELS,
```

When the banners are nested, it is sometimes easier to see the domain of the outer nest levels when the value labels for that level are offset with special characters. The CHARACTER subcommand is used to provide these characters.

```
CHARACTER VALUE.LABELS '='  '<'  '>',
```

The first character in quotes is the fill character that replaces the blanks surrounding the label text. The second and third characters are optional and provide single characters to be used at the left and right edges of the label area.

```
CHAR VALUE '=',
```

can be used to supply just a fill character without the special ending characters. “CHAR” is an acceptable

for “CHARACTER” without significant loss of meaning. “VALUE” can be used instead of the more verbose “VALUE.LABELS”.

Figure 5.3 The LABELS Section

```
SURVEY Paceset,  LABELS 'Paceset.lab”;
  BAN Age WITHIN Sex,  STUB Own,

  LAYOUT LABELS,  MARGIN 8;
```

```
               =============== Age / Sex ===============

                        Male                Female

        Total  Under  30 to   Over  Under  30 to   Over
       Sample     30     50     50     30     50     50
```

```
AGAIN, CELL.WIDTH 12;
```

```
                   ============ Age / Sex ============

                                    Male

      Total Sample    Under 30    30 to 50     Over 50
```

```
AGAIN,  NO VARIABLE.LABELS, CELL.WIDTH 7;
```

```
                        Male                Female

        Total  Under  30 to   Over  Under  30 to   Over
       Sample     30     50     50     30     50     50
```

```
AGAIN,  CHARACTER VALUE.LABELS '='  '<'  '>' ;
```

```
                <====== Male ======> <===== Female =====>

          Total  Under  30 to   Over  Under  30 to   Over
         Sample     30     50     50     30     50     50
```

```
AGAIN,  LEFT JUSTIFY LABELS;
```

```
                <====== Male ======> <===== Female =====>

        Total    Under  30 to  Over   Under  30 to  Over
        Sample   30     50     50     30     50     50
```

```
AGAIN,  CENTER LABELS;
```

```
                <====== Male ======> <===== Female =====>

         Total    Under  30 to  Over   Under  30 to  Over
         Sample    30     50     50     30     50     50
```

The value labels are usually right justified to line up with the numbers in the body of the table. This often makes an attractive presentation. However, these labels can be centered or left justified in the area of the cell width. The subcommands:

```
CENTER LABELS,
LEFT JUSTIFY LABELS,
RIGHT JUSTIFY LABELS,
```

are available to control the placement. RIGHT JUSTIFY LABELS is the assumed setting. The justification affects only labels that need multiple lines to print. The first line of the label is always right justified over the data area. Subsequent lines are then placed and justified beneath them. Figure 5.3 shows the use of the CHARACTER subcommand and the justification subcommands, and the printout that results. For example the value label “Under 30” appears:

```
Under  (right)       Under   (left)    Under    (center)
   30                30                 30
```

Banner variables which have a single value and, therefore, a single column present a special problem. Often the most attractive and meaningful labelling leaves the area for the variable label blank and uses the variable name or, if available, the extended label in the area where the value is usually placed. This is what is done if there are no value labels. However, if you provide a value label for the single value, both the variable label and the value label will print, each in its appointed place. If there is a value label and the extended variable label is blank, only the value label is used.

5.5 The TOTALS.AREA

5.6 A Section

The TOTALS.AREA usually contains the total count and any requested percents. If the table is weighted, this is the total weighted count. This section usually has only a single item but can have two items.

Figure 5.4 The TOTALS.AREA

```
SURVEY Paceset, LABELS 'Paceset.lab', WEIGHT Weight;
   BANNER Age Income.Groups,  STUB Own,  LAYOUT LABELS TOTALS,
   CELL.WIDTH 9;


Weighted by: weight

                                                ==== Income of  ====
                              ======== Age ======= ==== respondent ====

                                                          $20,
                                                 Under 000 to
                        Total  Under  30 to   Over   $20,   $40, Over $
                       Sample     30     50     50    000    000 40,000

Weighted Total            82     25     26     29      2     51     20
                         100.0% 100.0% 100.0% 100.0% 100.0% 100.0% 100.0%
```

AGAIN, TOTALS.AREA CONTAINS UNWEIGHTED.N;

```
Weighted by: weight

                    =========== Age ========== == Income of respondent ==

                                                          $20,000
                Total                                     Under       to     Over
               Sample Under 30 30 to 50  Over 50  $20,000  $40,000  $40,000

Unweighted
Total             80       28       27       23        2       50       20
```

AGAIN, TOTALS.AREA CONTAINS GOOD.N;

```
Weighted by: weight

                    =========== Age ========== == Income of respondent ==

                                                          $20,000
                Total                                     Under       to     Over
               Sample Under 30 30 to 50  Over 50  $20,000  $40,000  $40,000

Weighted
Base              73       25       23       24        2       51       20
```

The subcommand is TOTALS.AREA CONTAINS followed by one or two of the 5 possible choices. The choices are:

```
TOTAL.N          or   WEIGHTED.N
BASE             or   GOOD.N         or   WEIGHTED.BASE
UNWEIGHTED.N
UNWEIGHTED.BASE
RESPONSES
```

One popular arrangement omits the totals area and requests that the totals (or weighted totals) be included in the summary section of the table. This summary section can then be appropriately placed using the LAYOUT subcommand.

"TOTAL.N" and "WEIGHTED.N" can be used interchangeably. If there is no WEIGHT variable, the table is considered to be weighted by 1. Similarly "BASE", "GOOD.N", and "WEIGHTED BASE" are interchangeable. Figure 5.4 contains three tables each with only 2 sections, the labels and the totals area. It shows the subcommands and the printout when the TOTALS.AREA is changed for a weighted table.

Figure 5.5 The BODY Section: Including Means

```
SURVEY Paceset, LABELS 'Paceset.lab', WEIGHT Weight;

   BANNER Age,  STUB Income.Groups,
   MEANS Income,  PLACES MEANS 0,
   INCLUDE MISSING MEANS VALUES,  CELL.WIDTH 9,
   BODY CONTAINS COUNTS MEANS PERCENTS;
```

Weighted by: weight

		Age		
	Total Sample	Under 30	30 to 50	Over 50
Weighted Total	82	25	26	29
	100.0%	100.0%	100.0%	100.0%
Income of respondent				
Under $20,000	2	-	2	-
	18400		18400	
	2.3%		7.3%	
$20,000 to $40,000	51	16	19	16
	30023	31359	28265	30736
	61.7%	63.9%	70.4%	56.0%
Over $40,000	20	9	3	7
	46859	48240	43467	46067
	24.8%	36.1%	10.9%	26.0%
Missing	9	-	3	5
	11.2%		11.3%	18.0%

Base	73	25	23	24
	88.8%	100.0%	88.7%	82.0%
Unweighted Total	80	28	27	23
Income	34412	37449	29330	35598
S.D.	9273	9197	8027	8419

Figure 5.6 The BODY Section: Weighted and Unweighted Counts and INDEX

```
AGAIN,  INDEX,  MAX.INDEX 300,
BODY CONTAINS  WEIGHTED.N  UNWEIGHTED.N  PERCENTS  INDEX;
```

Weighted by: weight

	Total Sample	Age: Under 30	Age: 30 to 50	Age: Over 50
Weighted Total	82	25	26	29
	100.0%	100.0%	100.0%	100.0%
Income of respondent				
Under $20,000	2	-	2	-
	2		2	
	2.3%		7.3%	
	100		300+	
$20,000 to $40,000	51	16	19	16
	50	18	19	13
	61.7%	63.9%	70.4%	56.0%
	100	104	114	91
Over $40,000	20	9	3	7
	20	10	3	6
	24.8%	36.1%	10.9%	26.0%
	100	146	44	105
Missing	9	-	3	5
	11.2%		11.3%	18.0%
Base	73	25	23	24
	88.8%	100.0%	88.7%	82.0%
Unweighted Total	80	28	27	23
Income	34412	37449	29330	35598
S.D.	9273	9197	8027	8419

5.7 The BODY Section

The body of the table usually contains the counts followed by any requested percents. There are a number of ways that the contents of the body can be controlled both for contents and for order.

1. NO PERCENTS leaves out the percents
2. NO COUNTS leaves out the counts
3. BODY CONTAINS can be used to change the order
4. BODY CONTAINS can be used to add additional information

The subcommand to change the order or add additional features is “BODY CONTAINS”. It is followed by a list of one or more items to be included. They appear in the table in the order they are requested. The items that can be included in the body are:

1. COUNTS, the weighted N when there is a weight variable and the unweighted N when there is no weight variable. COUNTS must be present if the cell chi square or cell expected value are requested.
2. PERCENTS, including row, column, and totals percents
3. MEANS, included only when the MEANS variable is not the STUB variable
4. MEDIANS, when the MEDIAN/MEANS variable is not the STUB variable. NOTE: the mean variable and the median variable must be the same variable.
5. SUMS
6. WEIGHTED.N, this is the same as COUNTS when the table is weighted
7. UNWEIGHTED.N, this is the same as COUNTS when the table is not weighted.
8. INDEX

If you wish to have the percents appear before the counts, use:

```
BODY CONTAINS PERCENTS COUNTS,
```

Figure 5.5 shows the subcommands and the printout when means of an extra variable are added to the body of the table. The counts in a table are usually printed as integers, that is, without any fractional part. In a weighted table you may wish to see the fractional part. This is done with the PLACES subcommand:

```
PLACES COUNTS 2,
```

causes all of the counts in the totals area, body, and summary section to print with 2 digits beyond the decimal point. (All numbers are appropriately rounded, not truncated, when places are specified.) If PLACES is not used in a weighted table, the row and column totals may appear to differ from the sum of the cells.

Figure 5.6 shows the subcommands and the printout when both the weighed and unweighted counts are included in the body. It also includes an INDEX statistic. The INDEX provides a measure of how well the distribution of a given column matches the distribution across all the columns. It is calculated by dividing the column percent of each cell by the overall column percent for that row. The result is multiplied by 100 and printed without a fractional part.

In Figure 5.6, only 2.3 percent of the respondents are in the lowest income category. However, 7.3 percent of those in the middle age group are in that category. The division of 7.3 by 2.3 gives a result somewhat over 3. This is multiplied by 100 and if MAX.INDEX has been specified, it is compared with the MAX.INDEX setting. Any number greater than the MAX.INDEX value is printed as that value with the addition of a plus (+) sign. In Figure 5.6 this appears as “300+”.

When there is an empty cell in the body, the first line in the cell is usually represented as a dash rather than the number zero. Subsequent lines in an empty cell are usually left blank.

This can be changed to any other character. Often the replacements are either a blank or the number “0”. Character replacement is done with the CHARACTER subcommand. This is discussed in greater detail in chapter 6, “SURVEY: Enhancing Table Appearance”.

```
CHARACTER ZERO ' '
```

```
      CHAR      ZERO '0',
      CHAR      ZERO '0' '0',
```

If two characters are supplied, the first one is used for the first line in the empty cell and the second is used for subsequent lines in the empty cell.

There are 2 more lines that can be included in the body section. The expected value is the number we would expect to be the cell count if all we know about the table is the row and column totals. The cell chi square is calculated using the observed value and the expected value and is the amount that the cell contributes to the chi-square statistic for the table. The subcommands are:

```
      CELL.CHI,    EXPECTED.VALUE.
```

These statistics always follow the COUNT in the body of the table and do not print if the COUNT is omitted. Neither the cell chi-square nor the expected value are calculated for a body row that is the result of a dynamic combine. COMBINE is described in detail in the chapter “SURVEY: All About the Rows”.

Figure 5.7 The SUBTOTALS Section

```
   SURVEY PACESET,  LABELS 'Paceset.lab';
      BAN Age Sex, STUB Num.TV,

      DEFINE '1 TO 3' Num.TV 1 TO 3,
      DEFINE '4 or more' Num.TV 4 TO 6,
      LAYOUT QUESTION LABELS SUBTOTALS SUMMARY;


 Number of television sets in your home:

                      ======== Age ======= ==== Sex ====

                 Total  Under  30 to   Over
                Sample     30     50     50   Male Female

 1 TO 3            46     26     13      6     33     13
 ------          57.5%  92.9%  48.1%  26.1%  84.6%  32.5%

 4 or more         34      2     14     17      6     27
 ---------       42.5%   7.1%  51.9%  73.9%  15.4%  67.5%

 Base              80     28     27     23     39     40
                 100.0% 100.0% 100.0% 100.0% 100.0% 100.0%
 Mean             3.09   2.32   3.22   3.83   2.54   3.60
 S.D.             1.13   0.90   1.01   0.98   1.00   1.01
```

5.8 SUBTOTALS

SUBTOTALS are never printed unless DEFINE or DEFINE.NET is used to specify how the subtotals or nets are to be calculated. If either DEFINE or DEFINE.NET is used and there is no LAYOUT or SUBTOTALS subcommand, the subtotals are placed immediately following the body of the table and before the missing values. This is the SUBTOTALS section. The LAYOUT subcommand can be used to move the section. Figure 5.7 contains the command, subcommand and output when the SUBTOTALS section is positioned using the LAYOUT subcommand.

The placement of the subtotals in relationship to the rows in the body is controlled by the LAYOUT and the SUBTOTOTALS command. The subtotal rows are usually flush left in the margin area unless the DEFINE subcommand indicates a subtotal level. A DEFINE with no level indicator is considered level 1. A DEFINE with a level 2 indicator is placed 2 spaces to the right. Each successive indent level is placed 2 spaces further to the right.

```
     DEFINE    '1 TG 4 TV sets' Num.TV 1 to 4,
     DEFINE  2 '1 to 3 TV sets' Num.TV 1 to 3,
     DEFINE  3 '1 or 2 TV sets' Num.TV 1 2,
```

Figure 5.8 Indent Levels in Subtotals

```
 SURVEY PACESET,  LABELS 'Paceset.lab';
     BAN Age Sex, GR.STUB Num.TV Income.groups,
     DEFINE    "Totals TV's" Num.TV  1 TO 6,
     DEFINE 2  '1 TO 3'      Num.TV  1 TO 3,
     DEFINE 2  '4 to 6'      Num.TV  4 TO 6,

     DEFINE    'Totals'        Income.groups  1 to 3,
     DEFINE 2  'Low Income'    Income.groups  1 2,
     DEFINE 2  'Higher Income' Income.groups  3,

     LAYOUT LABELS TOTAL QUESTION SUBTOTALS;
 $
```

		======== Age =======			==== Sex ====	
	Total Sample	Under 30	30 to 50	Over 50	Male	Female
Total Sample	80	28	27	23	39	40
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Number of television sets in your home:						
Totals TV's	80	28	27	23	39	40
----------	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
1 TO 3	46	26	13	6	33	13
------	57.5%	92.9%	48.1%	26.1%	84.6%	32.5%
4 to 6	34	2	14	17	6	27
------	42.5%	7.1%	51.9%	73.9%	15.4%	67.5%
Income of respondent						
Totals	72	28	24	19	31	40
------	90.0%	100.0%	88.9%	82.6%	79.5%	100.0%
Low Income	52	18	21	13	21	30
----------	65.0%	64.3%	77.8%	56.5%	53.8%	75.0%
Higher Income	20	10	3	6	10	10
------------	25.0%	35.7%	11.1%	26.1%	25.6%	25.0%

Up to 9 indent levels can be used. They need not be successive. You can use levels 1, 3, and 5, if you wish to get a greater indentation. If you are using indent levels, you should also supply a margin that is wide enough to account for the extra spaces. Labels that are too long are wrapped on succeeding lines, but if the margin is too narrow, their appearance may not be satisfactory.

Figure 5.8 contains the SURVEY command, subcommands and the resulting table when a single indent level is used. It also illustrates the use of SUBTOTALS to replace both the body and the totals sections. GR.STUBS (GROUPS.STUBS) is used to display two tables as one.

The items that can appear in the subtotals section are the same as the items that can appear in the body except for the cell chi-square and the expected value which are not appropriate for subtotals. These items are:

```
COUNTS          PERCENTS        MEANS         SUMS
UNWEIGHTED.N    WEIGHTED.N    INDEX
```

Figure 5.9 Selecting the Contents of the Subtotals

```
SURVEY Paceset,  LABELS 'Paceset.lab';
    BANNER Age Sex,  STUB Num.TV;   MARGIN 24,
    NO MISSING,  NO SUMMARY,
    DEFINE “Income: Owns 1 or 2 TV's” Num.TV 1 2,
    DEFINE “Income: Owns 3+ TV's” Num.TV 3 to 5,
    SUBTOTALS BEFORE LOW VALUES,
,   SUBTOTALS CONTAIN MEANS,
    INDENT 10,
    MEANS Income,  PLACES MEANS 0,
    BODY CONTAINS COUNTS;
```

		======== Age =======			==== Sex ====	
	Total Sample	Under 30	30 to 50	Over 50	Male	Female
Total Sample	72	28	24	19	31	40
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Number of television sets in your home:						
Income: Owns 1 or 2 TV's	35853	39490	28771	27900	37687	29433
1	6	4	1	1	4	2
2	21	14	6	1	17	4
Income: Owns 3+ TV's	33525	33605	29533	36484	34810	33198
3	12	8	3	1	5	7
4	26	1	13	11	3	22
5 or more	7	1	1	5	2	5

COUNTS is the same as either the WEIGHTED.N or the UNWEIGHTED.N depending on whether the table is weighted or unweighted.

Usually a row in the subtotals section has the same items in the same order as a row in the body. If you wish to have the subtotals contain different information, use the SUBTOTALS CONTAINS subcommand.

```
BODY CONTAINS COUNTS PERCENTS,
SUBTOTALS CONTAIN SUMS MEANS,
```

Figure 5.9 shows the subcommands and the table that is produced when the body and the subtotals have different contents. In Figure 5.9 the SUBTOTALS are not in their own section but are interleaved in the BODY section.

Figure 5.10 Controlling Missing Values

```
SURVEY Paceset, LABELS 'Paceset.lab';
       BANNER AGE OWN,  STUB Income.groups,
       LAYOUT MISSING,
```

```
Missing                  8      -      3      4      1      7
                      10.0%         11.1%  17.4%   9.1%  10.1%
```

```
     AGAIN,  MISSING ALL;
```

```
Missing 1                -      -      -      -      -      -
Missing 2                5      -      2      2      1      4
                      6.2%          7.4%   8.7%   9.1%   5.8%
Missing 3                3      -      1      2      -      3
                      3.8%          3.7%   8.7%          4.3%
```

```
     AGAIN,  MISSING OBSERVED;
```

```
Missing 2                5      -      2      2      1      4
                      6.2%          7.4%   8.7%   9.1%   5.8%
Missing 3                3      -      1      2      -      3
                      3.8%          3.7%   8.7%          4.3%
```

```
     AGAIN,  MISSING M1 M3  COMBINED;
```

```
Missing                  3      -      1      2      -      3
                      4.0%          4.0%   9.5%          4.6%
```

5.9 The MISSING Section

Counts of the three types of missing data values can be displayed all together or broken out into separate counts for each type. In addition, individual types of missing values can be excluded from the table and the total count. This control of how missing data values are displayed and tallied, along with the ability to provide custom labels

for missing values, affords much latitude in handling responses such as “don’t know,” “refused” and “not applicable.”

Missing values of the stub variable are normally shown, tallied together as “Missing,” in the MISSING section of a table. This section usually appears unless “NO MISSING” is specified or there is a LAYOUT subcommand which does not include the keyword MISSING. This behavior can be controlled by the MISSING subcommand. The missing values can be reported separately. They can be printed conditionally. Missing can be used to remove certain cases from the table completely.

1. MISSING COMBINED, is the default. All three types of missing are combined into 1 row.
2. MISSING OBSERVED, prints only if there are missing values and prints 1 row for each type of missing that occurs for the stub variable.
3. MISSING ALL, print a row for each of the 3 types of missing.

It is possible to specify the exact types of missing data that should be included in each grouping of missing values and in the total count, as well as the manner in which they are to be displayed. The additional arguments M1, M2 and M3 follow the MISSING subcommand and must precede the arguments COMBINED, ALL and OBSERVED.

Figure 5.11 Identifying Missing on the Mean or Median Variable

```
SURVEY Paceset,
          LABELS 'Paceset.lab';
     BAN Age Own, STUB Sex,
     MEANS Income,
     INCLUDE MISSING MEANS VALUES,
     SUMMARY CONTAINS MEANS,
     MISSING COMBINED M4 SEPARATE;
$
```

		Age			Respondent ownership	
	Total Sample	Under 30	30 to 50	Over 50	no	yes
Total	80	28	27	23	11	69
Sample	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Sex						
Male	31	20	8	3	4	27
	38.8%	71.4%	29.6%	13.0%	36.4%	39.1%
Female	40	8	16	15	6	34
	50.0%	28.6%	59.3%	65.2%	54.5%	49.3%
Missing Income	8	-	3	4	1	7
	10.0%		11.1%	17.4%	9.1%	10.1%
Missing	1	-	-	1	-	1
	1.2%			4.3%		1.4%
Mean Income	34435	37388	29311	35790	26785	35689

This subcommand:

```
MISSING  M2  M3  COMBINED,
```

includes only missing value types 2 and 3 in the total count and prints the total of both M2 and M3 in the single category "Missing" in the table. This may be appropriate when missing one represents "Not Applicable," and thus the question was not asked of certain respondents. Missing two and missing three are tallied together — they could be labeled "Don't Know/Refused" or some other suitable label.

This:

```
MISSING  M2  M3  ALL,
```

would similarly exclude missing one, but would display missing two and missing three as separate counts. They would be labeled "Missing 2" and "Missing 3" unless alternate labels are supplied. To exclude missing type 1 and display missing 2 and missing 3 only if they are actually present in the data, use:

```
MISSING  M2  M3  OBSERVED,
```

Figure 5.10 illustrates the MISSING subcommand and the resulting missing sections. The final subcommand:

```
MISSING M1 M3 COMBINE,
```

causes the 5 cases with MISSING type 2 on variable Income.groups to be omitted from the table.

When M1, M2 and/or M3 do not follow the MISSING subcommand, it is assumed that all three types of missing values are to be included in the table and displayed as specified. When specific types of missing are selected, any case with a selected missing type is included in the table and any case with a type of missing that is not selected is dropped from consideration as the data are read. This means that they do not appear even in the total count. It has much the same effect as using the P-STAT programming language to delete the case.

```
SURVEY Paceset ( IF Income.groups MISSING2, DELETE ),
```

The use of the MISSING subcommand provides more flexibility if you are doing a series of tables. A given case is omitted on a table by table basis rather than dropped from the file for the entire SURVEY command.

When means on an extra variable are requested, a 4th type of missing is recorded. This is the count of cases with good values on the stub variable and missing values on the means variable. These cases are usually combined with MISSING 1 unless separately requested.

The MISSING subcommand is used to request separate identification of cases missing on the extra means or median variable. If the final two arguments are "M4 SEPARATE", these missing counts are reported separately. This can be used with any of the other MISSING options.

```
MISSING  M2  M3  ALL M4 SEPARATE,
```

This subcommand requests that MISSING 2 and MISSING 3 be reported separately even if they have no cases. Cases with a value of MISSING 1 are not included in the table in any way. Cases with a value of missing on the mean/median variable are reported separately. NOTE: unless INCLUDE MISSING MEANS VALUES is used, any case that is MISSING 1 on the stub variable and missing on the means variable is dropped from the table. MISSING 1 takes precedence when there is a conflict.

5.10 The SUMMARY Section

The assumed contents of the SUMMARY section changes with the type of table. In an ordinary unweighted table, the counts, the mean and the standard deviation are the usual contents. In a weighted table, the unweighted count is added to this list. When the table is a cross comparison table, the summary usually contains just the total number of responses. A number of other statistics can be added to the summary section:

Figure 5.12 The SUMMARY Section

```
  SURVEY Paceset,  LABELS 'Paceset.lab',  WEIGHT Weight;
    LAYOUT LABELS SUMMARY,
    BANNER Age Own,  STUB Income.groups;

Weighted by: weight

                                                  Respondent
                              ======== Age =======   ownership

                    Total  Under  30 to   Over
                   Sample     30     50     50     no    yes

Base                   73     25     23     24     10     63
                     88.8% 100.0%  88.7%  82.0%  88.7%  88.8%
Unweighted Total       80     28     27     23     11     69
Mean                 2.25   2.36   2.04   2.32   1.91   2.31
S.D.                 0.50   0.49   0.46   0.48   0.31   0.50


   AGAIN, MARGIN 16,  CELL.WIDTH 8,  GAP 2, NO PERCENTS,

   SUMMARY CONTAINS TOTAL.N  UNWEIGHTED.N    BASE  UNWEIGHTED.BASE
                    SUM   MEAN   S.D.  VARIANCE    STANDARD.ERROR,
                    QUARTILES    MODE  LOW   HIGH   RANGE;

Weighted by: weight

                                                  == Respondent =
                              ========= Age =========   == ownership  =

                    Total     Under   30 to
                   Sample        30      50 Over 50        no     yes

Total Sample           82        25      26      29        11      71
Unweighted Total       80        28      27      23        11      69
Base                   73        25      23      24        10      63
Unweighted Base        72        28      24      19        10      62
Total                 164        59      48      55        19     145
Mean                 2.25      2.36    2.04    2.32      1.91    2.31
S.D.                 0.50      0.49    0.46    0.48      0.31    0.50
Variance             0.25      0.24    0.21    0.23      0.10    0.25
Standard error       0.06      0.10    0.10    0.10      0.10    0.06
Q1                   2.00      2.00    2.00    2.00      2.00    2.00
Q2                   2.00      2.00    2.00    2.00      2.00    2.00
Q3                   3.00      3.00    2.00    3.00      2.00    3.00
Mode                    2         2       2       2         2       2
Minimum                 1         2       1       2         1       1
Maximum                 3         3       3       3         2       3
Range                   2         1       2       1         1       2
```

The possible contents of the SUMMARY section are:

1.	TOTAL.N	The total count (weighted and/or unweighted)
2.	BASE	The total non-missing count (weighted and/or unweighted)
3.	MEANS	The means of the stub variable or a specified other variable
4.	SUMS	The totals of the means variable
5.	STANDARD.DEV	The standard deviation of the means variable
6.	VARIANCE	The variance of the means variable
7.	STANDARD.ERROR	The standard error of the means variable
8.	MODE	The modal value of the stub variable
9.	LOW	The lowest value of the stub variable
10.	HIGH	The highest value of the stub variable
11.	RANGE	The range of the stub variable
12.	MEDIAN	The median of the stub (or means) variable
13.	QUARTILES	The three quartiles (includes median) of the stub (or means) variable

Figure 5.13 SUMMARY Section: Means on Another Variable

```
    AGAIN,  MEANS INCOME,  INCLUDE MISSING MEANS VALUES,
           CELL.WIDTH 10,  GAP 0,  PLACES MEANS 0,  PLACES MEDIANS 0;

 Weighted by: weight

                                                          ==== Respondent ===
                              ============ Age ============ ==== ownership  ===

                    Total
                   Sample  Under 30  30 to 50   Over 50        no       yes

 Total Sample          82        25        26        29        11        71
 Unweighted Total      80        28        27        23        11        69
 Base                  73        25        23        24        10        63
 Unweighted Base       72        28        24        19        10        62
 Total Income     2505546    928739    686455    840942    271649   2233897
 Mean               34412     37449     29330     35598     26777     35648
 S.D.                9273      9197      8027      8419      5303      9210
 Variance        85988925  84578509  64433966  70887812  28121707  84821028
 Standard error      1087      1847      1659      1732      1665      1163
 First quartile     32520     30850     24200     29000     25700     27700
 Median             32520     35100     26000     32520     26700     34400
 Third quartile     41000     45800     37000     42000     27600     42500
 Mode                   2         2         2         2         2         2
 Minimum                1         2         1         2         1         1
 Maximum                3         3         3         3         2         3
 Range                  2         1         2         1         1         2
```

If you use SUMMARY CONTAINS but have a layout with no provision for a summary section, the summary statistics will not print. The individual lines in the summary can be removed by preceeding the appropriate key-word with “NO”. For example, NO BASE, when used with the default settings, produces a summary section with the mean and the standard deviation. NO SUMMARY can be used to turn off the summary section. A subsequent use of SUMMARY without arguments will reinstate the default values.

Usually the stub variable, the means variable and the median variable are the same and the summary statistics are based on the same values. However, if the means/median variable is other than the stub variable, the means, median, sums, standard.deviation, variance, and standard error are calculated using the value for the means variables while the other statistics are calculated using the stub values. This can be clearly seen in Figure 5.13.

There are several methods for calculating the median. These methods and calculating the median of another variable are discussed in the chapter “SURVEY: The Statistics”.

5.11 MORE ABOUT LAYOUT CONTROL

It is possible to turn off sections of the layout by using NO followed by the name of the section and to turn the sections back on simply by using the section name. This means that the desired basic layout provided at the start of a job can often be left unchanged for the entire job with sections removed or added back as needed.

```
LAYOUT QUESTION  LABELS  BODY  MISSING  SUBTOTALS  TOTALS.AREA
       MISSING   SUMMARY,
 NO SUMMARY,  NO MISSING;

BANNER Age Gender Income.Group,
       STUBS Q1 to Q30 ;

STUBS Q31a to Q33a,
       SUMMARY ON, NO BODY;
```

The keyword ON following the section name is optional but can be included for clarity. The basic layout remains the same, as different sections are selected. In the example above, the tables produced for the stubs Q1 to Q30 are produced without the summary or missing sections. The tables produced for Q31a to Q33a have a summary section but no body or missing section. However, all of the tables have the included sections arranged according to the initial layout.

The construction that permits control over a given section as in:

```
SUMMARY CONTAINS MEAN,
```

defines the contents of the summary section. CONTAINS does not turn a section on. However, if you use:

```
SUMMARY INCLUDES MEAN,
```

the contents of the section are defined and the section is turned on. The definitions for the section remain in force until a subsequent CONTAINS or INCLUDES is processed. As always, a section must have a place in the layout. If it is defined and turned on but has no location in the layout, it is not printed.

5.12 Additional Formats for Value Labels

Usually the value labels have either the value or, if there is a labels file, the text that is associated with the value. If you want to see both the value and the label text, you can use either the SVAR.LABELS or BVAR.LABELS subcommands. The following are examples of possible SVAR.LABELS and BVAR.LABELS subcommands.

```
SVAR.LABELS VALUE TEXT '()',
BVAR.LABELS TEXT VALUE ':',
SVAR.LABELS VALUES TEXT ' - ',
BVAR.LABELS TEXT VALUE '[]',
```

These four subcommands pertain only to value labels for stub variables. BVAR.LABELS for banner variables can also be used. However, there is usually less room to format banner variable value labels and the addition of the value information may cause the text to be truncated.

The order of the values and text depends on the order that the keywords appear in the subcommand. The final argument is a character string that serves as a delimiter. If this string is a parentheses, bracket or curly brace pair,

it is assumed that the value is to be enclosed in the characters. If the value is any other string, that string is used between the value and text. Figure 5.14 contains examples of the output when SVAR.LABELS is used.

The string can be 1 to 4 non-blank characters. There will always be a single blank between the text and value or value and text.

Figure 5.14 Enhancing Labels Text With the Value

```
      -----Value then text with parentheses------

      svar.label value text '()'

      (1) Male          39     20     11      7
                       48.8%  71.4%  40.7%  30.4%


      ------Text then value with parentheses------

      svar.labels text value '()';

      Male (1)          39     20     11      7
                       48.8%  71.4%  40.7%  30.4%


      ------Value then text with string------

      svar.labels value text ' - ';

      1 - Male          39     20     11      7
                       48.8%  71.4%  40.7%  30.4%


      ------Text then value with single character------

      svar.labels text value ':';

      Male: 1           39     20     11      7
                       48.8%  71.4%  40.7%  30.4%


      ------Text then value with a string------

      svar.labels text value ’ val’

      Male val 1         39     20     11      7
                        48.8%  71.4%  40.7%  30.4%
```

SUMMARY

SURVEY

```
TITLE  B2 'Page .PAGE.' $

SURVEY Paceset,  LABELS 'Paceset.lab',
   TITLES,  PAGE.NUMBER 22,  LINES 24 ;

   BANNER  Age Own,  STUB Num.TV,
   BODY CONTAINS PERCENTS,
   SUMMMARY CONTAINS MEAN MEDIAN MODE RANGE,
   ROW COLUMN PERCENTS,  BREAK;
$
```

This chapter illustrates a variety of ways to enhance the appearance of the table by controlling the behavior of page changes and by selecting the contents of each section of the table.

Optional Identifiers:

PAGE.NUMBER nn

supplies a starting page number that will be used when the system variable .PAGE. is present in the titles.

```
SURVEY.Paceset,  LABELS 'Paceset.lab',  PAGE.NUMBER 43;
```

RUN.PAGE.NUMBER nn

supplies a starting value for the .RPAGE. system variable which is incremented whenever .RPAGE. is used in a title. .RPAGE.is used instead of .PAGE. when several SURVEY commands are executed in the same run and the page numbering should be continued across commands.

Optional Subcommands:

BREAK

causes a linefeed when the lines setting is exceeded and the printout continues on the next page.

CELL.CHI

computes and prints the contribution that each cell makes to the overall chi-square. The cell chi-square prints following the COUNTS and the expected values, in the BODY section.

EXPECTED.VALUE

computes the value that one would expect as the cell count if one knew only the marginals. This number is used in computing chi-square. It is also helpful in deciding if the contribution of the cell is important. If the expected value is less than 5, any significance it seems to have should be discounted.

INDEX

computes a statistic which is a measure of how well a column compares to the distribution of the variable as indicated by its row total.

MAX.INDEX nn

specifies the number beyond which the index statistic should print the details. If MAX.INDEX is set to 300, an index of 310 prints as 300+.

PAGE arg arg nn

If PAGE is used by itself without any arguments, a page change occurs when the lines setting is exceeded and the TITLES, QUESTION, LABELS, and TOTALS.AREA sections are repeated on the following page. If you wish to control which sections are to appear on the continuation pages and in which order, PAGE is followed by one or more of:

```
QUESTION  LABELS  TOTALS.AREA
```

If the final argument is a number, that is used to determine how many lines of question (extended label) text should be used on the continuation pages.

PAGE TITLES nn

specifies he number of top titles that should print on each continuation page. Both uses of the PAGE subcommand can appear within the table definitions.

PAGE TITLES ONLY nn

specifies that the titles are the only element to print on a continuation page. The optional number specifies how many of the titles are to be printed on a continuation page.

PAGE TABLE.TITLES nn

specifies the number of table titles to be printed on continuation pages. Table titles always print immediately after any top titles.

USE TITLES FIRST/CONTINUED nn nn nn nn

provides additional control over the placement of titles on FIRST or CONTINUED pages. The numbers are in pairs and refer to the title line. The first pair controls top titles. The second (optional) pair controls bottom titles.

```
 USE TITLES FIRST 1 3,
 USE TITLES CONTINUED
3 5 3 3
```

Optional Subcommands: The LAYOUT

BODY CONTAINS / INCLUDES arg arg arg

The body of the table usually contains the counts and any percents that may be requested. BODY CONTAINS is used to specify the order and the exact contents. The arguments include:

```
COUNTS   PERCENTS  MEANS  MEDIAN  SUMS  WEIGHTED.N  INDEX
```

CONTAINS defines the sections of the body but does not turn that section on. INCLUDES can be used to both define and turn the body section on. NO BODY is used to turn that section off.

NO VARIABLE LABELS

This is a control for the LABELS section. If used, the labels section contains only the value labels.

CENTER LABELS

causes the value labels to be centered within the cell width. RIGHT JUSTIFY LABELS is the assumed setting. LEFT JUSTIFY LABELS can also be used.

MISSING arg arg arg

The MISSING subcommand controls the way that missing values are selected and displayed. If MISSING is immediately followed by one or more of M1, M2, and M3, selection occurs and only those that are in the list are included in the table. The assumption is that the missing values that are included are to

be aggregated and presented as a single line. This is "COMBINED", the default. If "OBSERVED" is the final argument, only the types of missing that are observed in the data are reported. If "ALL" is the final argument, all included types of missing are individually reported even if there are no data with those missing values.

When the final two arguments are "M4 SEPARATE" any cases with missing on the mean/median variable are reported separately.

QUESTION LEFT / CENTER

The positioning of the stub extended labels is assumed to be LEFT unless QUESTION CENTER is used to center it on the page. When there are multiple extended labels for the row (stub) variable, they are usually all printed on each page. The PAGE subcommand is used to change the number that should print on continuation pages.

QUESTION QUESTION IN LABELS.AREA

This combines the question area and the labels area. The question text is printed in the margin space on the left of the variable labels and the value labels. This is not available for transposed files.

SUBTOTALS CONTAINS / INCLUDES arg arg

Usually the subtotals have the same contents as the cells in the body of the table. However, SUBTOTALS CONTAIN can be used to specify just what the SUBTOTALS should contain and in what order. INCLUDES is like CONTAINS in defining the contents. It has the added attribute of turning the section onn if a NO SUBTOTALS has turned it off. Possible selections are:

```
PERCENTS      MEANS           SUMS        INDEX
WEIGHTED.N    UNWEIGHTED.N
COUNTS
```

COUNTS is the same as WEIGHTED.N when the table is weighted or as the UNWEIGHTED.N when the table is not weighted.

The median of the subtotals is produced only if medians for the body are requested and SUBTOTALS CONTAINS is not used. The subtotal median is estimated from the medians of the rows in the table that are combined to produce the subtotal. Medians are not available when the subtotal section is a net rather than a true subtotal.

SUMMARY CONTAINS / INCLUDES arg arg arg

The normal contents of the SUMMARY section varies with the type of table. It can be controlled both for contents and order by the use of SUMMARY CONTAINS followed by 1 or more of the following keywords:

```
TOTAL.N          BASE   MEANS   SUMS    STANDARD.DEV       VARIANCE
STANDARD.ERROR   MODE   LOW     HIGH    RANGE     MEDIAN   QUARTILES
```

INCLUDES is like CONTAINS in defining the contents. It has the added attribute of turning the section onn if a NO SUMMARY has turned it off.

TOTALS.AREA CONTAINS / INCLUDES arg

The TOTALS.AREA usually contains the TOTAL.N. This is the weighted count in a weighted table. The choice of what it is to contain can be controlled by selecting one or two of:

```
TOTAL.N   BASE   UNWEIGHTED.N   UNWEIGHTED.BASE   RESPONSES
```

INCLUDES is like CONTAINS in defining the contents. It has the added attribute of turning the section onn if a NO TOTALS.AREA has turned it off.

Optional Subcommands: The LAYOUT

BVAR.LABELS TEXT/VALUE VALUE/TEXT cs

is used when you wish the banner value labels to include both the value and the text that is found in the labels file. The final argument is an optional character string of 1-4 characters which serve as a delimiter between the value and text. If the string is one of '()', '[]' or '{}', it is assumed that the value is to appear within the characters selected. For example: (22) .8

SVAR.LABELS TEXT/VALUE VALUE/TEXT cs

is used when you wish the stub value labels to include both the value and the text that is found in the labels file. The final argument is an optional character string of 1-4 characters which serve as a delimiter between the value and text. If the string is one of '()', '[]' or '{}', it is assumed that the value is to appear within the characters selected. For example: (22) .8

6
SURVEY: Enhancing Table Appearance

This chapter covers the various subcommands that permit fine-tuning of the appearance and contents of the table. There are controls for pagination. There are special features for the content and appearance of the labels and titles. Large numbers can be formatted with commas for easier reading. The SPREAD and COMPACT subcommands can be used to control the dynamic formatting of column width when means on an extra variable are used.

Tables can be produced with a special character such as a tab between each column to make it easier to move a table from P-STAT to a postprocessing program. Tables can be written out in a format that is easy to import into a spreadsheet. Blank lines within and between sections can be controlled with the SKIP.RULES subcommand. Tables with no good information can be omitted with the EMPTY.TABLES subcommand. Use of PostScript to produce camera ready table is covered in the next chapter.

6.1 SURVEY.LABELS: CUSTOM LABELING

Custom text and labels can be supplied to replace the default character strings that normally appear on tables produced by the SURVEY command. The default strings that can be replaced include “Total Sample”, “Unweighted Total”, “Mean”, “F value” and other such labels. The replacement labels can be text or blank strings. The text can be in English or any other language, and it can even include symbols or other unusual characters (what actually prints is dependent upon the input keyboard and the printer).

The replacement labels are supplied as labels for a dummy variable named “SURVEY.LABELS”. This variable must *NOT* exist in the P-STAT system file. This dummy variable and its labels can be part of an existing labels file or in a new labels file, either with other labels or by itself. The values for SURVEY.LABELS can be any of the following parenthesized values. The value in parentheses is what links the new label with the default string.

Replacement labels can be supplied for any of the following default strings:

```
(1)   Total Sample      (2)   Base              (3)   Responses
(4)   Unweighted Total  (5)   Unweighted Base

(11)  Total             (12)  Mean              (13)  S.D.
(14)  Variance          (15)  Standard error    (16)  Median
(17)  Mode              (18)  Maximum           (19)  Minimum
(20)  Range             (21)  First quartile    (22)  Third quartile
(23)  Percentile

(41)  F value           (42)  t value           (43)  Probability
(44)  Chi Square        (45)  DF for Chi        (46)  Fisher 2-tail
(47)  Yates chi-sq....  (48)  Some cells ....   (49)  Cramer’s V

(101) Total Sample      (102) Mean (banner)     (103) Good.N (banner)
(104  Median (banner)   (105) Totals (banner)   (106) Percentiles
(110) within
```

```
(120) Counts            (121) Row %            (122) Col %
(123) Tot %
(202) Weighted by:      (203) Page             (204) Table
(205) Contents

(210) BY                (211) Missing          (212) Overall Totals
(213) of list           (214) Missing (mean/median variable)

(301) Label for significance tests of means
(302) Label for test of proportions
(303) Label for test of independence
(304) Label when combined values are used in the tests
```

the next five values are used only when REPORT ALL RESULTS is used and none of the tests have been significant.

```
(310) used with TEST ALL RESULTS and CHI
(311) used with TEST ALL RESULTS and F
(312) used with TEST ALL RESULTS and TEST PROP
(313) used with TEST ALL RESULTS and TEST INDEPENDENCE
(314) used with TEST ALL RESULTS and TEST MEANS
```

General settings for missing values can be provided in the survey labels file:

```
(m1)  missing 1         (m2)  missing 2         (m3)  missing 3
```

The first three groups of labels 1-5, 11-22, and 41-49 (values that are between 1 and 99) are for labels that are associated with the rows of a table. The final group, comprising 'm1','m2' and 'm3', also has labels for rows of the table. The labels between 100 and 199 are associated with columns of the table. Labels between 200 and 299 are generally associated with text that appears in the headings of the table. Labels above 300 are associated with the significance testing that is described in the next chapter.

The first five values in SURVEY.LABELS are used for labels that normally appear in either the TOTALS or SUMMARY sections of tables. Note that “Total Sample” is equivalent to “Weighted Total” in weighted tables and “Totals” in multiple response tables. “Base” is equivalent to “Weighted Base” in weighted tables. The second group of values (11 through 20) are labels that normally appear in the SUMMARY section. The third group (41 through 49) appear after the SUMMARY when various statistical tests are requested.

The fourth group (101 - 122) is for labels which head a column in the table. The code of 101 is associated with the label for the row totals. A code of 102 as associated with means of a banner variable. A code of 103 is associated with the good count for a banner variable. A code of 104 is associated with the median of a banner variable, a code of 105 is associated with the totals of a banner variable that is declared as a quantity variable, and a code of 110 is associated with the word “WITHIN” which is used in labelling some nested tables.

The four codes 120, 121, 122, and 123 are only used when the banner variable request individual percents.

```
BANNER AGE ( PCT ), ROW COLUMN PERCENTS,
```

In this example, each value for banner variable Age has 3 columns, the count, the row percents and the column percents. The label for the counts in this situation is not the label associated with the actual value but the text “Counts”. An entry of 120 in SURVEY.LABELS provides an alternate label for this column. Similarly 121 is used to provide alternate text to replace “Row %” and “122 provides alternate text for “Col %”. The code of 123 is associated with “Tot %”.

The next group (202 through 205) is for the weighted table heading and the labels in the table of contents. The next group is for labels that appear in BY tables (210 through 212) and in a multiple response table with a list of means (213). Both “BY” and the table of contents are described in the chapter: “SURVEY: Other Features”. The values in the 300’s are used for the footnotes that appear when tests of significance are used. These tests are described in the chapters “SURVEY: The Statistics” and “SURVEY: More Statistics”..

Figure 6.1 SURVEY.LABELS: Custom Labelling

```
File Survey.lab

Weight ( xl ' ' ) /
SURVEY.LABELS
(1) Population:Weighted (2) Good N:Weighted
(4) Unweighted total    (5) Unweighted Good N
(11) Sums               (12) Average
(13) Standard deviation (14) S.D. squared      (16) Q2
(18) Low value          (19) High value        (20) High - Low
(21) Q1                 (22) Q3                (101) Respon*dents
(202) ' '               (211) No Reply to the question /

 SURVEY Pacesetw,  LABELS 'Paceset.lab' 'survey.lab',  WEIGHT Weight;
  LAYOUT LABELS SUMMARY,  BANNER Age Own,  STUB Income.groups,
  MARGIN 16,  CELL.WIDTH 8,  GAP 2, NO PERCENTS,

 SUMMARY CONTAINS TOTAL.N  UNWEIGHTED.N    BASE  UNWEIGHTED.BASE
                 SUM   MEAN   S.D.  VARIANCE    STANDARD.ERROR
                 MODE  QUARTILES    LOW   HIGH   RANGE;
 $
```

		========= Age =========			== Respondent = == ownership =	
	Respon dents	Under 30	30 to 50	Over 50	no	yes
Population: Weighted	82	25	26	29	11	71
Unweighted total	80	28	27	23	11	69
Good N:Weighted	73	25	23	24	10	63
Unweighted Good N	72	28	24	19	10	62
Sums	164	59	48	55	19	145
Average	2.25	2.36	2.04	2.32	1.91	2.31
Standard deviation	0.50	0.49	0.46	0.48	0.31	0.50
S.D. squared	0.25	0.24	0.21	0.23	0.10	0.25
Standard error	0.06	0.10	0.10	0.10	0.10	0.06
Mode	2	2	2	2	2	2
Q1	2.00	2.00	2.00	2.00	2.00	2.00
Q2	2.00	2.00	2.00	2.00	2.00	2.00
Q3	3.00	3.00	2.00	3.00	2.00	3.00
High value	1	2	1	2	1	1
Low value	3	3	3	3	2	3
High - Low	2	1	2	1	1	2

The last group is used for labels of the three types of missing values that normally appear in the MISSING section of the table. Note that when alternate labels are supplied for missing values using SURVEY.LABELS, they apply to *all* tables defined in the SURVEY subcommands. When they are supplied for a specific variable in

the label file, they apply only to tables in which that variable appears and they override the SURVEY.LABELS missing labels.

Figure 6.1 illustrates the use of SURVEY.LABELS. It contains a labels file with the special labels for many of the statistics. The table is the same as the table in Figure 5.12 except for the substitution of the special labels. Notice that by supplying an empty label for the extended label for variable Weight and by supplying another empty label for the code 202 there is no longer any line at the head of the table indicating the weight variable.

6.2 Special Characters

The CHARACTER subcommand is used to provide special characters in a variety of situations Some of these have already been described.

In the chapter “SURVEY: Creating Simple Tables” the BREAK character, which provides control over the way the value labels are broken across lines, was discussed in detail. CHARACTER for VALUE.LABELS was discussed in the chapter ”SURVEY: Special Features for the Layout Sections”, CHARACTER for ZERO was discussed briefly in the chapter “SURVEY: Creating Simple Tables”. The other forms of the CHARACTER subcommand are:

1. CHARACTER for VARIABLE.LABELS
2. CHARACTER for NEWLINE
3. CHARACTER for PERCENT
4. CHARACTER for SUBTOTALS
5. CHARACTER for UNDERLINE

The characters that are used if no special characters are provided are:

1.	BREAK	asterisk for break suggestions, not sign for no break. These characters are embedded in value labels at appropriate places.
2.	VARIABLE LABELS	equal sign for fill and both border characters
3.	VALUE LABELS	blanks for fill and border characters
4.	SUBTOTALS	dash for fill and border characters
5.	PERCENT	percent sign is assumed, characters 2 and 3 are set to blank
6.	UNDERLINE	see section below. Where possible the underline character is used
7.	ZERO	dash is supplied for the first line in an empty cell. Other lines are left blank.
8.	NEWLINE	no default. Provide an absolute breaking point for labels..

6.3 Subtotals and Nets

Nets and subtotals can be given a level number which is used to step the indentations making the patterns of subtotaling easier to see.

```
DEFINE   '1 to 4 TV sets'   Num.TV 1 TO 4,
DEFINE 2 '1 to 3 TV sets'   Num.TV 1 TO 3,
DEFINE 3 '1 or 2 TV sets'   Num.TV 1 2,
```

when an indent level is not specified, level 1 is assumed. Each indent level beyond the first increases the distance from the edge of the table by two print positions. Figure 6.2 illustrates how the indent levels are formatted. Figure 6.2 also illustrates the special character for the subtotals. These characters underline the subtotal labels.

```
CHARACTER FOR SUBTOTALS '=' '<' '>',
```

Figure 6.2 Special Characters and Define with Levels

```
SURVEY Paceset,  LABELS 'Paceset.lab';
  BANNER Age Own,  STUB Num.TV,  NO MISSING,  NO SUMMARY,
  DEFINE   '1 to 4 TV sets'   Num.TV 1 TO 4,
  DEFINE 2 '1 to 3 TV sets'   Num.TV 1 TO 3,
  DEFINE 3 '1 or 2 TV sets'   Num.TV 1 2,
  MARGIN 16,  INDENT 6,  SUBTOTALS BEFORE LOW VALUE,
  CHAR for ZERO ' ' ,    CHAR PERCENTS ' ',
  CHAR for VARIABLE.LABELS   '*' '|' '|',
 CHAR for UNDERLINE  '_', CHAR for SUBTOTALS '=' '<' '>' ;  $
```

		Age			Respondent ownership	
	Total Sample	Under 30	30 to 50	Over 50	no	yes
Total Sample	80	28	27	23	11	69
	100.0	100.0	100.0	100.0	100.0	100.0
Number of television sets in your home:						
1 to 4 TV sets	73	27	26	18	10	63
<=========>	91.2	96.4	96.3	78.3	90.9	91.3
1 to 3 TV sets	46	26	13	6	7	39
<=======>	57.5	92.9	48.1	26.1	63.6	56.5
1 or 2 TV sets	28	18	8	2	4	24
<=======>	35.0	64.3	29.6	8.7	36.4	34.8
1	6	4	1	1	1	5
	7.5	14.3	3.7	4.3	9.1	7.2
2	22	14	7	1	3	19
	27.5	50.0	25.9	4.3	27.3	27.5
3	18	8	5	4	3	15
	22.5	28.6	18.5	17.4	27.3	21.7
4	27	1	13	12	3	24
	33.8	3.6	48.1	52.2	27.3	34.8
5 or more	7	1	1	5	1	6
	8.8	3.6	3.7	21.7	9.1	8.7

The first character in quotes is the basic fill character. If the other characters are not supplied, it is used for all positions. The second character is used for the leftmost (border) character. The third character is the right

most (border) character. If this subcommand is not used, the subtotals are underlined with dashes. If blanks are supplied, the line is omitted.

Figures 6.2, 6.3, and 6.4 illustrate various uses of the CHARACTER subcommand. In Figure 6.2 any cells with a zero frequency are blank filled. The character that represents the percent sign is changed to a blank. The bordering characters which define the variable labels and the subtotals are changed. The value labels for the columns are underlined. (The problem of underlining is discussed further in the next section of this chapter.)

6.4 Variable Labels

CHARACTER FOR VARIABLE.LABELS is similar to CHARACTER FOR SUBTOTALS. It permits 1-3 single character arguments. The second and third arguments, if used, replace the left and right most characters respectively and serve to border the label area. If the underline character is used for all three arguments, the variable name is underlined rather than filled This is illustrated in Figure 6.7.

The subcommand CHARACTER can be abbreviated to CHAR and the word “for” is not required. This type of abbreviation is permitted throughout the subcommand language. However, aggressive use of abbreviation usually produces a command that is hard to read. In addition, as new features are added to SURVEY, the aggressive abbreviation may no longer be unique, causing a syntax error when the subcommand is parsed.

Usually the first line in a cell with no cases, an empty cell, is represented with a dash and the remaining lines of that cell are left blank. The first character following a CHARS for ZERO subcommand is used for the first line in the cell. The second character, if present, is used in all the remaining lines of that cell.

Figure 6.3 Formatting Empty Cells.

```
SURVEY Paceset,   LABELS 'Paceset.lab';
   BAN Income.groups,  STUB Age,
   LAYOUT LABELS TOTALS QUESTION BODY,
   CELL.WIDTH 8,
   CHAR ZERO '0' '0' ;
```

```
                        ====== Income of  =====
                        ====== respondent =====

                                   $30,000
                    Total   Under      to    Over
                   Sample $30,000 $60,000 $60,000

   Total              80       2      50      20
   Sample          100.0%  100.0%  100.0%  100.0%

   Age

   Under 30           28       0      18      10
                    35.0%    0.0%   36.0%   50.0%

   30 to 50           27       2      19       3
                    33.8%  100.0%   38.0%   15.0%

   Over 50            23       0      13       6
                    28.7%    0.0%   26.0%   30.0%
```

In Figure 6.3, the request is for a table with zeros in every line of the empty cell. If the table is to be exported to another program such as a text editor, it is often easier if all cells have either a zero or the "-" as a space holder.

```
CHAR ZERO '-' '-',
```

6.5 Newline Character

The BREAK characters are suggestions which may or may not be honored by the label formatter. The newline character specifies exactly where the breaks are to occur. These requests will be honored as long as the chunksof label fit within the constraints of the area. For example banner points are limited to 6 rows and one less than the current cell width. The variable labels are limited to 6 rows with a width determined by the number of banner points. Stub labels and labels for the subtotals and nets are limited by the specified margin but they can now use up to 24 rows. A newline character at the beginning of a label, skips a line and prints the first chunk on the second line of the cell.

Figure 6.4 shows the entry in the labels file, the command and the output when CHARACTER NEWLINE is used. The NEWLINE character is detected only by the SURVEY label formatter. It is not used for the QUESTION area because there is plenty of space for the question and it does not need to be broken into smaller pieces.

Figure 6.4 The NEWLINE Character

```
---------- The relevant entry in the labels file ----------

( Num.TV <Number of \television sets in \your home:' >

---------- The command ----------

SURVEY Paceset, LABELS 'paceset.lab';
CHARACTER NEWLINE '\',
BANNER Num.TV,  STUB Sex,
NO MISSING, NO SUMMARY;
$

---------- The output ----------

                    =======     Number of      =======
                    ======= television sets in =======
                    =======     your home:     =======

              Total                                 5 or
             Sample      1      2      3      4    more

Total            80      6     22     18     27      7
Sample        100.0% 100.0% 100.0% 100.0% 100.0% 100.0%


Male             39      4     18     11      4      2
                48.8%  66.7%  81.8%  61.1%  14.8%  28.6%

Female           40      2      4      7     22      5
                50.0%  33.3%  18.2%  38.9%  81.5%  71.4%
```

6.6 Character for PERCENTS

If a single character is provided following a CHARACTER for PERCENTS subcommand, that character replaces the percent sign throughout the table. When a second CHARACTER for PERCENTS is supplied, the first is used for the first line of percents that is printed. The second character is then used for the remaining PERCENTS.

Figure 6.5 illustrates the effects of CHAR for PERCENT. The first line in the table that contains percents occurs in the totals area. Therefore, that is the line which has the first character specified for percents (a percent sign). The lines in the body of the table which represent percents have the second character (a blank) replacing the percent sign. An optional third character is used only when the contents of the cell produce a percent that is greater than zero but less than .05 .

Figure 6.5 Changing the Percent Sign

```
SURVEY Paceset,   LABELS 'Paceset.lab';
   BAN Income.groups,  STUB Age,
   LAYOUT LABELS TOTALS QUESTION BODY,
   CELL.WIDTH 8,
   CHAR PERCENT '%'  ' ' ;
```

```
                          ====== Income of  =====
                          ====== respondent =====

                                      $30,000
                       Total   Under      to    Over
                      Sample $30,000 $60,000 $60,000

   Total                 80       2      50      20
   Sample             100.0%   100.0%  100.0%  100.0%

   Age

   Under 30              28       -      18      10
                       35.0             36.0    50.0

   30 to 50              27       2      19       3
                       33.8   100.0    38.0    15.0

   Over 50               23       -      13       6
                       28.7             26.0    30.0
```

6.7 When the Value Labels are Enough

There are times when the value labels adequately describe the contents of a given column in the table and the variable name is not wanted. A blank extended variable label can be used to replace the variable name and CHARACTER for VARIABLE.LABELS can be set to a blank.

```
   CHARACTER VARIABLE.LABELS ' ',
```

However, this removes the border character from all banner variables not just the variable with the blank extended label. If you wish to retain the variable labels with border characters for some variables and use just the value labels for other characters:

1. Do not set CHARACTER VARIABLE.LABELS to blank.
2. Use a single blank character for the extended variable labels that should be left blank.

Figure 6.5 illustrates the three ways that the labels can be formatted. The first example has the default character (=) for the variable label and the labels file contains an extended variable label for Sex which contains multiple blanks. The second example uses the default character for the variable label borders and the extended variable label for Sex contains a single blank. The third example has one or more blanks for the extended variable label for Sex and sets the character for variable labels to a blank. This affects all banner variables.

Figure 6.6 Blank Extended Variable Labels

```
SURVEY Paceset,  LABELS 'Blankxl.lab';
BANNER Sex Age, STUB Own, LAYOUT LABELS;

-----------------------------------------------------------

 Blankxl.lab includes:
 Sex <   > (1) Male (2) Female

                  ======  ===== ======== Age =======

            Total                Under  30 to   Over
           Sample   Male Female     30     50     50

-----------------------------------------------------------

Blankxl.lab includes:
Sex < > (1) Male (2) Female

                                   ======== Age =======

               Total                Under  30 to   Over
              Sample   Male Female     30     50     50

-----------------------------------------------------------

 With subcommand added:
 CHARACTER VARIABLE.LABELS < >

                                        Age

            Total                Under  30 to   Over
           Sample   Male Female     30     50     50
```

6.8 Controlling Underlining

The labels in the banners of tables produced by SURVEY are not underlined when the tables print on the terminal — underlining is not possible on terminals. When the tables are directed to a disk file or an on-line print-

er, the banner labels are underlined on systems that support underlining and they are not underlined on those that do not support underlining. It is possible to specify pseudo underlining or to force underlining on many systems.

Figure 6.7 Underling Variable Labels

```
SURVEY Paceset,  LABELS 'Paceset.lab';
   BAN Age Sex,  CHAR VARIABLE.LABELS '-' '<' '>',
   STUB Income.group,a
   NO MISSING,   NO SUMMARY,  NO SKIP;
```

		<------- Age ------>			<--- Sex --->	
	Total Sample	Under 30	30 to 50	Over 50	Male	Female
Total Sample	80	28	27	23	39	40
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Income of respondent						
Under $30,000	2	-	2	-	-	2
	2.5%		7.4%			5.0%
$30,000 to $60,000	50	18	19	13	21	28
	62.5%	64.3%	70.4%	56.5%	53.8%	70.0%
Over $60,000	20	10	3	6	10	10
	25.0%	35.7%	11.1%	26.1%	25.6%	25.0%

```
AGAIN, CHAR VARIABLE.LABELS '_' '_' '_';
```

		Age			Sex	
	Total Sample	Under 30	30 to 50	Over 50	Male	Female
Total Sample	80	28	27	23	39	40
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Income of respondent						
Under $30,000	2	-	2	-	-	2
	2.5%		7.4%			5.0%
$30,000 to $60,000	50	18	19	13	21	28
	62.5%	64.3%	70.4%	56.5%	53.8%	70.0%
Over $60,000	20	10	3	6	10	10
	25.0%	35.7%	11.1%	26.1%	25.6%	25.0%

Pseudo underlining, which is underlining in the line below the banner labels rather than in the same line, is suitable for any output device — terminals, disk files or printers. It is requested by using CHARACTER (or CHAR) in a SURVEY subcommand:

```
SURVEY  Project5,  LABELS  'Project5.lab' ;
  BANNER  Age,  STUB  Region,  CHAR FOR UNDERLINE  '_' ;
```

Here, the underscore is supplied as the underline character. Any character (=, -, *, etc.) can be supplied. The pseudo underlining appears on the line below the labels section containing the banner variable and value labels.

On systems that support *true* underlining (CMS, for example, but not PC/windows or UNIX), it is sufficient to include UNDERLINE in the PP command that specifies the attributes of the disk file or printer:

```
PP  'DiskFile',  OW  80,  UNDERLINE  $
```

Then, tables directed to this disk file have true underlining, while tables directed to the terminal do not. True underlining uses the underscore character, and the underline appears directly under (and on the same line as) the banner labels.

On systems that do not usually support true underlining (PC/windows and UNIX), *forced* underlining can be specified. The UNDERLINE identifier is included in the SURVEY command and CHAR is included in the subcommand:

```
SURVEY  Project5,  LABELS  'Project5.lab',  UNDERLINE ;
  BANNER  Age,  STUB  Region,  CHAR FOR UNDERLINE  '_' ;
```

The tables printed on the terminal have pseudo underlining and tables directed to a disk file or printer have true underlining, if it is at all possible. True underlining is indicated by a "+" sign at the beginning of the line. The beginning "+" sign is a signal to printers or print filters that recognize Fortran control characters that the printer should not advance a line but instead should overprint the text on the current line.

These issues of underlining pertain to tables that are ultimately printed on dot matrix or line printers, and not to tables with PostScript print controls that are printed on laser printers. It can be useful to produce a set of practice tables with various options and to direct them to different print destinations in order to evaluate the results and select the best options.

Underlining the names of the banner variables when system level underlining is not supported requires a different approach. The characters that are associated with variable labels usually fill and border the label area. However, if all three of the character arguments are the underscore character, pseudo-underlining is done.

```
CHAR VARIABLE.LABELS '_' '_' '_'.
```

Figure 6.7 illustrates the two usages of CHARACTER VARIABLE.LABELS.

6.9 Commas in Large Numbers

It is easier to read a table with very large numbers when a comma is used in the formatting. The down side is that it requires more space to print a number with commas than it does without commas. When space is allocated for the columns of a table, the setting for CELL.WIDTH and the actual amount of space that is required to format the number are both considered. When the COMMAS subcommand is used, large numbers are printed with commas properly placed. Figure 6.8 illustrates the resulting printout when the COMMAS subcommand is used.

Figure 6.8 Commas in Large Numbers

```
SURVEY Paceset,
         LABELS 'Paceset.lab';
   BAN Income.groups, STUB Sex,
   MEANS Income, INCLUDE MISSING MEANS VALUES,
   NO BODY, NO MISSING,
   COMMAS;
 $
```

```
                          ===== Income of respondent =====

                   Total      Under $30,000 to       Over
                  Sample    $30,000    $60,000    $60,000

Total                 80          2         50         20
Sample             100.0%     100.0%     100.0%     100.0%

Sex

Base                  71          2         49         20
                    88.8%     100.0%      98.0%     100.0%
Mean Income    34,434.65  18,400.00  29,970.41  46,975.50
S.D.            9,395.35     565.69   4,965.42   4,347.32
```

6.10 Spread or Compact Formatting

The width of the columns that represent the values of the banner variables is usually determined by the setting of the CELL.WIDTH subcommand. The width of columns containing summary information, such as means or the values of a quantity variable, is never less than the cell width but may be larger if more space is needed to print the numbers with the appropriate number of decimal places.

Figure 6.9 Cell Width, Spread and Compact Formatting of the Columns

```
SURVEY Paceset,   LABELS 'Paceset.lab';
   BAN Income.groups, STUB Sex,

   MEANS Income,
   INCLUDE MISSING MEANS VALUES,
   PLACES MEANS 2,
   LAYOUT LABELS SUMMARY,
   SUMMARY CONTAINS MEANS S.D,
   CELL.WIDTH 5;
$
```

```
                        = Income of  =
                        = respondent =

                             $30,
                        Unde  000
                         r $ to $ Over
                  Total  30,  60, $60,
                 Sample  000  000  000

    Mean Income  34435  18K  30K  47K
    S.D.          9395  566 4965 4347
```

Note: No room for all the digits. 18K used instead of 18400

AGAIN, CELL.WIDTH 6;

```
                    === Income of  ==
                    === respondent ==

                           $30,
                            000
                    Under  to $  Over
              Total  $30,   60,  $60,
             Sample   000   000   000

Mean Income   34435 18400 29970 46976
S.D.           9395   566  4965  4347
```

Note: No room to print the fractional part.

AGAIN, CELL.WIDTH 8;

```
                      ====== Income of  =====
                      ====== respondent =====

                                $30,000
                Total   Under      to    Over
               Sample $30,000 $60,000 $60,000

Mean Income   34434.6 18400.0 29970.4 46975.5
S.D.           9395.4   565.7  4965.4  4347.3
```

Note: Room for only a single digit of the fractional part.

AGAIN, COMPACT;

```
                      ====== Income of  =====
                      ====== respondent =====

                                $30,000
                Total   Under      to    Over
               Sample $30,000 $60,000 $60,000

Mean Income  34434.65 18400.0 29970.4 46975.5
S.D.          9395.35  565.69 4965.42 4347.32
```

Note: S.D and Means are now separately formattted.

AGAIN, SPREAD;

```
                        ==== Income of respondent ===

                                     $30,000
                  Total     Under        to      Over
                 Sample   $30,000   $60,000   $60,000

Mean Income   34434.65  18400.00  29970.41  46975.50
S.D.           9395.35    565.69   4965.42   4347.32
```

Note: Cell width is ignored. Space is allocated so that 2 places print.

When a variable other than the stub variable is used, or when weights are used to estimate a very large population, the cell may not be wide enough to print the number to the requested precision. Figure 6.8 illustrates the way that SURVEY handles this problem. In the first example, the numbers that do not fit are represented in thousands. 18K = 18,000. If even larger numbers are involved 18M is used to represent 18,000,000. In the second example, the numbers fit nicely without the requested 2 decimal places. A cell width of 8, used in the third example, allows a single place to be printed. In this situation all of the statistics in the summary section are printed using the same number of decimal places.

If the COMPACT subcommand is added, the cell width is still honored, but each statistic is separately formatted. In Figure 6.9 using COMPACT, the standard deviation is printed with both decimal places, even though there is only room to print the means with a single decimal place. In the final example, the SPREAD subcommand overrides the cell width subcommand and each column is given enough space to print all the numbers completely.

NOTE: when the COMMAS subcommand is used, SPREAD is assumed.

6.11 TABBED AND SPREADSHEET OUTPUT

If the printed output is to be imported into another program for additional processing, it can be useful to have tabs or some other special character between the columns rather than blanks. When the TABS identifier is used all blanks are compressed out of the tabular output and either a tab or another special character is placed between each field. Such output can be brought into spreadsheets such as Excel or word processing or publishing software and formatted precisely as desired.

The major difference between output created with the TABS identifier and that created by the SPREADSHEET identifier is in the formatting of strings such as the variable and value labels. In the TABS output the string output is formatted exactly like the printed tables. In the SPREADSHEET output these labels are not broken across lines but take as much space as needed in the appropriate cell.

6.12 TABS Output

The *identifier* TABS requests tabbed output:

```
SURVEY  PaceSet,  LABELS  'PaceSet.Lab',  PR  'PaceSet.lst',  TABS ;
```

If TABS is *not* followed by an argument, the tab character is placed between the fields. Other special characters are selected by following TABS with one of the keywords:

```
COMMA  or  COLON  or  SEMICOLON  or  VBAR (vertical bar)

SURVEY  PaceSet,  LABELS 'PaceSet.Lab',  PR 'PaceSet.lst',
   TABS  COMMA ;
```

TABS should not be used with POSTSCRIPT or any of the FONT and FORMAT identifiers, because they are incompatible The tabbed output is a single page so it is important to specify an output width that is large enough to hold all the columns. Currently a single page can contain 90 columns plus the row labels and the row totals. An output width of 1500 should be large enough.

Tabbed output is typically directed to a disk file, which is subsequently imported into word processing or publishing software. When tabbed output goes to the terminal or other default destination, it either appears as continuous lines of characters, lines of characters with single blanks between the columns (the non-printing tab character), or as tables with arbitrary tabs (the default tab locations in UNIX, for example, are every 8 columns).

What is placed between the columns is the ASCII or EBCDIC representation of the non-printing tab character. Depending on the software into which the output is imported, the tabs may show as a special symbol, a control character (control-I in UNIX editors, for example) or a blank, or they may not show at all. However, once tabs are defined at specific locations, the output will “jump” into alignment. The tab positions should be adjusted so that the columns are attractively spaced and the table fits on the page. Figure 6.9 shows a table before and after

using “TABS VBAR”. “NEVER SKIP” is frequently used with tabbed output. It causes all blank lines to be omitted from the output.

Figure 6.10 TABS subcommand for Delimited Output

```
SURVEY Paceset,   LABELS 'Paceset.lab';
   BAN Num.TV (M MED), STUB Sex,  NO MISSING,
   NO SKIP, NO SUMMARY;


                    ====== Number of television sets in your home: ======

              Total                                    5 or
             Sample      1      2      3      4       more     Mean    Median

Total            80      6     22     18     27        7     3.09      3.00
Sample        100.0% 100.0% 100.0% 100.0% 100.0%   100.0%

Sex

Male             39      4     18     11      4        2     2.54      2.00
                48.8%  66.7%  81.8%  61.1%  14.8%   28.6%
Female           40      2      4      7     22        5     3.60      4.00
                50.0%  33.3%  18.2%  38.9%  81.5%   71.4%
```

```
 SURVEY Paceset,   LABELS 'Paceset.lab',  TABS VBAR;
   BAN Num.TV (M MED),
   STUB Sex,
   NO MISSING,
   NO SKIP,
   NO SUMMARY;

||====== Number of television sets in your home: ======||||||

|Total|||||5 or||
|Sample|1|2|3|4|more|Mean|Median

Total|80|6|22|18|27|7|3.09|3.00
Sample|100.0%|100.0%|100.0%|100.0%|100.0%|100.0%|||

Sex

Male|39|4|18|11|4|2|2.54|2.00
|48.8%|66.7%|81.8%|61.1%|14.8%|28.6%||
Female|40|2|4|7|22|5|3.60|4.00
|50.0%|33.3%|18.2%|38.9%|81.5%|71.4%||
```

6.13 SPREADSHEET Output

The SPREADSHEET ***identifier*** can also have an argument specifying the character that is used to delimit the values. If a delimiter is not specified the TAB character is assumed. Several other settings are automatically done.

1. Output width is set the the widest possible output line. This is currently 5,000 characters.
2. Number of lines is set to many million.
3. Margin area is set to 80 which is currently the longest allowed value label.
4. All blank lines are turned off.
5. Titles are joined into a single string for each title line that is specified. Each title line is printed left justified.
6. If there is a single banner variable, the variable label or extended variable label is placed in the middle cell of the body as a single long string. If there are multiple banner variables, the labels are in the leftmost cell for each variable.
7. A subcommand: QUOTES ON can be used to place quotes around all character strings except titles, and the lines which print the BY variable and Weight variable values. QUOTES OFF is the default.
8. Underlining and table.titles are not supported.

The following 2 lines from figure 6.10:

```
|Total|||||5 or|||
|Sample|1|2|3|4|more|Mean|Median
```

would be just this single line in SPREADSHEET output:

```
|Total Sample|1|2|3|4|5 or more|Mean|Median
```

6.14 OTHER FORMATTING OPTIONS

A table can be positioned on the page by using the TOP.EDGE and LEFT.EDGE subcommands. When the printout is going to a character printer (not PostScript) the TOP.EDGE argument is the number of lines to skip before the first line of the table and the LEFT.EDGE argument is the number of columns on the left edge to be skipped before each line begins.

The bottom edge is controlled by the LINES setting. This can be provided as part of the printer setting, as an identifier in the command or as a subcommand. The right edge is controlled by the output width setting. OUTPUT.WIDTH can be provided as part of the printer setting, as an identifier or as a subcommand.

If different settings are made for either LINES or OUTPUT.WIDTH, the subcommand setting takes precedence over the identifier setting, which takes precedence over the printer parameter settings.

6.15 PLACES Subcommand

The PLACES subcommand is always followed by a keyword indicating which statistic is being considered and by an integer which supplies the number of places desired. The keywords that can be used include:

1. COUNTS
2. MEANS
3. MEDIANS
4. SUMS
5. COUNTS

Figure 6.10 illustrates all five of the PLACES controls. Percents are usually printed with 1 decimal place unless the cell width is set to 5. When there are no places for percents they may not appear to add to 100. PLACES for COUNTS in the body and for SUMS in the summary section are usually printed as integers, without any fractional part. However, when a table is weighted, they may well have a fractional part which will not be seen unless you use the PLACES subcommand. Means and medians are usually printed to 2 places unless the PLACES subcommand is used. They are individually controlled. Changing the places for means does not affect medians.

Figure 6.11 The 5 PLACES subcommands

```
SURVEY Pacesetw,    LABELS 'Paceset.lab',   WEIGHT Weight;
   BANNER Age Own,  STUB Num.TV,
   CELL.WIDTH 9,  GAP 2,
   NO MISSING,
   SUMMARY CONTAINS  MEANS  MEDIAN  SUMS,

   PLACES PERCENTS 2,
   PLACES MEANS 3,
   PLACES MEDIANS 0,
   PLACES COUNTS 1,
   PLACES SUMS 1 ;
```

Weighted by: weight

		Age			Respondent ownership	
	Total Sample	Under 30	30 to 50	Over 50	no	yes
Weighted Total	82.0 100.00%	24.8 100.00%	26.4 100.00%	28.8 100.00%	11.4 100.00%	70.6 100.00%
Number of television sets in your home:						
1	5.8 7.06%	3.6 14.43%	1.0 3.65%	1.2 4.34%	1.0 8.42%	4.8 6.84%
2	20.7 25.22%	12.4 50.13%	7.0 26.34%	1.3 4.49%	3.0 26.19%	17.7 25.06%
3	18.1 22.08%	7.0 28.23%	5.0 18.78%	5.1 17.82%	2.7 23.82%	15.4 21.80%
4	29.3 35.71%	.9 3.61%	12.5 47.44%	14.9 51.68%	3.5 30.65%	25.8 36.53%
5 or more	8.1 9.92%	.9 3.61%	1.0 3.78%	6.2 21.68%	1.2 10.92%	6.9 9.76%
Mean	3.162	2.318	3.214	3.819	3.094	3.173
Median	3	2	4	4	3	3
Total	259.3	57.5	84.8	110.0	35.4	223.9

6.16 Controls for the use of Blank Lines

The SKIP subcommand permits control over blank lines in the body of the table. A special form of SKIP:

```
NEVER SKIP
```

produces a table with no blank lines. This is useful only when the table is to be saved for editing in some other program. SKIP.RULES now permits greater control over blank lines before a given section and between items in the section. The arguments that are supported for SKIP.RULES ARE:

1. QUESTION ON/OFF
   The argument is used to control the blank line before the QUESTION (XL) section of a table. ON is the default.
2. BODY ON/OFF
   The first argument controls the blank line before the section. The second controls the blank line between cells within the body. ON ON is the default.
3. SUBTOTALS ON/OFF
   Like the arguments for BODY. The second argument only applies if there are two or more subtotals together. ON ON is the default.
4. MISSING OFF/ON
   The default for MISSING is ON OFF. The second argument is only used if MISSING OBSERVED or MISSING ALL are used.
5. SUMMARY OFF/ON
   The default for SUMMARY is ON OFF. IF the second argument is set to ON, a blank line is inserted between each item in the summary section.
6. LABELS ON/OFF
   The default for LABELS is ON ON. If the second argument is OFF, the blank line that usually separates the variable labels from the value labels is omitted.
7. TOTALS.AREA ON/OFF
   Since the totals area contains just a single item, the second parameter is not supported. The default is ON: a blank line before the totals area section.
8. F.CHI ON/OFF
   The default is a blank line before any F or CHI test results and a blank line between the groups if both are requested.
9. TITLES ON/OFF
   TITLES is used to control spacing of both the regular top titles and also the information that is included if either WEIGHT or BY are used. The assumption is a blank line before any titles and blank lines before WEIGHT or BY information. The first argument insures no blank line at the top of the page before the titles. The second argument controls the blanks before the WEIGHT and BY information.
10. TABLE.TITLES ON/OFF
    This controls the line that is inserted before TABLE.TITLES. Blank lines can be included within the table titles if they are desired.

Figure 6.12 illustrates the use of SKIP.RULES when used to control the blank lines in the SUMMARY section. There is no blank line between the counts and percents in the base as they comprise a single element in the summary. However the use of:

```
SKIP.RULES SUMMARY ON ON
```

causes the blank line that usually precedes the summary section to be printed and a blank line to be inserted after the base and after the line with the means.

Figure 6.12 SKIP.RULES: Summary Section

```
SURVEY Paceset,   LABELS 'paceset.lab';
BANNER Age, STUB Sex, NO MISSING;
```

	Total Sample	Age: Under 30	Age: 30 to 50	Age: Over 50
Total Sample	80 100.0%	28 100.0%	27 100.0%	23 100.0%
Sex				
Male	39 48.8%	20 71.4%	11 40.7%	7 30.4%
Female	40 50.0%	8 28.6%	16 59.3%	15 65.2%
Base	**79** **98.8%**	**28** **100.0%**	**27** **100.0%**	**22** **95.7%**
Mean	**1.51**	**1.29**	**1.59**	**1.68**
S.D.	**0.50**	**0.46**	**0.50**	**0.48**

AGAIN, SKIP.RULES SUMMARY ON ON;

	Total Sample	Age: Under 30	Age: 30 to 50	Age: Over 50
Total Sample	80 100.0%	28 100.0%	27 100.0%	23 100.0%
Sex				
Male	39 48.8%	20 71.4%	11 40.7%	7 30.4%
Female	40 50.0%	8 28.6%	16 59.3%	15 65.2%
Base	**79** **98.8%**	**28** **100.0%**	**27** **100.0%**	**22** **95.7%**
Mean	**1.51**	**1.29**	**1.59**	**1.68**
S.D.	**0.50**	**0.46**	**0.50**	**0.48**

The first of the ON/OFF argument pairs is always used to determine whether a blank line precedes the item. The second of the ON/OFF argument pairs is used to determine whether interior blank lines should be printed.

```
SKIP.RULES BODY ON OFF
```

has the same effect as the SKIP 0 subcommand. Using all of the ten possible SKIP.RULES subcommands with arguments OFF OFF is like using NEVER SKIP.

6.17 Joining Titles

The JOIN subcommand can be used to combine the sections of an individual title line into a single string. The string can be positioned so that it is left, center or right justified.

```
JOIN T2 LEFT
```

causes any titles for the second title line to be joined into a single title which is left justified in the output. Title lines that are not specifically joined are printed with each section individually placed.

```
JOIN ON RIGHT
```

causes all title lines to be joined. The final argument should be one of LEFT, CENTER or RIGHT. If it is omitted, CENTER is assumed. "JOIN OFF" can be used to turn off title joins.

6.18 Formatting the Labels

The number of lines permitted for stub and subtotal labels is 24 . The formatting depends on the margin. With MARGIN set to 16, the subtotal label:

```
TOTAL WILLIAM JEFFREYS HOSPITAL CORPORATION SUBSIDIARY
```

is printed as:

```
TOTAL WILLIAM
JEFFREYS
HOSPITAL
CORPORATION
SUBSIDIARY
```

6.19 Treatment of Empty Tables

The assumed behavior prints all tables whether or not there is any good data. If you want to eliminate any tables that have no data you can use the EMPTY.TABLES subcommand. By an empty table we mean a stub/banner combination which contains no data. Either an empty stub variable or banner variables that happen to have no data can cause an empty table. If the problem is in the banner variables, the table is not empty if any of the banner variables has good data. SQUEEZE COLUMNS can be used to leave out individual banner variables which have no good values.

This creates a problem when using GR.STUBS. The default assumption is that EMPTY.TABLES includes any GR.STUB which would be an empty table if defined as an ordinary STUB variable. The subcommand looks like:

```
EMPTY.TABLE OFF          is the same as
EMPTY.TABLE ALL OFF
```

The subcommand:

```
EMPTY TABLE ON            is the same as
EMPTY TABLE ALL ON
```

This causes the tables to be treated in the usual fashion (i.e., displayed).

The GR.STUB question can also be handled by using the EMPTY.TABLE subcommand. To request that all empty tables should be omitted except those that are specified as GR.STUB use:

```
EMPTY.TABLE OFF,
EMPTY.TABLE  GROUPED  ON,
```

If you wish to have only those empty GR.STUB tables omitted, use:

```
EMPTY.TABLE ON,
EMPTY.TABLE  GROUPED  OFF,
```

The EMPTY.TABLE subcommand does not work with the transposed table format. Table numbers are not increased for an empty table. For GR.STUBS, the table number increases if any of the grouped tables have data.

Multiple response stub variables are handled in the same way. If there is no data on any of the multiple response stub variables, that table is considered empty. If any of the multiple response stub variables has even a single non-missing value, the table is not empty.

6.20 Lead Blanks in the Printout

Usually each line in the printout begins with a single blank. This is used in some situations as a carriage control character that tells the printer what to do next. In many situations the lead blank has no effect except to use space. This can be controlled by using the identifier NO LEAD.BLANK in the command.

```
Survey Paceset, LABELS 'pace.lab',  NO LEAD.BLANK;
```

The identifier LEAD.BLANK can also be used and is the default.

SUMMARY

SURVEY

```
 SURVEY  PaceSet,   LABELS  'PaceSet.Lab',   TABS;

    BANNER  Age  Sex,  STUB  Income.Groups,  MEAN    Income,
    PLACES  MEANS  0,  CELL.WIDTH  9,       CHAR  FOR  PERCENT ' ' ;
$
```

This chapter illustrates a variety of ways to enhance the appearance of the table through a choice of contents, an arrangement of the pieces of the table and the use of special characters.

Optional Identifiers:

DECIMAL.COMMA

Use the European format for writing out numbers. The comma becomes the delimiter between the whole and the fractional parts of the value. The decimal point is used as the separator between thousands. I.e., the number 1,234.56 will be displayed as 1.234,56.

NO LEAD.BLANK

causes the single blank character that usually begins each line of the printout to be omitted.

TABS

specifies that the SURVEY output use *tab characters* instead of blanks between columns of numbers and in other places where blanks normally appear. This is useful when the output is to be brought into publishing software for editing and font changes. The output appears compressed until the tab locations are set, and then the output appears aligned. TABS should *not* be used with POSTSCRIPT nor have any of the FONT and FORMAT options.

TABS COMMA / COLON / SEMICOLON / VBAR

has the same effect as TABS with no argument except that the designated character is used instead of the tab character to separate the fields.

SPREADSHEET COMMA / COLON / SEMICOLON / VBAR

Similar to TABS but with additional features. Variable and value labels are NOT divided across lines but appear in the single relevant cell. Maximum values are automatically set for output width and lines. The QUOTES subcommand may be used to cause these labels to be enclosed in quotes.

Optional Subcommands:

CHARACTER FOR arg “cs” “cs” “cs”

provides characters to underline the banner points, represent blank cells, label percentages, force or prevent breaks in labels, and/or fill in around value labels, variable labels and subtotals. The first argument is one of the keywords: BREAK, PERCENTS, SUBTOTALS, UNDERLINE, VALUE.LABELS, VARIABLE.LABELS, ZEROS or NEWLINE. It determines which characters are replaced. The arguments that follow are single characters in quotes to be used instead of the usual characters.

To replace the dash that is printed for an empty cell use:

```
 CHARACTER  ZEROS  '.' ,
```

This is used for the first line in the cell. The lines that follow are left blank unless a second character is provided.

```
CHARACTER  ZEROS  '.'  '.',
```

To replace the percent sign that prints after all percents when there is a cell width of 6 or more:

```
CHAR PERCENTS ' '
```

A blank will remove the percent sign. If two characters are used, the first replaces the percents signs in the first line of percents and the second replaces the percent signs in all the following lines. A third character, if provided, is used when the contents of the cell are not zero but are so small that they cannot print. In this situation an asterisk is used unless the CHARACTER PERCENTS has a third argument to replace the asterisk

Three fill characters can be provided for the VARIABLE.LABELS which apply to column (banner) variables, for the VALUE.LABELS (only for the outer values when there are nested banners) and for SUBTOTALS. The first character is the normal fill character. The second is used for the start (left-most) fill position and the third is used for the end (right-most) fill position. Thus:

```
CHAR VARIABLE.LABELS  '-'  '<'  '>' ,
```

produces a banner variable label such as this:

```
<-- Age -->
```

```
CHAR VARIABLE.LABELS '_' '_' '_',
```

produces a banner variable label with pseudo underlining such as:

```
    Age
_________
```

Underlining of the banner value labels is done automatically when the output device supports underlining (printers sometimes do and terminals generally do not). When underlining is not supported, an underline character can be specified:

```
CHAR FOR UNDERLINE  '=',
```

It prints under the label of each banner point.

A break character other than the assumed one (the asterisk) may be supplied for value labels. In this example, labels are broken (if necessary) where a question mark is found in the labels file:

```
CHARACTER FOR BREAK '?' '=' '=',
```

The question mark does not print in the label. A second *no-break* character (an equal-sign here) may be supplied. It is used in labels to ensure that they do not break at blanks and to center or justify the labels. The equal-sign prints as a blank in labels. (Note that if a label does not fit, it will break nonetheless.) The third character is only used when you need to control the break in the extended labels for the banner variables.

CHARACTER may be abbreviated to CHAR. The word FOR is optional.

COMMAS

Use commas in formatting all large numbers. When COMMAS is used, SPREAD is assumed.

COMPACT

Honor the cell width for the columns representing the values of a banner variable. This is slightly different from the assumed formatting which formats all of the statistics that depend on the mean using the same number of places. When COMPACT is in effect, the formatter formats these statistics individually.

EMPTY.TABLES ALL/GROUPED OFF/ON

EMPTY.TABLES determines the way that empty tables are handled. The default is to print them which is the same as EMPTY.TABLES ALL ON. EMPTY.TABLES ALL OFF causes all empty tables including empty parts of a GR.STUB table to be omitted. The table number is not increased when an empty table is skipped.

GROUPED may be used to control the tables which use GR.STUB.

```
EMPTY.TABLES ALL OFF,  EMPTY.TABLES GROUPED.ON
```

This causes all empty tables except the sections with an empty GR.STUB variable to be omitted. If the first argument (ALL or GROUPED) is omitted, ALL is assumed. ON or OFF must be specified.

JOIN T1-T9 LEFT / RIGHT / CENTER

JOIN is used with the regular titles when a single title longer than 80 characters is needed. The first argument indicates which title line is to have its individual left/right/center portions joined. The second argument is used to indicate how the new longer title is to be justified.

JOIN ON LEFT/RIGHT/CENTER causes all defined title lines to be joined and appropriately justified.

LEFT.EDGE nn

when the printout is not destined for PostScript, LEFT.EDGE specifies the number of print columns on the left edge of the page which should be left blank.

PLACES arg nn

The argument that follows PLACES can be any of the following:

```
COUNTS  MEANS  SUMS   PERCENTS   MEDIANS
```

The number of decimal places that are to be printed follows the argument.

QUOTES ON / OFF

This is used with SPREADSHEET to control the usage of quotes around the variable and value labels text.

SKIP.RULES arg ON/OFF ON/OFF

SKIP.RULES is used with any of the following list of arguments to control the blank lines that precede a section or are placed within a section of the printout. Possible values for the first argument are:

```
QUESTION      BODY        SUBTOTALS     MISSING
SUMMARY       LABELS      TOTALS.AREA   F.CHI
TITLES        TABLE.TITLES
```

SPREAD

The cell width is taken as a minimum in formatting the columns representing the values of a banner variable. Additional space is allocated as needed to print the values in full with the requested number of places for the fractional part.

TOP.EDGE nn

when the printout is not destined for PostScript, TOP.EDGE specifies the number of lines from the top of the page that are to be left blank before the table prints.

SURVEY.LABELS codes

```
(1)   Total Sample      (2)   Base              (3)   Responses
(4)   Unweighted Total  (5)   Unweighted Base
```

```
(11)  Total            (12)  Mean             (13)  S.D.
(14)  Variance         (15)  Standard error   (16)  Median
(17)  Mode             (18)  Maximum          (19)  Minimum
(20)  Range            (21)  First quartile   (22)  Third quartile
(23)  Percentiles

(41)  F value          (42)  t value          (43)  Probability
(44)  Chi Square       (45)  DF for Chi       (46)  Fisher 2-tail
(47)  Yates chi-sq....  (48)  Some cells had   (49)  Cramer's V

(101) Total Sample     (102) Mean (banner)    (103) Good.N (banner)
(104) Median (banner)  (105) Totals (quantity) (106) Percentiles
(110) within

(120) Counts (quantity) (121) Row %           (122) Col %
(123) Tot %

(202) Weighted by:     (203) Page             (204) Table
(205) Contents

(210) BY               (211) missing          (212) Overall Totals
(213) of list          214) Missing (mean/median variable)

(301) Label for significance tests of means
(302) Label for test of proportions
(303) Label for test of independence
(304) Label when combined values are used in test

(m1)  missing 1        (m2)  missing 2        (m3)  missing 3
```

the next five values are used only when REPORT ALL RESULTS is used and none of the tests have been significant.

```
(310) used with TEST ALL RESULTS and CHI
(311) used with TEST ALL RESULTS and F
(312) used with TEST ALL RESULTS and TEST PROP
(313) used with TEST ALL RESULTS and TEST INDEPENDENCE
(314) used with TEST ALL RESULTS and TEST MEANS
```

7
SURVEY: PostScript Output

The first part of this chapter covers the use of PostScript controls to produce "camera-ready" with proportional fonts and spacing controls. PostScript output requires either a PostScript printer or the availability of a program such as Acrobat which takes PostScript output and converts it to an alternate format. Acrobat produces files in pdf format which can be processed and printed from the Acrobat reader, a program that is available for most computing environments without charge.

7.1 POSTSCRIPT OUTPUT

Tables produced by the SURVEY command can be output with PostScript codes suitable for a laser printer which has PostScript capability. Various line and box styles and combinations of fonts can be selected. The orientation of the table on the page can be specified. Many of the PostScript controls can be supplied as identifiers or as subcommands. The identifiers that can be used are:

1. POSTSCRIPT PORTRAIT or LANDSCAPE
2. FONT, BFONT, TFONT, LFONT, and QFONT
3. LEFT.EDGE and TOP.EDGE
4. CHARACTER.SET
5. FORMAT

POSTSCRIPT requests tables output with typesetting codes suitable for PostScript printers such as the Apple LaserWriter and similar laser printers. The PR identifier is also used with POSTSCRIPT to direct the output to a disk file or on-line laser printer. When PR is not included in the command, the PostScript control *codes* (not tables) appear on the terminal (or other default output destination). When the PostScript output has been directed to a disk file, it can be printed after exiting P-STAT, using the appropriate operating system command. All of the PostScript identifiers are available as subcommands. When the word POSTSCRIPT is encountered, the necessary header information for PostScript printout is written to the output file. If you have not yet defined a print file, this heading information will print on your terminal. A PR subcommand applies only to the current table. If you are doing several tables, use PR as an identifier.

```
SURVEY Paceset,  LABELS 'Paceset.lab',   PR 'Pace.ps';
POSTSCRIPT,
```

SURVEY can also be used within a POSTSCRIPT block as one of several commands to be sent to the same PostScript print file. When the POSTSCRIPT block is started with a POSTSCRIPT command, the print destination and the POSTSCRIPT identifier should not be used. All SURVEY commands issued within a PostScript block are automatically formatted for PostScript and sent to the proper print file. POSTSCRIPT, POSTSCRIPT.CLOSE and POSTSCRIPT.SETUP are described in the manual "P-STAT: Plots, Graphs and Postscript Support".

Figure 7.1 shows the default orientation, format and font of a table produced when POSTSCRIPT is specified. The default orientation depends on the output width of the print destination. When it is 80, the table has *portrait* (vertical) orientation; when it is 132 or greater, it has *landscape* (horizontal or rotated 90 degrees) orientation. The default format has the banner and stub variable names or labels underlined. The default font is Times Roman. The default point size is 10 for output widths up to 80, 8 for widths up to 132, and 7 for widths greater than 132. Various optional identifiers can be used to specify the orientation, format and fonts for the tables.

Figure 7.1 Default PostScript Table: Format 0 and Times Roman

```
SURVEY  PaceSet,  TITLES,

LABELS  'PaceSet.lab',     PR  'PaceSet.ps',
POSTSCRIPT,
FONT  'TIMES' 8,
FORMAT 1 ;

BANNER  Age  Sex,
STUB  Income.Groups,
MEAN    Income,
PLACES  MEANS  0,  CELL.WIDTH  9,
CHAR  FOR  PERCENT ' ' ;
$
```

Default PostScript Table for Output Width 80

		Age			Sex	
	Total Sample	Under 30	30 to 50	Over 50	Male	Female
Total Sample	72	28	24	19	31	40
	100.0	100.0	100.0	100.0	100.0	100.0
Income of respondent						
Under $20,000	2	-	2	-	-	2
	2.8		8.3			5.0
$20,000 to $40,000	50	18	19	13	21	28
	69.4	64.3	79.2	68.4	67.7	70.0
Over $40,000	20	10	3	6	10	10
	27.8	35.7	12.5	31.6	32.3	25.0
Missing	-	-	-	-	-	-
Base	72	28	24	19	31	40
	100.0	100.0	100.0	100.0	100.0	100.0
Mean Income	34398	37388	29311	35580	36759	32634
S.D.	9334	9156	8038	8450	9340	9149

The OUTPUT.WIDTH (OW) general identifier can be used in a PRINT.PARAMETERS command or in the SURVEY command to set the output width. The arguments PORTRAIT and LANDSCAPE can follow the POSTSCRIPT identifier to explicitly set the orientation. Similarly, the FONT identifier or subcommand can be included to define the font and its point size. Additional font identifiers specify fonts for various portions of the tables. The FORMAT identifier or subcommand requests the desired table format.

The LEFT.EDGE and TOP.EDGE subcommands specify the left and top margins. The right and bottom margins default to the space remaining after the table prints. The arguments for LEFT.EDGE and TOP.EDGE give the number of *inches* from the edge of the paper that printing should start. They should be positive numbers and they can be fractional. Depending on the point size of the font and the width of the table, printing can run off the edges of the page when large numbers are specified. When LEFT.EDGE and TOP.EDGE are not used, printing begins one inch from the left and one inch from the top of the page. RIGHT.EDGE and BOTTOM.EDGE subcommands are only used when a border is placed around the table. This is described later in this chapter.

CHARACTER.SET is used to provide the accented characters that are needed for many European languages.

```
CHARACTER.SET  ISO
```

provides information for most of the characters in the ISO 8859-1 specifications. "CHARACTER SET GERMAN" can be used to limit the characters to those that were provided in earlier releases using the UMLAUTE.ANSI identifier. The abbreviation CH.SET is supported. Settings from an earlier POSTSCRIPT.SETUP command are honored.

7.2 Specifying Box and Line Styles

There are currently three supported styles or formats for PostScript tables — regular, journal and boxed. They are designated by number:

0. *regular*
   SURVEY table format with no lines or boxes, except for underlining of the banner variable name (label) and the stub variable name (question). The underline of the stub variable name can be removed by using FORMAT -1. FORMAT -2 removes the underline of the banner variable labels and FORMAT -3 removes them both.
1. *journal*
   format with lines above and below the table, lines enclosing the column totals (totals area), and with underlining of the banner and stub variable names. Double lines are used between tables on the same page.
2. *boxed*
   format with the cells in the body of the table (excluding the row totals) boxed, and with underlining of the banner and stub variable names as in format 0.

See Figures 7.1, 7.2, and 7.3 for examples of these three formats. The desired format is requested by number, using the FORMAT identifier:

```
FORMAT  2,            boxed format
FORMAT -1,            regular format without banner label underline
```

Changes to the table layout, such as those created using the LAYOUT or TRANSPOSE subcommands, and the length and number of lines of labeling affect the format appearance — see Figure 7.3.

7.3 Specifying Fonts

The FONT *subcommand* supplies a font and point size to be used for the entire table including the titles. Other font identifiers, BFONT, LFONT, QFONT and TFONT supply fonts and point sizes for specific parts of the table — for the body, labels, question and titles, respectively.

There can be from one to three arguments for any of these subcommands. The arguments can be a font name by itself, a font name and a style, a font name and a point size, or a font name and style and point size. In addition, a quoted string giving an exact font name and style can be supplied in place of the first and second font arguments.

The three generic names and styles that can be given as the first or first and second arguments are:

```
TIMES              COURIER              ARIAL
TIMES BOLD         COURIER BOLD         ARIAL BOLD
TIMES ITALIC       COURIER OBLIQUE      ARIAL ITALIC
```

Notice that ITALIC is used with TIMES and ARIAL and that OBLIQUE is used with COURIER to specify a slanted style. Figure 7.1 shows the TIMES (Times Roman) font. Figure 7.2 shows the ARIAL ITALIC font. COURIER font can be seen in nearly all the figures in this manual. (It is the only nonproportional font commonly available, and thus emulates a terminal or other nonproportional-spacing output device.)

An optional second or third argument giving the point size can follow the font name or font name and style:

```
FONT  'ARIAL BOLD'  10,
```

Figure 7.2 PostScript Table: Format 1 and ARIAL ITALIC

```
SURVEY  PaceSet,
LABELS  'PaceSet.lab',   PR 'PaceSet.ps',
POSTSCRIPT,
FONT  'ARIAL ITALIC' 8,
FORMAT 1 ;

BANNER  Age  Sex,
STUB  Income.Groups,
MEAN    Income,      PLACES  MEANS  0,
CELL.WIDTH  9,       CHAR  FOR  PERCENT ' ' ;
$
```

```
File  PaceSet.ps:
```

Default PostScript Table for Output Width 80

		Age			Sex	
	Total Sample	Under 30	30 to 50	Over 50	Male	Female
Total Sample	72	28	24	19	31	40
	100.0	100.0	100.0	100.0	100.0	100.0
Income of respondent						
Under $20,000	2	-	2	-	-	2
	2.8		8.3			5.0
$20,000 to $40,000	50	18	19	13	21	28
	69.4	64.3	79.2	68.4	67.7	70.0
Over $40,000	20	10	3	6	10	10
	27.8	35.7	12.5	31.6	32.3	25.0
Missing	-	-	-	-	-	-
Base	72	28	24	19	31	40
	100.0	100.0	100.0	100.0	100.0	100.0
Mean Income	34398	37388	29311	35580	36759	32634
S.D.	9334	9156	8038	8450	9340	9149

For most output, font sizes of 8, 10, or 12 points are recommended. However, you can use any font size supported on your printer. When a combination that is not available is specified, either the table does not print or a default font such as Courier is used. Certain font/style/size combinations can be too large for the output width or

the table orientation. When that is the case, only a portion of the table may print. Reduce the font point size or the number of lines and the output width.

A precise font name/style combination can be supplied as a quoted argument for FONT or for any of the other font subcommands. The name must be in the correct case with the correct spacing, and it should be a font that is available on your laser printer. This command:

```
SURVEY  PaceSet,   LABELS  'PaceSet.Lab',    PR  'PaceSet.ps',
POSTSCRIPT,        FONT 'AvantGarde-BookOblique'  10 ;
```

produces tables using the specified font in point size 10. The font, with this precise capitalization and hyphen, is recognized by PostScript on the LaserWriter Plus.

Figure 7.3 PostScript Transposed Ratings Table: Format 1 and Title Font

```
SURVEY   Ratings,       LABELS  'Ratings.Lab',  PR  'Ratings.ps',
POSTSCRIPT  PORTRAIT,   FORMAT 1,
FONT  TIMES  8,         TFONT  ARIAL  BOLD  10 ;

  TRANSPOSE,  GROUP.STUBS  Q1  TO  Q5,     CELL.WIDTH 6,
  LAYOUT   LABELS  TOTALS  BODY  SUMMARY,  MARGIN 20,
  TITLE  'Soft Dessert Ratings' ;
```

File Ratings.ps:

Soft Dessert Ratings

	Total Sample	Poor	Medi ocre	Okay	Good	Excel lent	Base	Mean	S.D.
How would you rate the flavor?	10 100.0%	2 20.0%	-	3 30.0%	3 30.0%	2 20.0%	10 100.0%	3.30	1.42
the taste?	10 100.0%	1 10.0%	3 30.0%	-	2 20.0%	4 40.0%	10 100.0%	3.50	1.58
the texture?	10 100.0%	-	1 10.0%	4 40.0%	-	4 40.0%	9 90.0%	3.78	1.20
the color?	10 100.0%	-	1 10.0%	1 10.0%	3 30.0%	3 30.0%	8 80.0%	4.00	1.07
the aroma?	10 100.0%	1 10.0%	1 10.0%	1 10.0%	4 40.0%	2 20.0%	9 90.0%	3.56	1.33

Different fonts, styles and/or sizes can be supplied for the various parts of tables.

1. BFONT supplies font information for the *body* of tables. The default or specified fonts are used in the other parts of the tables.

2. LFONT gives font information for the *labels* portion of tables. Labels include the banner variable names or extended labels, the banner variable value labels, and the stub value labels. The stub variable name or extended label (the question) is not included.
3. QFONT supplies font information for the *question*.
4. TFONT gives the font for any top and bottom titles.

Figure 7.4 shows a table with a larger, bold face title supplied for emphasis. FONT and TFONT give the overall and title font information, respectively.

7.4 Controlling Title Fonts

The font for each line of the titles can be individually controlled. Usually the font associated with the subcommand "TFONT" is used for all titles lines. This font is the 3rd font of the 9 fonts that can be defined in the POSTSCRIPT and POSTSCRIPT.SETUP commands. In these commands the TITLES identifier is used to associate a specific titles line with a particular font name.

As many as 9 different font styles and sizes can be provided for the titles. However, all the sections of a title line (left, center and right) must have the same font and point size.

```
POSTSCRIPT.SETUP,  FONT5 TIMES BOLD 14,
                   TITLE TOP 1 5 $
```

The TITLES identifier has 3 required arguments. The first argument is either "TOP" or "BOTTOM". The second argument is a number, 1-9 for top titles and 1-3 for bottom titles. The final argument is a number between 1 and 9 indicating the font (FONT1 - FONT9) that is to be used for that titles line. Both the FONTn and the TITLE identifiers should be used. It is advisable to use only FONT3 and FONT5 - FONT9 when defining titles as the other 3 fonts are associated with BFONT (FONT1), LFONT (FONT2), and QFONT (FONT4). This is not a problem if, for example, you wish to use the same font for a title line as the font that is to be used for the body of the table.

Each font must be entered individually. The names are FONT1 - FONT9. The titles information can be entered as individual TITLE identifiers or as a single TITLES identifier with many arguments. The following example provides a very large font for the top title in FONT5 and a very small font for the first bottom title in FONT6.

```
POSTSCRIPT.SETUP,  FONT5 TIMES BOLD 14,  FONT6 TIMES 6,
                   TITLE TOP 1 5 BOTTOM 1 6 $
```

These fonts must be defined before the SURVEY command that will use them. The SURVEY command needs no changes. The titles are defined as usual. When a title that has a special font is printed, that font is used instead of the font associated with TFONT (FONT3). It is not necessary to skip lines if you have a font with a large point size. The next title will be spaced appropriately. However, if you are using very large fonts for your titles, you may need to reset the number of lines on a page to avoid printing beyond the end of the page. Figure 7.6 contains the commands which produce the tables in Figure 7.7. This figure illustrates the use of special fonts for titles as well as the BORDER and SHOWPAGE subcommands described below.

The font that is used for TABLE.TITLES can be controlled by:

```
USE TABLE.TITLES QFONT  (or BFONT, TFONT or LFONT)
```

The assumed font for TABLE.TITLES is TFONT, the font that is usually used for titles.

7.5 Joining Titles

The JOIN subcommand can be used to combine the sections of an individual title line into a single string. The string can be positioned so that it is left, center or right justified.

```
JOIN T2 LEFT
```

causes any titles for the second title line to be joined into a single title which is left justified in the output. Title lines that are not specifically joined are printed with each section individually placed.

Figure 7.4 MARGIN and SCALE

Default Settings: SCALE .3

		Age			Sex	
	Total Sample	Under 30	30 to 50	Over 50	Male	Female
Compact Disc Player	28 35.0%	10 35.7%	13 48.1%	5 21.7%	12 30.8%	16 40.0%
Portable Telephone	35 43.8%	15 53.6%	8 29.6%	10 43.5%	21 53.8%	14 35.0%
Answering Machine	37 46.2%	18 64.3%	12 44.4%	7 30.4%	20 51.3%	16 40.0%

SCALE .5

		Age			Sex	
	Total Sample	Under 30	30 to 50	Over 50	Male	Female
Compact Disc Player	28 35.0%	10 35.7%	13 48.1%	5 21.7%	12 30.8%	16 40.0%
Portable Telephone	35 43.8%	15 53.6%	8 29.6%	10 43.5%	21 53.8%	14 35.0%
Answering Machine	37 46.2%	18 64.3%	12 44.4%	7 30.4%	20 51.3%	16 40.0%

MARGIN .9 Inches, Default Scale Value

		Age			Sex	
	Total Sample	Under 30	30 to 50	Over 50	Male	Female
Compact Disc Player	28 35.0%	10 35.7%	13 48.1%	5 21.7%	12 30.8%	16 40.0%
Portable Telephone	35 43.8%	15 53.6%	8 29.6%	10 43.5%	21 53.8%	14 35.0%
Answering Machine	37 46.2%	18 64.3%	12 44.4%	7 30.4%	20 51.3%	16 40.0%

```
JOIN ON RIGHT
```

causes all title lines to be joined. The third argument should be one of LEFT, CENTER or RIGHT. If it is omitted, CENTER is assumed. "JOIN OFF" can be used to turn off title joins. JOIN can be used in any SURVEY output even though it is not PostScript output. However, its major use is in PostScript output where proportional fonts allow more characters in each output line.

7.6 Controlling the Margins

The basic layout of a SURVEY table is determined by the number of characters that must be placed in each column. This is works well for the body of the table as all the numbers are scaled to the same width, even in a proportional font. The area for the stub labels is a problem when proportional fonts such as Times-Roman are requested because the real width of the strings is not readily available to the SURVEY command and can vary considerably from one font to another.

The assumption is made that a label in a proportional font uses somewhat less than 70% of the space required by a monospace font. Therefore, the area that is allotted for the margin in the PostScript output is scaled down by 30%. If the label is all upper case wide characters such as M, O, and W, it may overflow into the body of the table. However, if the label consists largely of lower case letters such as i and l, there is extra space before the body of the table.

SCALE is a subcommand, is used to change the scaling factor. The assumed setting is .3. A larger setting decreases the margin area. A smaller setting increases the margin. The number should be a fraction between 0 and 1. If you are using mostly lower case characters, a setting of .4 will usually produce a satisfactory result. The actual formatting of the labels, that is the number of characters that will be placed on a single line, depends on the MARGIN setting. SCALE determines the offset of the body of the table from the margin area.

The MARGIN subcommand can be used by itself to provide some control over this spacing. If you provide a number that is less than 8, it is interpreted as a measurement in inches. The appropriate number of characters to be allowed per line is calculated from this number. This subcommand:

```
MARGIN 1.3,
```

causes the program to allocate approximately 1.3 inches for the margin area. If you are using an 8 point font, the label formatter will place up to 25 characters in each line of the stub value label. This will break up the labels using the same logic that is used when "MARGIN 25" is specified but the actual space that is used will be less.

If you provide both a margin in inches and the number of columns to be used in the margin area, the final usage wins. You may use the combination of MARGIN and SCALE to further adjust the margin area. Figure 7.4 contains 3 examples which illustrate the output when SCALE and MARGIN are used.

7.7 Providing a Border

In PostScript output the entire table can be surrounded by a border. This border is usually placed 1/2 inch outside the margins of the table. The subcommand that is used to get a border is:

```
BORDER,
```

The line width is usually about the width of the underscore character. This can be controlled by specifying a wider setting with the LINE.WIDTH subcommand.

```
LINE.WIDTH BORDER 1.5,
```

If the bottom edge is not specified, the LINES setting is used to calculate the proper location. The placement of any bottom titles depends on the LINES setting which may need to be adjusted if both BORDER and bottom titles are used. If RIGHT.EDGE is not used, the placement of the right edge of the border depends on the width of the table. If RIGHT.EDGE is used, it must be the distance from the left edge of the paper, not the width of the right margin. In Figure 7.5, the right edge is set at 6.5 inches. This is 2 inches from the physical right side of an 8.5 by 11 inch sheet of paper.

A border is usually associated with a single table. If you are constructing a page with several tables, draw the border with the table which is to be placed at the top left of the paper, providing a bottom and a right margin large enough to contain all the tables you wish incorporated. Then when you define the next table, use "NO BORDER" so that a second border will not be drawn.

Figure 7.5 PostScript Multi-Page Command

```
POSTSCRIPT.SETUP,
  FONT5 TIMES BOLD 14,       TITLES TOP 1 5,
  FONT6 TIMES  6,            TITLES BOTTOM 1 6  $


SURVEY Paceset, LABELS 'Paceset.lab',
                   PR 'pace.ps';

  POSTSCRIPT PORTRAIT, NO SHOWPAGE,
  TITLE 'Test Run. .DATE.',
  TITLE B1 LEFT '*footnotes look best in small type',

  TOP.EDGE 1.75,   LEFT.EDGE 2.,
  BOTTOM.EDGE 2,   RIGHT.EDGE 6.5,

  BORDER,   LINE.WIDTH BORDER 1.5,

  NO MISSING,
  MEANS INCOME,   PLACES MEANS 0,
  BANNER Age (M) Own, STUB Income.group;

  NO BORDER,  TOP.EDGE 5.2,
  NO TITLES,
  BANNER Income.groups (M), STUB Num.TV,
  SHOWPAGE;                                    $
```

7.8 SHOWPAGE: Multiple Tables Per Page

The subcommands, SVAR.PER.PAGE and GROUP.STUBS, that are used when printing character mode tables with similar banners and other attributes, work equally well with PostScript output. In addition, PostScript permits you to place two or more very different tables on the same page. This is possible because the PostScript output is not written out one line at a time but is "drawn" on the paper. It is possible to place text at selected (or random) places on the page almost indefinitely. The images are collected until the PostScript command "SHOWPAGE" is encountered. SHOWPAGE is the signal to take what has been drawn so far and commit it to paper (or a screen).

In the SURVEY command, it is the adroit use of "NO SHOWPAGE" and "SHOWPAGE" combined with the use of "TOP.EDGE" and "LEFT.EDGE" that permits multiple tables to be placed on the page. SURVEY usually issues the "SHOWPAGE" command whenever the lines counter indicates a complete page or a new STUB variable is encountered.

Figure 7.6 PostScript Multi-page Output

Test Run. Thu May 26 1994

		Age				Respondent ownership	
	Total Sample	Under 30	30 to 50	Over 50	Mean Income	no	yes
Total Sample	72 100.0%	28 100.0%	24 100.0%	19 100.0%	34174	10 100.0%	62 100.0%
Income of respondent							
Under $20,000	2 2.8%	-	2 8.3%	-	18400	1 10.0%	1 1.6%
$20,000 to $40,000	50 69.4%	18 64.3%	19 79.2%	13 68.4%	30007	9 90.0%	41 66.1%
Over $40,000	20 27.8%	10 35.7%	3 12.5%	6 31.6%	46800	-	20 32.3%
Base	72 100.0%	28 100.0%	24 100.0%	19 100.0%	34174	10 100.0%	62 100.0%
Mean Income	34398	37388	29311	35580		26785	35626
S.D.	9334	9156	8038	8450		5410	9277

		Income of respondent			
	Total Sample	Under $20,000	$20,000 to $40,000	Over $ 40,000	Mean Income
Total Sample	72 100.0%	2 100.0%	50 100.0%	20 100.0%	34398
Number of television sets in your home:					
1	6 8.3%	1 50.0%	3 6.0%	2 10.0%	34592
2	21 29.2%	-	15 30.0%	6 30.0%	36213
3	12 16.7%	-	10 20.0%	2 10.0%	32096
4	26 36.1%	1 50.0%	18 36.0%	7 35.0%	33023
5 or more	7 9.7%	-	4 8.0%	3 15.0%	37843
Base	72 100.0%	2 100.0%	50 100.0%	20 100.0%	34398
Mean Income	34398	18400	30007	46976	
S.D.	9334	566	4921	4347	

*footnotes look best in small type

However, if you use the “NO SHOWPAGE” subcommand, SURVEY does not issue a PostScript SHOWPAGE until *you* enter a “SHOWPAGE” subcommand. This makes it possible to combine tables from two different files on a single page.

The consequence of this flexibility is a responsibility. *You* must remember the final SHOWPAGE. If you forget it and send the printout to your printer, nothing will print. When you are planning multiple pages of this type, the use of the POSTSCRIPT commands described in the manual “P-STAT: PostScript Support” can be helpful.

```
POSTSCRIPT, LANDSCAPE, PR 'Test.ps' NO SHOWPAGE$

SURVEY School1, LABELS 'School.lab';
  BANNER Grade, STUB Attendance, MEAN Score;
$
SURVEY School2, LABELS 'School.lab';
  BANNER Grade, STUB Attendance, MEAN Score, LEFT.EDGE 5.5;
$
POSTSCRIPT.CLOSE $
```

7.9 PostScript and the Lines Setting

The number of lines that can print when using PostScript in SURVEY depends on the size of the paper, the margins and the LINES setting for the PostScript printer. TOP.EDGE can be set in inches as a SURVEY subcommand. The subcommand BOTTOM.EDGE is only used to help place a border around the tables. The POSTSCRIPT.SETUP command must be used if you wish to change the initial setting of the BOTTOM.EDGE.

```
POSTSCRIPT.SETUP, BOTTOM.EDGE 1.25 $
```

The SURVEY command assumes a 1 inch bottom margin which controls the placing of bottom titles. However there is some slack in this computation which amounts to 3 times the point size of the body font plus 1 for the space between the lines (the leading). This slack is there to accommodate small changes in the point size of the titles. Changing the point size changes the number of lines that can fit on the page.

If you are printing a table in LANDSCAPE on standard 8.5x11 paper, there is 6.5 inches available. This is equivalent to 468 PostScript units which are measured as 72 units per inch. If the body font is set to the default of 8 points with a leading of 1, a LINES setting of 52 should fit easily with some allowance for larger titles. However, if you have several titles lines in a very large point size, you may need to reduce the LINES setting to avoid overflowing the page.

If you are printing a table in PORTRAIT orientation with 9 inches available, you can fit 58 lines per page even with a 10 point font size. Since the bulk of the table is printed in the BODY font, you can get a good estimate of the appropriate LINES setting by doing a trial run with the titles you plan to use.

7.10 PostScript Controls for Underlining Subtotals

The assumed behavior is underlining for the last subtotal label. The CHARACTER subcommand is used to turn underlining both on and off.

```
CHARACTER SUBTOTALS ' '            turns the underlining off
CHARACTER SUBTOTALS ' ' ' ' '-'    turns underlining back on
```

The use of the CHARACTER SUBTOTALS or NETS with any other characters will cause those characters to be used in the line below the last line of the subtotal/net label. This is only useful when you are using a monospace font.

CHARACTER SUBTOTALS '=' '<' '>’

causes an underlining which looks like:

```
<==========>
```

```
SURVEY SchoolA, LABELS 'School.lab', PR 'test.ps';
  POSTSCRIPT LANDSCAPE, SHOWPAGE OFF,
  BANNER Grade, STUB Attendance, MEAN Score;
$

SURVEY SchoolB, LABELS 'School.lab', PR 'test.ps';
  POSTSCRIPT LANDSCAPE, SHOWPAGE,  LEFT.EDGE 5.5,
  BANNER Grade, STUB Attendance, MEAN Score;
$
```

7.11 PostScript Compressed Format

The appropriate spacing for 9 point and larger fonts results in output that requires more space. To give more control over the output width the COMPRESS FORMAT subcommand is available.

```
COMPRESS FORMAT ON,
```

causes the spacing for individual letters to be allocated 1/72 for fonts 8 points or less and 2/72 of on inch less for 9 point or larger fonts. This allows more banner points per page and still produces reasonable looking output.

```
COMPRESS FORMAT OFF,
```

causes the formatting to revert to the default settings.

If your labels composed largely of narrow letters such as “i” and “l” you can squeeze in even more by supplying the appropriate fraction:

```
COMPRESS FORMAT 3 ON,
```

causes the spacing to be adjusted down by 3/72 of an inch. This spacing for the width of the letters also causes a small adjustment to the leading (the space between lines and between the text and a drawn underline.)

SUMMARY

SURVEY

```
TITLE  B2 'Page .PAGE.' $

 SURVEY  PaceSet,   LABELS  'PaceSet.Lab',  TITLES,
    POSTSCRIPT,     PR      'PaceSet.ps',   OW 80,
    FONT  ARIAL  ITALIC 8,   FORMAT 1 ;

    BANNER  Age  Sex,  STUB  Income.Groups,  MEAN    Income,
    PLACES  MEANS  0,  CELL.WIDTH  9,        CHAR  FOR  PERCENT ' ' ;
$
```

This chapter illustrates a variety of ways to enhance the appearance of the table by using the PostScript features.

Optional Identifiers:

CHARACTER.SET ISO / GERMAN

provides the necessary translations for the ISO 8859-1 character set to be written out as part of the PostScript instructions. If the argument GERMAN is used, the translations are for the subset of characters that are often used in German language documents.

Optional Subcommands:

POSTSCRIPT PORTRAIT / LANDSCAPE

requests that the table be output using PostScript codes suitable for a laser printer. (This identifier should *not* be used for output going to a dot matrix or line printer.) PORTRAIT or LANDSCAPE can be used as arguments to specify the orientation.

The PR *identifier* should be used with POSTSCRIPT to direct output to a disk file or on-line printer, or else the PostScript codes appear on the terminal. The additional identifiers supply optional selections for table formats, fonts and orientation. When they are not used, the PostScript tables are in the regular SURVEY format and in Times Roman font.

The output width of the print destination controls the point size of the font and whether the tables print in portrait or landscape (rotated 90 degrees) orientation. The defaults are portrait and point size 10 for output widths up to 80, and landscape and point size 8 for widths of 132 to 1200. OW, which supplies the output width, can be included in a PRINT.PARAMETERS command, in the SURVEY command or as a SURVEY subcommand. *NOTE: for output widths greater than 132, the subcommand must be used.* The LINES identifier can be included in the SURVEY command to define the page length.

FORMAT nn

supplies the number of the desired table format. Possible arguments for FORMAT are 0, 1 or 2. They correspond to these formats:

0. regular SURVEY table format with no lines or boxes, except for underlining of the banner variable name (label) and the stub variable name (question). The underlining for the stub and/or variable labels can be removed by specifying the number as -1, -2, or -3.
1. journal format with lines above and below the table and enclosing the column totals, and with underlining of the banner and stub variable names as above.

2. boxed format with the cells in the body of the table (excluding the row totals) boxed, and with underlining of the banner and stub variable names as in format 0.
3. not supported. If used becomes FORMAT 0.
4. format 4 is like format 1 except that the labels section is always underlined. With format 1 the labels section is only underlined when there are row totals.

Format 1 undlerlines both the banner variable name and the question section. These underlines can be omitted by using -1 to omit underlining of the banner variable names; -2 to omit the underlining of the question; and -3 to omit all underlining.

Changes to the table layout, such as those created using LAYOUT or TRANSPOSE, and the length and number of lines of labeling affect the format appearance.

FONT cs cs nn or 'cs' nn

supplies a font and point size to be used for the tables. There can be one, two or three arguments for FONT, and the arguments can be the supported keyword font names or a quoted exact font string and a point size.

These three names and/or styles can be supplied as the first or first and second arguments:

```
TIMES               COURIER               ARIAL
TIMES BOLD          COURIER BOLD          ARIAL BOLD
TIMES ITALIC        COURIER OBLIQUE       ARIAL OBLIQUE
```

An argument giving the point size can follow the font name or font name and style (the point size, if provided, is the final argument). There is no limit as such to the point size. However, some font/style combinations may not be available in all sizes.

Alternatively, the precise font name/style combination can be supplied in quotes as the first argument for FONT. The name must be in the correct case with the correct spacing, and it must be a font available on your laser printer:

```
FONT 'AvantGarde-BookOblique'  10 ,
```

This font, with this precise capitalization and hyphen, is recognized by PostScript on the Apple LaserWriter Plus. (See BFONT, LFONT, QFONT and TFONT also.)

BFONT cs cs nn or 'cs' nn

supplies font and point size information to be used for the *body* of the table. The default font or specified fonts are used in other portions of the table. See FONT for details about the permitted arguments.

LFONT cs cs nn or 'cs' nn

supplies font and point size information to be used for the *labels* of the banner and stub variables. (The question is not included in the category labels.) The default font or specified fonts are used in other portions of the table. See FONT for details about the permitted arguments.

QFONT cs cs nn or 'cs' nn

supplies font and point size information to be used for the *question* (stub variable name or extended label). The default font or specified fonts are used in other portions of the table. See FONT also.

TFONT cs cs nn or 'cs' nn

supplies font and point size information to be used for the table *title*. The default font or specified fonts are used in other portions of the table. See FONT.

LEFT.EDGE nn

gives the number of inches away from the left edge of the page that printing should start. When LEFT.EDGE is not used, printing begins one inch from the left of the page. The argument should be a positive number of inches and it can be fractional. Depending on the point size of the font and the width of the table, printing can run off the right edge of the page if a relatively large number of inches is specified. When LEFT.EDGE is used in a character (not PostScript) plot, the number is the number of print columns from the left that are to be skipped.

TOP.EDGE nn

gives the number of inches away from the top edge of the page that printing should start. When TOP.EDGE is not used, printing begins one inch from the top of the page. (See the discussion under LEFT.EDGE. When TOP.EDGE is used in a character (not PostScript) plot, the number is the number of print lines from the top that are to be skipped.

BOTTOM.EDGE nn

provides the distance in inches of the bottom edge of the table from the physical bottom of the page. This is used only for the border.

RIGHT.EDGE nn

provides the distance in inches that the right edge of the table is from the LEFT edge of the physical page. This is used only for the border.

BORDER

causes a solid line border to surround the table. This box is placed 1/2 inch outside the rectangle defined by the top left corner and the bottom right corner. If the edges have been explicitly defined, they are used in determining the placement. If the edges are not explicitly defined, the placement of the box depends on the size of the table.

COMPRESS FORMAT ON / OFF

causes the spacing of the font to be compressed by 1 or 2 seventy-seconds of an inch to allow for more columns per page.: The exact amount of the adjustment can be specified.

```
COMPRESS FORMAT 3 ON,
```

Use of OFF causes the spacing to revert to the default widths. When COMPRESS FORMAT is on, the spacing between a line and its underline is also slightly adjusted downwards.

JOIN arg arg

is used to join the pieces of a title line into a single long title which can be centered, left or right justified. The first argument is one of T1-T9, B1-B3, ON or OFF. The second argument is one of LEFT/RIGHT/CENTER. If the second argument is not used, CENTER is assumed. JOIN can be used even when the output is not in PostScript.

LINE.WIDTH BORDER nn

sets the width of the line used for the border. It is usually set at .5 which in PostScript equals 1/144 of an inch. A border of 1.5 makes an attractive medium weight border for a table.

MARGIN nn

sets the width of the left margin area in inches. This must be a number less than 8 and it may be fractional. When MARGIN is used this way, the SURVEY command estimates the number of characters to use in the labels.

SCALE nn

where nn is a number less than 1. This is used to estimate how much space the margin and the columns should use when the font is proportional. If the stub labels have many wide uppercase characters, the label may overflow the area used for the left margin. If the labels, on the other hand, are composed of many thin lower case characters, there may be extra white space in the labels area. If the labels are overflowing, lower the scale. If there is too much white space, increase the scale.

SHOWPAGE

is usually issued automatically by the SURVEY command at the end of each table or whenever the current LINES setting has been exceeded. NO SHOWPAGE turns off the automatic feature. This, with the use of TOP.EDGE and LEFT.EDGE permits you to place more than one table on a page. However, *you* must remember to issue a SHOWPAGE subcommand when the table is filled and ready for printing.

USE TABLE.TITLES TFONT / QFONT/ BFONT / LFONT

PostScript only, select the font to be used for TABLE.TITLES. TFONT is assumed.

8
SURVEY:The Statistics

The statistics that are produced by the SURVEY command range from counts and percents, the “bread and butter” of a tabulation package to t-tests, F-tests, chi-square, and some measures of cell significance. This chapter describes the statistics that are available and how to produce them. It discusses some of the pitfalls and problems but it is not a substitute for a knowledge of statistics. Interpreting even simple statistics must be done with caution.

The goal is not to teach statistics, but to provide the information you need to produce results. The first section talks about the basic statistics. The second section is devoted to the median, a statistic which appears at first glance to be simplicity itself, but proves to have a host of complications.

8.1 THE BASIC STATISTICS

The basic statistics are the counts and percents, the statistics included in the body of the table and the summary section. It includes chi-square, F-tests, and t-tests. The formulas used are standard text book formulas. There are a few options which control the ways that the numbers are calculated. There are more options selecting what to print.

The precision that a program uses for handling data and computations affects the accuracy of the results. Single precision means that a number is stored in a single word in the computer. Double precision uses two words to store each number. If the data and the computations are done in single precision, the results may differ and be less accurate than results produced in double precision. This is true even in survey data where most of the numbers are small integers. However, when you have many respondents and are accumulating squares and sums of squares to produce a variance, the numbers grow large rapidly. This is why all data files in P-STAT are stored in double precision and all the SURVEY calculations are double precision.

8.2 Counts

Unweighted counts seldom cause problems. If the data are carefully collected and validated, the counts in the cells are just what they purport to be. When the tables are weighted some interesting problems can arise. SURVEY assumes that the each entry in the cell is likely to be the value 1. Therefore, printing is usually done without any fractional part. When this happens with weighted tables, the rows and columns may not add up to the number that is printed as a row or column total. The solution for this is easy:

```
PLACES COUNTS 2,
```

Printing the weighted cell counts with one or two decimal places will produce a result that more closely matches the row and column totals.

In P-STAT all numeric data is read and stored in double precision. Thus, a very large number of significant digits can be handled before there is any loss of information. In the SURVEY command everything is also stored in double precision even the cells of the table. If you have a table with thousands of cells none of them larger than a few thousand, you may wish to specify single precision for the body of the table. The permits larger tables and more tables processed in each pass through the data file. All the calculations are still done in double precision.

```
SURVEY Paceset,  LABELS 'Paceset.lab',  SINGLE ;
```

DOUBLE is the default and is always the preferred setting when the data are weighted.

There are sometimes questions about interpretation when the data are weighted. You should know why and how any weights in your data are created. In SURVEY the statistics that are produced when weights are in effect are usually based on the weighted not the unweighted data. This must be considered when you evaluate the results.

8.3 PERCENTS

There is nothing mysterious about percents. They are simply a value divided by the total number with the result multiplied by 100. However, you must decide which total number you wish to use. In SURVEY, the assumption is that the total (weighted) count is to be used. This may be the row total, the column total or the total count for the table. These totals include all the cases even those with missing data. You may instead decide to use the totals that represent the good (non-missing) count or the number of responses rather than the number of respondents. These features of PERCENTS are covered in detail in the chapter "SURVEY: Creating Simple Tables".

Figure 8.1 COMPUTE: Changing the Base for Row Percents

```
SURVEY Paceset,
     LABELS 'Paceset.lab';
   BANNER Sex Sex WITHIN Own,
   STUB Age,
   NO MISSING, NO SUMMARY,
   ROW PERCENTS, COMPUTE 0 0 1 2 1 2;
```

		Sex / Respondent ownership					
		Sex		no		yes	
	Total Sample	Male	Female	Male	Female	Male	Female
Total	80	39	40	5	6	34	34
Sample	100.0%	48.8%	50.0%	12.8%	15.0%	87.2%	85.0%
Age							
Under 30	28	20	8	-	2	20	6
	100.0%	71.4%	28.6%		25.0%	100.0%	75.0%
30 to 50	27	11	16	4	2	7	14
	100.0%	40.7%	59.3%	36.4%	12.5%	63.6%	87.5%
Over 50	23	7	15	1	2	6	13
	100.0%	30.4%	65.2%	14.3%	13.3%	85.7%	86.7%

When row percentages are requested, they are normally based on the row totals. It is possible to request row percents based on other columns. This is done by using the COMPUTE subcommand followed by a vector with one number for each banner point (column) excluding the row totals. The numbers in the compute vector refer to the corresponding column as defined by the ranges of the banner variables. Columns that are SQUEEZed out are included in the COMPUTE vector. Columns that are added by banner variable extensions (M MED T) are not included. Thus:

```
BANNER Age (1,2) Income(1,4 M)  Education (1,3)
```

always has 9 columns as far as COMPUTE is concerned even when empty columns are not to be printed. If the number is zero, then the row totals are used. Figure 8.1 illustrates both the command and the printout when COMPUTE is used.

In this example:

```
BANNER  Sex   Sex  WITHIN  Own,  STUB  Age,
   ROW PERCENTS,  COMPUTE  0 0  1 2  1 2  ;
```

The first two columns (banner points) are the values 1 and 2 for Sex. They are followed by the values for variable Sex within each value of variable Own. Since Own has two values, there are six columns in all. Row percents for Sex (the first two columns) are based on the row totals. The columns within Own that are labelled “Male” have percents based on the values in column 1. The columns within Own that are labelled “Female” have percents based on the values in column 2.

Figure 8.2 Column Percents

```
SURVEY Paceset, LABELS 'Paceset.lab';
    BANNER Age, STUB Own, LAYOUT LABELS TOTALS QUESTION BODY;
$
```

	======== Age =======			
	Total Sample	Under 30	30 to 50	Over 50
Total	80	28	27	23
Sample	100.0%	100.0%	100.0%	100.0%
Respondent ownership of electronic items				
no	11	2	6	3
	13.8%	7.1%	22.2%	13.0%
yes	69	26	21	20
	86.2%	92.9%	77.8%	87.0%

```
AGAIN, NO.COL.100 PERCENT;
```

	======== Age =======			
	Total Sample	Under 30	30 to 50	Over 50
Total				
Sample	80	28	27	23
Respondent ownership of electronic items				
no	11	2	6	3
	13.8%	7.1%	22.2%	13.0%
yes	69	26	21	20
	86.2%	92.9%	77.8%	87.0%

When the column or row percents are based on the row or column totals, they are always 100%. This adds little interest to the table although it may help if there is any question about the base that is used in calculating the

percentages. When column percents are removed by adding NO.COL.100 to the PERCENTS subcommand the entire line is removed from the printout. This is illustrated in Figure 8.2.

The argument NO.ROW.100 percents has the much same effect on the rows that NO.COL.100 has on the columns. However, when the table has percents and no counts the row total replaces the meaningless “100%”. The command, subcommands and resulting printout are shown in Figure 8.3

Figure 8.3 Row Percents

```
SURVEY Paceset,   LABELS 'Paceset.lab';
    BANNER Age, STUB Own,
    LAYOUT LABELS TOTALS QUESTION BODY;
    ROW COL NO.ROW.100 NO.COL.100 PERCENT, NO COUNTS;
 $
```

	======== Age =======			
	Total Sample	Under 30	30 to 50	Over 50
Total	80	28	27	23
Sample		35.0%	33.8%	28.7%
Respondent ownership of electronic items				
no	11	18.2%	54.5%	27.3%
	13.8%	7.1%	22.2%	13.0%
yes	69	37.7%	30.4%	29.0%
	86.2%	92.9%	77.8%	87.0%

8.4 Controlling the Percents

There are several ways to control the printing of the percents. One way uses the OMIT.PERCENTS subcommand. A second way uses the COMPUTE subcommand with additional control information. When percents are requested it is assumed that they are to be calculated everywhere that there are frequencies available. The keywords NOPRINT.ROW.100 and NOPRINT.COL.100 are used with the PERCENTS subcommand when the somewhat meaningless 100% figure in the row and column totals is not wanted. The OMIT.PERCENTS subcommand provides even more control.

OMIT.PERCENTS can be followed by any one of the following keyword arguments:

```
TOTALS.AREA      BODY          SUBTOTALS
MISSING          SUMMARY       ROW.TOTALS
BANNER.TOTALS    RESET         ARITHMETIC.OP
```

The OMIT.PERCENTS subcommand can be used several times if needed to define a given table. Figure 8.4 shows a table which uses three of the OMIT.PERCENTS subcommands;

The use of OMIT.PERCENTS RESET, restores all setting to the default settings. Individual settings can be changed by use of OFF and ON.

```
OMIT.PERCENTS SUMMARY ON,          is the same as
OMIT.PERCENTS SUMMARY,
```

and causes all percents to be omitted from the summary section of the table.

```
OMIT.PERCENTS SUMMARY OFF,
```

causes the percents for the SUMMARY section to be restored. Any other settings remain unchanged. The argument ARITHMETIC.OP is used for the arithmetic operations described in the next chapter.

Figure 8.4 OMIT.PERCENTS: Subcommand and Output

```
----------The Command----------

SURVEY Paceset,  LABELS 'paceset.lab";
LAYOUT LABELS BODY MISSING,
ROW COL PERCENTS,
BANNER Age ( T ), STUB Sex,
OMIT.PERCENTS ROW.TOTALS,
OMIT.PERCENTS BANNER.TOTALS,
OMIT.PERCENTS MISSING;

----------The Table----------

                     =========== Age ===========

               Total  Under  30 to   Over
              Sample     30     50     50 Good N

 Male             39     20     11      7     38
                       51.3%  28.2%  17.9%
                       71.4%  40.7%  30.4%

 Female           40      8     16     15     39
                       20.0%  40.0%  37.5%
                       28.6%  59.3%  65.2%

 Missing           1      -      -      1      1
```

8.5 Omitting Percents from Specific Columns of the Table

Just as a 0 has a special meaning (compute the percents based on the row totals), negative column positions (-1 to -995) and the values -996, -997, -998 and -999 have special meanings.

This use of negative numbers in the COMPUTE subcommand makes it possible to:

1. Omit percents from selected banner points.
2. Mix row and column percents on a single line.
3. Produce row percents for all banner points and omit some column percents.
4. Produce column percents for all banner points and omit some row percents.

The use of -999 on a column reference causes all percents to be omitted from that column of the table..

```
COMPUTE 0 -999 -999
```

causes all percents to be omitted from the second and third banner points. The percents for the row total can be omitted by using the OMIT.PERCENTS subcommand described above. Note: when a COMPUTE subcommand does not contain values for all banner points, 0 (base percents on row totals) is assumed for those that are omitted.

Figure 8.5 Controlling the Percents

```
SURVEY paceset [ GEN good.n;
                 IF income.groups GOOD, SET good.n=1;],
       LABELS paceset.lab;
     BAN sex good.n  income.groups,
     STUB age,
     LAYOUT LABELS TOTALS QUESTION BODY,
     ROW PERCENTS, COMPUTE -997 -997 -999  3 3 3;
$
```

		==== Sex ====			==== Income of respondent ====		
	Total Sample	Male	Female	good.n	Under $30, 000	$30, 000 to $60, 000	Over $ 60,000
Total	80	39	40	72	2	50	20
Sample	100.0%	100.0%	100.0%		2.8%	69.4%	27.8%
Age							
Under 30	28	20	8	28	-	18	10
	100.0%	51.3%	20.0%			64.3%	35.7%
30 to 50	27	11	16	24	2	19	3
	100.0%	28.2%	40.0%		8.3%	79.2%	12.5%
Over 50	23	7	15	19	-	13	6
	100.0%	17.9%	37.5%			68.4%	31.6%

Mixing row and column percents in a single row is accomplished by using -997 to indicate a column that should have column percents instead of row percents. Since a separate row for column percents is the default you must also request ROW PERCENTS. Thus the following subcommand:

```
ROW PERCENTS, COMPUTE -997 -997 1 1 1 2 2 2
```

Creates an output line which contains column percents for the first 2 columns, row percents based on the first column for the next 3 columns and row percents based on column 2 for the final 3 columns.

```
C C R R R R R R  (C=col pct, R=row pct)
```

The first two banner points will have column percents while the rest have row percents based on columns 1 and 2.

The statement ROW PERCENTS BASED ON GOOD.N in this situation causes the two column percents to be based on the good rather than the total N and has no effect on row percents based on a specific column. Figure 8.5 illustrates the use of both -999 and -997.

If both ROW and COLUMN percents are requested with the mixed row COMPUTE used in Figure 8.5, there will be two rows with percents. One row is identical to the percent row in Figure 8.5. The other row contains column percents for the Income.groups variable. In this situation the use of -998 instead of -997 instructs the SURVEY command to place all of the column percents together. Figure 8.6 illustrates the output for the first row

of the table under the permutations of PERCENT and COMPUTE. The printout of the first two examples produces confusing results because they have one line with mixed row and column percents. The second two examples have the column percents all on the same line. This is a more sensible arrangement.

Figure 8.6 Mixing Row and Column Percents

NOTE: COLUMN PERCENTS ARE IN A BOLD FONT

ROW PERCENTS, COMPUTE -997 -997 -999 3 3 3,

30 to 50	27	11	16	24	2	19	3
	100.0%	28.2%	40.0%		8.3%	79.2%	12.5%

COLUMN PERCENTS, COMPUTE -997 -997 -999 3 3 3,

30 to 50	27	11	16	24	2	19	3
	33.8%				**100.0%**	**38.0%**	**15.0%**

ROW COLUMN PERCENTS, COMPUTE -997 -997 -999 3 3 3,

30 to 50	27	11	16	24	2	19	3
	100.0%	28.2%	40.0%		8.3%	79.2%	12.5%
	33.8%				100.0%	38.0%	15.0%

COLUMN ROW PERCENTS, COMPUTE -997 -997 -999 3 3 3,

30 to 50	27	11	16	24	2	19	3
	33.8%				**100.0%**	**38.0%**	**15.0%**
	100.0%	**28.2%**	**40.0%**		8.3%	79.2%	12.5%

ROW COLUMN PERCENTS, COMPUTE -998 -998 -999 3 3 3,

30 to 50	27	11	16	24	2	19	3
	100.0%				8.3%	79.2%	12.5%
	33.8%	**28.2%**	**40.0%**		**100.0%**	**38.0%**	**15.0%**

COLUMN ROW PERCENTS, COMPUTE -998 -998 -999 3 3 3,

30 to 50	27	11	16	24	2	19	3
	33.8%	**28.2%**	**40.0%**		**100.0%**	**38.0%**	**15.0%**
	100.0%				8.3%	79.2%	12.5%

The examples in Figures 8.5 and 8.6 show the use of negative numbers, which cannot be positions in the table, as special controls.

1. -999 indicates that the corresponding position in the banner columns should never have a percent printed.
2. -998 is used when there are multiple output lines to indicate a banner point that is to have column percents printed not in a mixed percentage line but in a column percent line only.
3. -997 indicates a column percent in a mixed percentage line situation.

There is one further permutation to handle the situation where some of the column percents are suppressed and the row percents are based either on the row totals or on one of the banner columns. Usually the column number indicates the base for the percents. If both row and column percents are requested and the base is a negative number, the absolute value of that number is used to indicate the base and the column percents are suppressed.

Since the number 0 is used to indicate the row totals and negative zero is not allowed, -996 is used to indicate a column that is to have the row totals as a base and column totals suppressed. Figure 8.7 shows the command and the resulting output.

Figure 8.7 Suppressing the Column Percents

```
SURVEY paceset [ GEN good.n;
                 IF income.groups GOOD, SET good.n=1;],
        LABELS paceset.lab;
      BAN sex good.n  income.groups,
      STUB age,
      LAYOUT LABELS TOTALS QUESTION BODY,
      ROW COLUMN PERCENTS, COMPUTE 0 0 -996 -3 -3 -3;
```

		==== Sex ====			==== Income of respondent ====		
	Total Sample	Male	Female	good.n	Under $30, 000	$30, 000 to $60, 000	Over $ 60,000
Total	80	39	40	72	2	50	20
Sample	100.0%	48.8%	50.0%	90.0%	2.8%	69.4%	27.8%
	100.0%	100.0%	100.0%				
Age							
Under 30	28	20	8	28	-	18	10
	100.0%	71.4%	28.6%	100.0%		64.3%	35.7%
	35.0%	51.3%	20.0%				
30 to 50	27	11	16	24	2	19	3
	100.0%	40.7%	59.3%	88.9%	8.3%	79.2%	12.5%
	33.8%	28.2%	40.0%				
Over 50	23	7	15	19	-	13	6
	100.0%	30.4%	65.2%	82.6%		68.4%	31.6%
	28.7%	17.9%	37.5%				

8.6 F and t Tests

Several statistics that test the differences between the means of the columns of a given banner variable, the independence of the stub and banner variables, and the sample-to-sample fluctuation of the means are optional subcommands in SURVEY. These statistics are printed beneath full tables, as part of the summary, or beneath

the summary unless a LAYOUT command is used with F.CHI included to indicate an alternate location for the results of these tests and the chi-square test.

Differences between the means of the columns of a banner variable are tested for significance using an F test or a t test (when there are just two columns). Including the F.TEST subcommand in the table definition requests these tests. A t value, which is the square root of the F value, is computed in addition to the F value when just two means are being compared. Probabilities of .05 and below are generally accepted as indicating significant differences — differences that are not just the effects of random error.

Figure 8.8 shows F and t tests. The GROUP.STUB variables "Eff" and "Ser" are continuous measurements, so the body of the table is not useful, but the summary statistics can be produced without the body. The means of the variable Eff (efficiency) in the processing lab and treatment groups, for example, differ significantly. You can conclude that mean efficiency is higher in the long baking treatment, and that it differs among the three labs. It is not possible to conclude where the differences lie among the labs without additional pair-wise comparisons. It may be that all three means are significantly different, that the mean in SW's lab differs from the other two but that they (TG's and BK's lab means) do not differ from each other, and so on.

Figure 8.8 Summary Statistics, F and t Tests

```
SURVEY Cell,   LABELS 'Cell.lab';

 BANNER Proc Treat,  GROUP.STUBS Eff Ser,   PLACES MEANS 3,
 SUMMARY CONTAINS MEANS S.D  S.E,   F.TEST,
 CELL.WIDTH 9, MARGIN 14,
 LAYOUT LABELS QUESTION SUMMARY;
```

						=== Bake	===
		===== Processing Lab =====				=== Treatment	===
	Total Sample		TG	SW	BK	Short	Long
Eff							
Mean	4.921		5.470	4.066	5.276	4.175	5.621
S.D.	1.103		0.490	1.413	0.496	1.151	0.326
Standard error	0.138		0.110	0.301	0.106	0.207	0.057
F value			14.7035			47.9747	
Probability			0.0000			0.0000	
t value						6.9264	
Ser							
Mean	18.764		13.959	27.103	14.794	24.024	13.823
S.D.	12.466		4.978	17.471	5.213	15.823	4.387
Standard error	1.558		1.113	3.725	1.112	2.842	0.764
F value			9.5711			12.6898	
Probability			0.0002			0.0007	
t value						3.5623	

AGAIN, **F.TEST PRINT PROB .01**;

		===== Processing Lab =====			=== Bake === Treatment ===	
	Total Sample	TG	SW	BK	Short	Long
Eff						
Mean	4.921	5.470	4.066	5.276	4.175	5.621
S.D.	1.103	0.490	1.413	0.496	1.151	0.326
Standard error	0.138	0.110	0.301	0.106	0.207	0.057
Probability F		0.0000			0.0000	
Ser						
Mean	18.764	13.959	27.103	14.794	24.024	13.823
S.D.	12.466	4.978	17.471	5.213	15.823	4.387
Standard error	1.558	1.113	3.725	1.112	2.842	0.764
Probability F		0.0002			0.0007	

AGAIN, **F.TEST PRINT PROB .0001 ;**

		===== Processing Lab =====			=== Bake === Treatment ===	
	Total Sample	TG	SW	BK	Short	Long
Eff						
Mean	4.921	5.470	4.066	5.276	4.175	5.621
S.D.	1.103	0.490	1.413	0.496	1.151	0.326
Standard error	0.138	0.110	0.301	0.106	0.207	0.057
Probability F		0.0000			0.0000	
Ser						
Mean	18.764	13.959	27.103	14.794	24.024	13.823
S.D.	12.466	4.978	17.471	5.213	15.823	4.387
Standard error	1.558	1.113	3.725	1.112	2.842	0.764

(Note the lack of any F statistic information here)

Figure 8.8 shows the subcommands with variations. The first variation:

```
F.TEST PRINT PROBABILITY,
```

requests that the F.test be done but that the report include only the probability, omitting the F and t values. The second variation:

```
F.TEST PRINT PROBABILITY .0001,
```

requests that the F test probabilities be printed only if they are greater than .0001. A more usual request would provide levels such as .01 or .05.

```
F.TEST .01,
```

requests the full F test printout for any probabilities that are .01 or less.

8.7 Chi-Square

The *independence* of nominal (unordered categorical) banner and stub variables is assessed using the chi-square statistic. If the two variables are not associated, comparable numbers of cases (based on the marginals) are expected in the cells of the table. The expected and observed cell frequencies are compared in computing chi square. A probability of .05 or lower implies that the two variables are not independent.

Figure 8.9 Chi-Square

```
SURVEY Paceset,    LABELS 'Paceset.lab' ;
BANNER Income.group,  STUB Age,  NO MISSING, SKIP 0, CHI;

                      ==== Income of  ====
                      ==== respondent ====

                                 $30,
                          Under 000 to
                  Total   $30,   $60, Over $
                 Sample    000    000 60,000

 Total             80      2     50     20
 Sample         100.0% 100.0% 100.0% 100.0%

 Age

 Under 30          28      -     18     10
                  35.0%         36.0%  50.0%
 30 to 50          27      2     19      3
                  33.8% 100.0%  38.0%  15.0%
 Over 50           23      -     13      6
                  28.7%         26.0%  30.0%

 Base              78      2     50     19
                  97.5% 100.0% 100.0%  95.0%
 Mean             1.94   2.00   1.90   1.79
 S.D.             0.81   0.00   0.79   0.92

 Chi Square             7.1653 *
 DF for Chi             4.
 Probability            0.1274
 * Some cells had an expected value of less than 5.
```

Chi square is requested by the CHI (or CHI.SQUARE) subcommand. When the table is a two-by-two table and any expected cell frequency is less than 5, Yates's correction for continuity is used in the chi square computation. When the survey is a two-by-two table and the total count is less than or equal to 50, an exact significance level is calculated. This probability is labeled "Fisher 2-tail" in the output. Chi square also tests the *difference between proportions* in a two-by-two table. The chi-square value is equal to the square of the corresponding normal score.

Sometimes it is interesting to find out which cells are contributing the most to the total chi-square value. This can be done by requesting CELL.CHI as part of the body of the table. When there are cells with a small number of cases, the value of the statistic is lessened. Figure 8.9 shows a table with the chi-square request. Notice the warning message at the bottom of the table. This message is there to warn you that the significance of the chi-square value is questionable. The subcommand EXPECTED.VALUE can be used to locate the cells which have the low expected values.

The calculation and printing of the chi-square can be controlled in the same way that the F-test is controlled.

```
CHI .05,
CHI PRINT PROBABILITY,
CHI PRINT PROBABILITY .01,
```

PRINT PROBABILITY causes the chi-square value and the degrees of freedom that are usually printed to be omitted. The final number, if it is provided, provides a threshold for testing. If the probability is greater than the supplied value, the chi-square statistic is not displayed.

Figure 8.10 Cramer's V

```
SURVEY Paceset,   LABELS 'Paceset.lab';
     BANNER Income.group, STUB Age, NO MISSING,
     SUMMARY CONTAINS MEAN S.D, SKIP 0, CRAMER.V;
```

		==== Income of ====		
		==== respondent ====		
	Total Sample	Under $30, 000	$30, 000 to $60, 000	Over $ 60,000
Total Sample	80 100.0%	2 100.0%	50 100.0%	20 100.0%
Age				
Under 30	28 35.0%	-	18 36.0%	10 50.0%
30 to 50	27 33.8%	2 100.0%	19 38.0%	3 15.0%
Over 50	23 28.7%	-	13 26.0%	6 30.0%
Mean	1.94	2.00	1.90	1.79
S.D.	0.81	0.00	0.79	0.92
Cramer's V		0.1588		

Cramer's V is similar to chi-square. However, it is not affected by sample size. Since it is not dependent on the sample size it is useful when comparing results from tables which are based on different sample sizes. Figure 8.10 shows both the command and the output. If CHI is used, Cramer's V prints just beneath it. If there are controls on the desired probability for chi-square, those controls also apply to Cramer's V.

8.8 Standard Error and Standard Deviation

The *variability of the means* with repeated sampling is measured by the standard error. This is requested by including S.E (or STANDARD.ERROR) in the SUMMARY subcommand. Figure 8.8 also contains the standard error of the means. You can see that the variability of the mean in SW's lab is generally greater than those of the other two labs.

Figure 8.11 Standard Deviation for a Sample

```
SURVEY Paceset,
    LABELS 'Paceset.lab';
  BANNER Age Own,  STUB Income.group,
  LAYOUT LABELS QUESTION SUMMARY,
  PLACES MEANS 3, CELL.WIDTH 8,
  SUMMARY CONTAINS MEAN STANDARD.DEV VARIANCE STANDARD.ERROR;
 $
```

```
                                                 == Respondent =
                          ========= Age ========= == ownership  =

                    Total   Under   30 to
                   Sample      30      50 Over 50      no     yes

Income of respondent

Mean                2.250   2.357   2.042   2.316   1.900   2.306
S.D.                0.496   0.488   0.464   0.478   0.316   0.499
Variance            0.246   0.238   0.216   0.228   0.100   0.249
Standard error      0.059   0.092   0.095   0.110   0.100   0.063

 AGAIN, SAMPLE;

                                                 == Respondent =
                          ========= Age ========= == ownership  =

                    Total   Under   30 to
                   Sample      30      50 Over 50      no     yes

Income of respondent

Mean                2.250   2.357   2.042   2.316   1.900   2.306
S.D.                0.493   0.479   0.455   0.465   0.300   0.495
Variance            0.243   0.230   0.207   0.216   0.090   0.245
Standard error      0.058   0.091   0.093   0.107   0.095   0.063
```

The formulas for the standard deviation, variance and standard error assume that these statistics are to be computed as estimates of how well the sample approximates the universe from which it was drawn. The formulas for

the population statistics and sample statistic are slightly different to account for the fact that "a small sample tends to underestimate the variance of the parent population" ("Facts From Figures", M.J. Moroney, Penguin Books). If the statistics that are desired are the statistics for the sample itself, the subcommand SAMPLE can be used. SAMPLE is used by itself without any arguments.

```
SUMMARY CONTAINS STANDARD.DEV STANDARD.ERROR,  SAMPLE,
```

Figure 8.11 illustrates both the subcommands and the results when the standard deviation, variance and standard error are computed using the two different methods. As you can see the differences are small. As the sample sizes increase the differences virtually disappear. Even with our sample of 80 people it is necessary to print the statistics to three decimal places to see the differences clearly.

These differences, although small, illustrate that it is necessary for anyone using even basic statistics to have an awareness both of the data and of the methods that are used in processing them. Two programs supposedly producing the same statistic can have different results. In addition to differences caused by variations in the formulas, differences can occur because of computer hardware. Strange as it may seem, because the PC conforms to IEEE standards, its results may be slightly more accurate than those from a large mainframe computer.

Occasionally the known population is very small and the sample is a very large part of that population. For example, if you are conducting a survey of all the brew pubs in the Eastern United States and there are only 350 of them, you can provide that total by using the POPULATION identifier in the SURVEY command. With this information, an adjustment is made to the standard deviation, variance and standard error to account for the fact that this is a small finite population.

Figure 8.12 shows the command, the subcommands and the statistics that are produced when we claim a total population of 200 for our sample of 80 people in file Paceset. Obviously this should only be used when the study is about a small population and the population totals are known. When the sample is the entire population you will notice that the standard deviation, variance and standard error for the total sample becomes zero. When you have included everyone there is no further need for statistics.

Figure 8.12 Finite Population Statistics

```
SURVEY Paceset,
  LABELS 'Paceset.lab',
     POPULATION 200;

  BANNER Age Own,  STUB Income.group,
  LAYOUT LABELS QUESTION SUMMARY, PLACES MEANS 3,
  SUMMARY CONTAINS MEANS S.D VARIANCE STANDARD.ERROR;
```

		======== Age =======			Respondent ownership	
	Total Sample	Under 30	30 to 50	Over 50	no	yes
Income of respondent						
Mean	2.250	2.357	2.042	2.316	1.900	2.306
S.D.	0.397	0.453	0.436	0.454	0.308	0.414
Variance	0.158	0.205	0.190	0.206	0.095	0.172
Standard error	0.047	0.086	0.089	0.104	0.097	0.053

8.9 MEDIANS

Medians may be calculated and printed in the summary section of tables. They may be requested for the banner variables and, if the medians are on a variable other than the stub variable, they may be included in the body of the table. The subcommands which govern the medians are very similar to those which govern means and permit a list of median variables corresponding to the list of stub variables. However there are a few restrictions on medians of external variables. This is because the median can only be calculated when all of the data values are available. Means, on the other hand, can be computed from a sum and a frequency.

When medians are based on an extra variable, the information for subtotals and combines is not available. Because medians of an extra variable require all the data, they are computed as the data are read. Subtotals and combines are not calculated until all the data are accumulated and ready to print. Therefore, subtotals and combines have medians which are an estimate. This estimate is the mean of the medians of those rows which comprise the subtotal or the dynamically combined row. Medians are *not* available for nets or when the BY identifier is used.

Figure 8.13 The Medians on an External Variable

```
SURVEY PACESET, LABELS 'Paceset.lab';

    BANNER Age Own, STUB Income.Groups,
    MEDIAN Income,
    SUMMARY CONTAINS BASE MEANS MEDIAN,
    INCLUDE MISSING MEANS VALUES;
```

		Age			Respondent ownership	
	Total Sample	Under 30	30 to 50	Over 50	no	yes
Total Sample	80	28	27	23	11	69
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Income of respondent						
Under $30,000	2	-	2	-	1	1
	2.5%		7.4%		9.1%	1.4%
$30,000 to $60,000	50	18	19	13	9	41
	62.5%	64.3%	70.4%	56.5%	81.8%	59.4%
Over $60,000	20	10	3	6	-	20
	25.0%	35.7%	11.1%	26.1%		29.0%
Missing	8	-	3	4	1	7
	10.0%		11.1%	17.4%	9.1%	10.1%
Base	72	28	24	19	10	62
	90.0%	100.0%	88.9%	82.6%	90.9%	89.9%
Mean Income	34398	37388	29311	35580	26785	35626
Median	32510	34550	25950	32520	26525	34200

Figure 8.13 illustrates medians of an external variable. If the SUMMARY subcommand is not used, the medians will automatically be added to that section. The use of the SUMMARY subcommand specifies exactly what should be included (or omitted) and the order of the summary contents. If the SUMMARY subcommand does not include MEDIAN, the medians will not appear there even if the MEDIAN subcommand has been used.

In Figure 8.13 the use of "MEANS Income" instead of "MEDIANS Income" produces exactly the same results. This is because the mean and the median in a given table are always done on the same variable. "INCLUDE MISSING MEANS VALUES" is used in Figure 8.13 so that all the cases in the file are represented even though variable Income has some missing data. The MISSING subcommand, described in the chapter "SURVEY: Special Features for the Layout Sections", can be used to separate the cases with missing values on the stub variable from those which are missing only on the means variable.

Figure 8.14 illustrates the same command with the addition of a request for counts and medians in the cells of the table. In this figure, a total of 21 different medians, out of a possible 24, are computed. (There are 3 cells with no data.) This means that 21 different arrays of numbers must be sorted to get the exact middle of each array. While there is no limit on the number of medians that can be requested, getting medians on the hundreds of cells that are often tabulated will almost certainly require extra passes through the data file.

Figure 8.14 Medians in the Body of the TABLE

```
AGAIN, BODY CONTAINS COUNTS MEDIAN,
PLACES MEANS 0,
PLACES MEDIAN 0;
```

		Age			Respondent ownership	
	Total Sample	Under 30	30 to 50	Over 50	no	yes
Total Sample	80	28	27	23	11	69
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Income of respondent						
Under $30,000	2	-	2	-	1	1
	18400		18400		18000	18800
$30,000 to $60,000	50	18	19	13	9	41
	30085	31350	25900	31800	26700	30850
Over $60,000	20	10	3	6	-	20
	45850	46950	42500	46050		45850
Missing	8	-	3	4	1	7
	10.0%		11.1%	17.4%	9.1%	10.1%
Base	72	28	24	19	10	62
	90.0%	100.0%	88.9%	82.6%	90.9%	89.9%
Mean Income	34398	37388	29311	35580	26785	35626
Median	32510	34550	25950	32520	26525	34200

Figure 8.15 illustrates the use of medians in all possible situations; in the summary, in the body of the table, in subtotals and as columns describing the banner variables. Medians of the banner variables are requested by following the variable name with the desired statistics enclosed in parentheses. This is described in detail in the chapter "SURVEY: More About the Columns".

```
Varname ( MED )  Varname ( MED )
```

The medians for any subtotals are not an exact. They are estimated by using the average of the rows that comprise the subtotal. The advantage of the median over the mean is that it removes the effect of outliers. One case with an enormous value on the median variable cannot distort the value of the median as it can the mean. The effect of the outliers on the subtotal has already been removed by computing the exact median for each row. The estimated median for the subtotals may not be exact but, since the effect of the outliers has already been removed, it will be very close.

When the MEDIAN subcommand is used without a variable name as an argument, it produces a median based on the values of the individual stub variables. The stub variable may be numeric or character, but the categories defined by this variable should be ordered for a meaningful result. (Character variables are mapped alphabetically by the SURVEY and MAP commands — thus, the order is generally happenstance.)

Figure 8.15 Medians everywhere

```
AGAIN, DEFINE '$60,000 and under' Income.groups 1 2,
BANNER AGE ( MED M );
```

		Age				
	Total Sample	Under 30	30 to 50	Over 50	Median Income	Mean Income
Total Sample	80	28	27	23	32500	34174
	100.0%	100.0%	100.0%	100.0%		
Income of respondent						
Under $30,000	2	-	2	-	18400	18400
	18400		18400			
$30,000 to $60,000	50	18	19	13	30085	30007
	30085	31350	25900	31800		
Over $60,000	20	10	3	6	45800	46800
	45850	46950	42500	46050		
$60,000 and under	52	18	21	13	-	29561
----------	29636	31350	25186	31800		
Missing	8	-	3	4	-	-
	10.0%		11.1%	17.4%		
Base	72	28	24	19	-	34174
	90.0%	100.0%	88.9%	82.6%		
Mean Income	34398	37388	29311	35580		
Median	32510	34550	25950	32520		

Figure 8.16 Medians: Using METHOD.GROUPED

```
SURVEY Paceset,  LABELS 'Paceset.lab';
  BANNER Age Own,  STUB Income.Group,   NO MISSING, NO SKIP,
  SUMMARY MEAN MEDIAN,  METHOD.DEFAULT ;
 $
```

		Age			Respondent ownership	
	Total Sample	Under 30	30 to 50	Over 50	no	yes
Total Sample	80	28	27	23	11	69
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Income of respondent						
Under $30,000	2	-	2	-	1	1
	2.5%		7.4%		9.1%	1.4%
$30,000 to $60,000	50	18	19	13	9	41
	62.5%	64.3%	70.4%	56.5%	81.8%	59.4%
Over $60,000	20	10	3	6	-	20
	25.0%	35.7%	11.1%	26.1%		29.0%
Mean	2.25	2.36	2.04	2.32	1.90	2.31
Median	2.00	2.00	2.00	2.00	2.00	2.00

```
  AGAIN,  METHOD.GROUPED ;
```

		Age			Respondent ownership	
	Total Sample	Under 30	30 to 50	Over 50	no	yes
Total Sample	80	28	27	23	11	69
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%
Income of respondent						
Under $20,000	2	-	2	-	1	1
	2.5%		7.4%		9.1%	1.4%
$20,000 to $40,000	50	18	19	13	9	41
	62.5%	64.3%	70.4%	56.5%	81.8%	59.4%
Over $40,000	20	10	3	6	-	20
	25.0%	35.7%	11.1%	26.1%		29.0%
Mean	2.25	2.36	2.04	2.32	1.90	2.31
Median	2.68	2.78	2.53	2.73	2.44	2.73

A strict definition of the median is that case which is in the middle of the distribution. If there are an even number of cases, the simple solution is to take the value halfway between the two middle values. This is the method that is assumed. It can be specifically requested by using the subcommand "METHOD.DEFAULT". However, if there are fractional weights, the first value of the two values that define the median is used instead. This method can be specifically requested by using the subcommand "METHOD.FIRST". METHOD.FIRST is used when there are fractional weights instead of METHOD.DEFAULT.

When either METHOD.DEFAULT or METHOD.FIRST is used, it is difficult to estimate the real median when the data are grouped. In Figure 8.16, medians are requested for the stub variable Income.groups. The most one can say is that the median of 2 is approximately $30,000 for the overall total and for all the columns. METHOD.GROUPED assumes that this is a truly continuous variable distributed evenly across the range. A resulting value of 2.68 can be used to estimate that the median is 68 percent of the way between 20,000 and 40,000 or approximately 33,600. This estimate is closer than the estimate when METHOD.DEFAULT is used. METHOD.GROUPED can be used even when there are fractional weights.

Figure 8.17 Medians in a Multiple Response Table

```
SURVEY Paceset,  LABELS 'Paceset.lab';
  BANNER Num.TV ( MED ), MR.STUB Phone to Ereader, MEDIAN Income;
$
                        ====== Number of television sets in your home =====

                    Total                                  5 or            Median
                   Sample      1      2      3      4     more      6      Income

Total Sample          72      6     21     12     26      6      1   32510.00
                    100.0% 100.0% 100.0% 100.0% 100.0% 100.0% 100.0%

Which of the following do you own?

Smart phone as        39      2     12      8     14      2      1   36700.00
primary              54.2%  33.3%  57.1%  66.7%  53.8%  33.3% 100.0%
telephone

One or more           27      2      7      2     13      2      1   35100.00
landline based       37.5%  33.3%  33.3%  16.7%  50.0%  33.3% 100.0%
telephones

One or more           29      3      9      4     11      1      1   35700.00
tablet computers     40.3%  50.0%  42.9%  33.3%  42.3%  16.7% 100.0%

One or more           35      5     12      5     12      -      1   33800.00
Ereaders             48.6%  83.3%  57.1%  41.7%  46.2%         100.0%

Missing               10      1      3      3      2      1      -         -
                     13.9%  16.7%  14.3%  25.0%   7.7%  16.7%

Base                  62      5     18      9     24      5      1
                     86.1%  83.3%  85.7%  75.0%  92.3%  83.3% 100.0%
Responses            130     12     40     19     50      5      4         -
Median Income      34200  33800  37350  37000  32510  37000  48000
```

8.10 Summary Statistics in Multiple Response Tables

Medians on an external variable can be used in multiple response tables. Each respondent is only counted once in any median in the summary section. Figure 8.17 illustrates this with a multiple response stub variable. The medians are all based on the 62 respondents who had at least 1 good reply to the multiple response stub questions and also had a non-missing reply on variable Income. However, in a multiple response situation, particularly if both the banner variable and the stubs are multiple response, a single respondent may be represented more than once.

In Figure 8.17, there is no problem in interpretation. While a case may be represented in all 4 rows in the body of the table, it is only represented once in any given row. The median income of people owning an answering machine was $33,800. The median income of those people with 3 television sets was $37,000.

Figure 8.18 Quartiles in the SUMMARY Section

```
.SURVEY Paceset,   LABELS 'Paceset.lab';

    BAN Age (MED),  STUB SEX,
    MEDIAN INCOME,  QUARTILES,

    BODY CONTAINS COUNTS MEDIAN,
    SUMMARY CONTAINS MEANS MEDIAN,
    PLACES MEDIAN 0, PLACES MEANS 0,

    COMMAS,  NO MISSING,
    CELL.WIDTH 11,  MARGIN 14;
 $
```

	Age				
	Total Sample	Under 30	30 to 50	Over 50	Median Income
Total Sample	72	28	24	19	32,500
	100.0%	100.0%	100.0%	100.0%	
Sex					
Male	31	20	8	3	34,700
	34,700	38,700	30,800	32,500	
Female	40	8	16	15	30,800
	30,900	31,350	25,600	34,400	
Mean Income	34,435	37,388	29,311	35,790	
First quartile	26,700	30,812	23,720	28,650	
Median	32,520	34,550	25,950	33,460	
Third quartile	42,000	45,600	37,750	42,800	

When the statistic (mean, median, etc.) is of an external variable, each respondent is counted but once and the results can be easily interpreted. When there is no external variable statistics, other than the base and the number of responses depend on the type of multiple response stub variable. When the multiple response stub variable is

0/1 (dummy variable) the mean is the number of responses divided by the good.n. This is the average number of responses. When the mean is 1-n coded, the mean is the sums of the responses divided by the number of responses or the value of the average response.

8.11 Other Percentiles

Other percentiles can be computed. The subcommand QUARTILES is available to produce the .25, .5, and .75 percentiles in the summary section only. PLACES MEDIAN applies to all three quartiles. If the SUMMARY section contains the median, the use of QUARTILES or of the GET.PERCENTILES subcommand (described below) controls what is actually included in the summary section.

The GET.PERCENTILES subcommand can be used to request any single percentile.

```
SUMMARY CONTAINS MEDIAN, GET.PERCENTILE .1,
```

Figure 8.19 illustrates the GET.PERCENTILES SUBCOMMAND. Only a single percentile can be computed in a single group of TABLES. To produce many different percentiles with a single command, use either the PERCENTILES command or the COUNT command.

Figure 8.19 Getting Individual Percentiles in the SURVEY Command

```
.SURVEY Paceset,   LABELS 'Paceset.lab';

    BAN Age (MED),  STUB SEX,
    MEDIAN INCOME,  GET.PERCENTILE .9,

    BODY CONTAINS COUNTS MEDIAN,
    SUMMARY CONTAINS MEANS MEDIAN,
    PLACES MEDIAN 0, PLACES MEANS 0,

    COMMAS,  NO MISSING,
    CELL.WIDTH 11,  MARGIN 14;
```

	Age				
	Total Sample	Under 30	30 to 50	Over 50	.90 Percentile Income
Total Sample	72	28	24	19	47,780
	100.0%	100.0%	100.0%	100.0%	
Sex					
Male	31	20	8	3	51,920
	51,920	54,180	38,200	34,700	
Female	40	8	16	15	45,600
	46,770	37,000	43,430	50,260	
Mean Income	34,435	37,388	29,311	35,790	
.90 Percentile	48,560	52,420	42,400	49,090	

8.12 Limitations

When medians are requested for variables other than the stub variables, a separate pass is made through the input data file. When all the medians are based on the stub variables, the medians are calculated at print time from the body of the table itself. This causes a problem if the stub variables are fractional. The stub value is truncated to obtain the row position for the variable and there is no record of the original values. Figure 8.20 illustrates this problem. A second problem occurs when the stub variable has a range that is so large that it exhausts the work-space available for the table cells.

However, if the body is omitted from the printout, either with a LAYOUT subcommand or the NO BODY subcommand, and the contents of the body are not needed to fulfill other requests, the extra pass through the data file is made and the medians are computed correctly even with fractional stub values. This permits medians for many stub variables with large ranges or fractional values to be done in a single GROUP.STUBS command.

The following subcommands require the body information even though the body does not print. Use of these features with sparse data can cause an out of memory message. With fractional stub variables, the medians will be based on the truncated values rather than the original fractional values.

```
CHI
SUMMARY CONTAINS with any of LOW    HIGH   RANGE or MODE
SUBTOTALS or NETS
OUT Psfile
```

Figure 8.20 Medians and Fractional Stub Values

```
File Work contains:

    VAR1     VAR2
       1     0.920
       2     1.630
       3     2.115
```

```
SURVEY Work;
STUB VAR1,  MEDIAN VAR2,
NO MISSING, NO SKIP,
SUMMARY CONTAINS MEAN MEDIAN;
```

```
Total              3
Sample         100.0%

VAR1

       1           1
                3.33%
       2           1
                3.33%
       3           1
                3.33%

Mean VAR2       1.55
Median          1.63
```

```
SURVEY Work;
STUB VAR2,
NO MISSING, NO SKIP,
SUMMARY CONTAINS MEAN MEDIAN;
```

```
Total              3
Sample         100.0%

VAR2

       0           1
                3.33%
       1           1
                3.33%
       2           1
                3.33%

Mean            1.55
Median          1.00
```

8.13 Means on External Variables

Means on a variable other than the stub variable were covered in the first chapter of this manual. The MEANS subcommand can be used with a list of variable names which correspond to a list of stub variables. The banners, layout and other details for the tables must be the same.

```
BANNER Age Gender Education,
STUB Item1 to Item10,
MEANS Avg1 to Avg5 Count1 to Count5;
```

The first 5 tables with stub variables Item1 to Item5 will have means calculated using the variables Avg1 to Avg5. The last 5 tables will have the variable Count1 to Count5 paired with the stub variables Item6 to Item10;

When you are doing a large batch run with many tables and some of them have means on an external variable while others have means on the stub variable, it is necessary to have some way of reverting to the means variable. The problem with:

```
BANNER Age Gender Education,
STUB Q33, MEANS Income;
STUB Q22, MEANS Q22,
STUB Q23;
```

is that the second table has means on Q22, the stub variable but so does the table for Q23. There are two ways to reset the MEANS variable;

```
BANNER Age Gender Education,
STUB Q33, MEANS Income;
NO MEANS;
MEANS;
STUB Q22,
STUB Q23;
```

or

```
BANNER Age Gender Education,
STUB Q33, MEANS Income;
STUB Q22,MEANS DUMMY,
STUB Q23;
```

8.14 Survey.labels for the Statistics

Special values for the various strings used to label the statistics can be provided using the special variable name “Survey.labels”.

```
(41)  F value          (42)  t value          (43)  Probability
(44)  Chi Square       (45)  DF for Chi       (46)  Fisher 2-tail
(47)  Yates chi-sq....  (48)  Some cells ....  (49)  Cramer’s V
(301) Label for significance tests of means
(302) Label for test of proportions
(303) Label for test of independence
(304) Label when combined values are used in the tests
```

the next five values are used only when REPORT ALL RESULTS is used and none of the tests have been significant.

```
(310) used with TEST ALL RESULTS and CHI
(311) used with TEST ALL RESULTS and F
(312) used with TEST ALL RESULTS and TEST PROP
(313) used with TEST ALL RESULTS and TEST INDEPENDENCE
(314) used with TEST ALL RESULTS and TEST MEANS
```

SUMMARY

SURVEY

```
 SURVEY Paceset,  LABELS 'Paceset.lab';
        BANNER Sex Sex WITHIN Own,   STUB Age,
        NO MISSING, NO SUMMARY,
        ROW PERCENTS, COMPUTE 0 0 1 2 1 2;
$

SURVEY Cell,  LABELS 'Cell.lab';
       BANNER Proc Treat,  GROUP.STUBS Eff Ser,
       PLACES MEANS 3,
       SUMMARY CONTAINS MEANS S.D S.E,   F.TEST,
       CELL.WIDTH 9, MARGIN 14,  LAYOUT LABELS QUESTION SUMMARY;
$

SURVEY Paceset,  LABELS 'Paceset.lab';
       BANNER Age Own,  STUB Income.Group,
       MEDIAN Income,
       SUMMARY MEAN MEDIAN,  METHOD.DEFAULT ;
$
```

Optional Identifiers:

POPULATION nn

provides the size of the population from which the sample is taken. This number is used to provide the necessary correction for finite sample statistics. It affects the standard deviation, the variance and the standard error. The number is usually smaller than 500 because as the population gets even that large the finite population correction makes very little difference.

Optional Subcommands:

CHARACTER TEST 'c' 'c'

provides values to be used when significance tests are done against the row totals and MARK ALL TESTS is in effect. The two characters replace the + and - which are the default values.

CHI

requests that a chi-square test be done on each stub/banner variable pair. Usually the printout contains the chi-square, the probability, and the degrees of freedom. The printout can be limited to just the probability and to just the instances where the chi-square is significant at a specified level.

```
CHI PRINT PROB,
CHI .05,
CHI PRINT PROB .05,
```

COMPUTE nn nn nn nn

specifies the columns to be used as the basis for row percents when the row totals are not the desired base. The numbers refer to the actual location of the column in the banner with the row totals considered to be column 0. Negative column numbers can be used to further control which row and column percents are actually displayed.

```
-999    requests no percents for that column
-998    use when both row and column percents are desired for the
        columns that are to have only column percents
-997    requests column percents rather than row percents for
        the column
-1 to -995 used when both row an column percents are desired but
        the specified column is to have only row percents based
        on the column pointed to by the absolute value of the
        number
-996    is used in place of a negative zero to indicate a column
        whose row percent is to be based on the row totals and
        whose column percent is to be ignored.
```

F.TEST

requests that an F-test be done on each stub/banner variable pair. Usually the printout contains the F value, the probability, and if the banner variable has only two columns, the t value. The printout can be limited to just the probability and to just the instances where the F value is significant at a specified level.

```
F.TEST .05,
F.TEST PRINT PROB,
F.TEST PRINT PROB .05,
```

GET.PERCENTILES nn

requests that a percentile other than the median be used. The placement and print places are controlled by the median subcommands. Only 1 percentile can be produced in a single table with the exception of QUARTILES in the summary which are controlled by the QUARTILES subcommand.

MARK ALL TESTS

requests that all columns that are involved in significant differences be labelled, not just the larger test value.

MEANS vn vn vn
DUMMY

MEANS can be followed by a list of variables which match up to a list of stub variables. It can also be followed by DUMMY which resets the MEANS variable to the current stub variable.

MEDIAN

requests that medians be computed and placed in the summary section of the table. If SUMMARY is explicitly used, it must, however, include MEDIAN.

MEDIAN vn vn vn

requests that medians be computed on some variable other than the stub variable. If a list of variables is supplied it must correspond to the stub variable list. When means and medians are both requested, they must be the same in any single table. Medians can also be included in the body of the table by using the BODY CONTAINS subcommand.

METHOD.DEFAULT

requests that median for the stub and banner variables be calculated using the middle value or if there are an even number of values, the number half way between the two middle values.

METHOD.FIRST

requests that the program use the first of the values as the median when there is an even number of cases. METHOD.FIRST is used instead of METHOD.DEFAULT when there are fractional weights.

METHOD.GROUPED

assumes that this is a truly continuous variable distributed evenly across the range when computing the median. METHOD.GROUPED is allowed even if there are fractional weights.

OMIT.PERCENTS arg

TOTALS.AREA	BODY	SUBTOTALS
MISSING	SUMMARY	ROW.TOTALS
BANNER.TOTALS	RESET	ARITHMETIC.OP

The arguments for

PERCENTS

PERCENTS is preceded by one or more of:

ROW COLUMN TOTAL NO.COL.100 NO.ROW.100

and may be followed by one of the following:

GOOD.N TOTAL.N RESPONSES

NO.COL.100 and NO.ROW.100 should always follow the use of COLUMN or ROW and cause any superfluous printing of the value “100%” to be omitted.

PLACES COUNTS nn

requests that the counts in the body of the table, the summary section and the totals area be printed to the specified number of places. If the table is not weighted, this is unnecessary. If the table is weighted, it is often useful to see the fractional part.

QUARTILES

requests that the .25 and .75 percentiles be included in the summary section along with the median. NOTE: if the SUMMARY subcommand is used, it must include either MEDIAN or QUARTILES or they will not appear.

REPORT ALL RESULTS

requests that a report be printed at the end of the table when ever F, Chi or significance tests are performed but have no significant results.

REQUIRE COUNTS / EXPECTED.VALUE nn

provides a threshold for the cell tests of significance. Any cell with a value that is not greater than the threshold is not flagged by the tests of significance.

SAMPLE

requests that the standard deviation be computed using the formula for the sample deviation rather than the formula for the population deviation. In the formula this means using n rather than n-1.

9
SURVEY: More Statistics

This chapter contains two major sections. The first section describes the arithmetic operations that can be performed on the columns of the tables. The second section describes the use of significance tests. The final section covers the TREND subcommand.

9.1 ARITHMETIC OPERATIONS

There are four arithmetic operations that can be used as subcommands to operate on the columns of the banners. They all create additional banner points which are the results of the arithmetic operations. These commands do not work with the TRANSPOSE subcommand. The subcommands are:

```
ADD   SUBTRACT   MULTIPLY  DIVIDE
```

The inputs are the frequencies in the individual cells with one exception. If the operation is a SUBTRACT, the input can be the percents. This special use of SUBTRACT is discussed later in this chapter.

Figure 9.1 Creating Combinations From a Single Variable

```
    ----------The Command----------

  SURVEY Paceset,
         LABELS 'paceset.lab';
      BAN Income.groups,
      ADD 'Sub*totals' Income.groups,
      STUB Sex, CELL.WIDTH 8, NO SUMMARY, NO MISSING;


   ----------The Output ----------

                        ============= Income of respondent ============

                                    $20,000              Sub      Sub      Sub
                  Total   Under        to    Over   totals   totals   totals
                 Sample $20,000  $40,000 $40,000      1+2      1+3      2+3

 Total              80       2       50      20       52       22       70
 Sample          100.0%   100.0%  100.0%  100.0%   100.0%   100.0%   100.0%

 Sex

 Male               39       -       21      10       21       10       31
                  48.8%            42.0%   50.0%    40.4%    45.5%    44.3%

 Female             40       2       28      10       30       12       38
                  50.0%   100.0%    56.0%   50.0%    57.7%    54.5%    54.3%
```

Figure 9.2 Combining Two Banner Points

```
----------The Command----------

SURVEY Paceset,
       LABELS 'paceset.lab';
    BAN Income.groups,
    ADD '$30,000 or more' income.groups 2 Income.groups 3,
    STUB Sex, CELL.WIDTH 8, NO MISSING, NO SUMMARY;


----------The Output----------

                     ===== Income of respondent ====

                                 $30,000
                Total   Under       to    Over $30,000
               Sample $30,000 $60,000 $60,000 or more

Total             80       2      50      20      70
Sample         100.0%   100.0%  100.0%  100.0%  100.0%

Sex

Male              39       -      21      10      31
                48.8%           42.0%   50.0%   44.3%

Female            40       2      28      10      38
                50.0%   100.0%   56.0%   50.0%   54.3%
```

The columns that are used and the columns created depend on the arguments that are provided. The basic modes are:

1. create a new banner point for each unique combination of values in a specified variable.
2. given two variables match the values so that variable 1/value 1 is paired with variable 2/value 2.
3. create a new banner point defined by individual variable/value pairs. The input to the new banner point can be taken from any combination of banner points that are on a given page of the output.

In this final type of specification, the input to subtract and divide can reference just two banner points. The input to add and multiply can specify multiple banner points. For a subtraction the second referenced banner point is subtracted from the first.

```
Subtract 'Difference' Q1A 2 Q33 3,
```

The input columns are those defined by variable Q1A value 2 and variable Q33 value 3. If the operation is a divide, the first column that is defined is divided by the second column.

The values used in these subcommands refer to the actual values of the data. Ranking the columns applies only to the input variables. The generated variables are always printed in natural order (low to high).

The labelling that is produced depends on the form of the request. If the request involves a single variable, the generated banner points are within the domain of that variable name or extended variable label and the string

that is supplied is used to label the individual banner points. If multiple banner points are generated, the value label is created from the supplied string and the values of the input columns.

In Figure 9.1, there are three new banner points generated from the Income.groups variable. The input string "Sub*totals* is combined with the appropriate arithmetic operator and the values that are involved. Since the request is an ADD, the operator is the "+" character. Note: the usual rules for labels break characters apply to these input strings. In Figure 9.2, the add request is for a single new column and the input label "$20,000 or more" is used without change.

Figure 9.3 illustrates the use of paired variables. In this situation the supplied string is used as the variable label. The SURVEY.LABELS code of 401 can be used here to provide an alternate string for this label. Because the label in this situation is not formatted across multiple lines, the number of characters that can be printed is determined by the number of generated banner points and the cell width.

The following are legal uses of these subcommands;

1. `ADD 'sum' Var1 1 3 5 Var2 2 4,`
   This produces a single new output column.
2. `ADD 'totals' Var1,`
   The number of output columns is the number of unique combinations of the values of the input variable.
3. `ADD 'Compare' Var1 Var2,`
   The number of output columns depends on the range of values for the first variable. The range for the second variable must also have the same range.

Figure 9.3 Using Paired Variables

```
---------- The Command ----------

SURVEY Work;
  BAN Var2 Var3, STUB Var1,  NO MISSING, NO SUMMARY,
  ADD 'var2+var3'  Var2 Var3;


---------- The Output ----------

                ==== VAR2 === ==== VAR3 ===   var2+var3

           Total
          Sample      1      2      1      2      1      2

Weighted     944    394    550    562    382    956    932
Total      100.0% 100.0% 100.0% 100.0% 100.0% 100.0% 100.0%

VAR1

       1     364     44    320    162    202    206    522
            38.6%  11.2%  58.2%  28.8%  52.9%  21.5%  56.0%

       2     580    350    230    400    180    750    410
            61.4%  88.8%  41.8%  71.2%  47.1%  78.5%  44.0%
```

---

4. `SUBTRACT 'Diff' Var1 3 Var1 5,`
   The single output column is the difference between the two input columns
5. `SUBTRACT 'Diff' Var1 1 Var2 1,`
   The single output column is the difference between the two columns. This is allowed even though the two columns are not from the same variable.
6. `MULTIPLY 'Product' Var3 1 3 5`
   This produces a single new banner point which is the product of three of the banner points belonging to Var3.
7. `DIVIDE 'Quotient' Var5 3 Var4 1,`
   This produces a single new banner point which is the result of dividing the values for Var5 value 3 by the values for Var4 value 1.

Subtraction can result in negative numbers. Division can result in very small numbers. Multiply can result in very large numbers. These results may be more satisfactory if additional subcommands such as CELL.WIDTH, and PLACES COUNTS are used to help with the formatting of the output.

DIfferent arithmetic operations can be combined in a single table:

```
ADD '1+2' Age 1 2,  SUBTRACT '3-1' Education 3 1,
```

However, they must be the same type of operation. A paired variable operation cannot be combined with a value/variable or single variable type of operation.

When the operation is either an add or subtract, the cell contains the resulting frequency and, if requested, the row and column percents. If the operation is either a multiply or a divide, row percents are not computed. Whatever the operation, the percents can be omitted by using:

```
OMIT.PERCENTS ARITHMETIC.OP,
```

Figure 9.4 Subtraction Using Percents

		========== Income of respondent =========					
	Total Sample	Under $30, 000	$30, 000 to $60, 000	Over $ 60,000	diff 1-2	diff 1-3	diff 2-3
Total	80	2	50	20			
Sample	100.0%	2.5%	62.5%	25.0%	-60.0%	-22.5%	37.5%
Sex							
Male	39	-	21	10			
	100.0%		53.8%	25.6%	-53.8%	-25.6%	28.2%
	48.8%		42.0%	50.0%	-42.0%	-50.0%	-8.0%
Female	40	2	28	10			
	100.0%	5.0%	70.0%	25.0%	-65.0%	-20.0%	45.0%
	50.0%	100.0%	56.0%	50.0%	44.0%	50.0%	6.0%

9.2 Subtraction Using Percents

It is possible to use SUBTRACT based on the percents rather than the cell frequencies. The following subcommands are used to produce the table in Figure 9.4.

```
SUBTRACT PERCENTS 'Diff' Income.groups, ROW COL NO.COL.100 PERCENTS,
```

The keyword PERCENTS can only be used with the SUBTRACT subcommand. The cells do not contain counts. They contain only differences between any percents that have been requested.

Figure 9.5 ADD to Create Banner Subtotals

```
SURVEY paceset, LABELS paceset.lab;
STUB sex,
CELL.WIDTH 8,
NO MISSING,,
BAN income.group ( NO.VALUES ),
ADD 'Under $30,000' income.group 1,
ADD '$30,000 to $60,000' income.group 2,
ADD '$60,000 and under' income.group 1 2,
ADD 'Over $60,000' Income.group 3;
$
```

```
                    ===== Income of respondent ====

                                 $30,000 $60,000
                Total   Under      to      and    Over
               Sample $30,000 $60,000   under $60,000

Total             80       2      50       52      20
Sample         100.0%   100.0%  100.0%   100.0%  100.0%

Sex

Male              39       -      21       21      10
                48.8%           42.0%    40.4%   50.0%

Female            40       2      28       30      10
                50.0%   100.0%   56.0%    57.7%   50.0%

Base              79       2      49       51      20
                98.8%   100.0%   98.0%    98.1%  100.0%
Mean            1.51     2.00     1.57     1.59    1.50
S.D.            0.50     0.00     0.50     0.50    0.51
```

9.3 Using ADD to Create Subtotals

When the input to an ADD subcommand is the values from a single variable, the result is a subtotal. However, these subtotals are always placed after the final value. Because ADD does not check how many values are being added, it is possible to ADD a single value and construct the output the way that you want it.

Figure 9.5 illustrates the creation of a subtotal interleaved between the values. The keyword NO.VALUES following the banner variable name causes the usual banner columns to be omitted. Then the four ADD subcommands create the columns that are finally produced. The columns are in the order in which they are entered. Summary statistics and significance tests are available for these values. However, medians on external variables and the INDEX subcommand are not supported for these subtotals.

9.4 SIGNIFICANCE TESTS

Testing individual cells for significance raises is a capability that should be used with caution. In a 10 by 5 table with a resulting 50 tests, it is very likely that some of the cells will appear to be significantly different from other cells just by chance. The results, therefore, should be viewed only as an indication that there may be something of interest going on and further examination of the data may be worth doing.

The tests assume that the cells being tested are independent of each other. When a test is done between columns of variables such as region or gender, it is reasonable to assume that none of the participants are represented in both cells. However, it is possible to request tests between variables and between specific columns as represented by variable/value pairs. In such situations, care must be exercised as the tests are only appropriate when the assumption of independence is valid.

There is some protection built into the testing. When the tests are requested a test is done to see if the smaller of Np (the number expected) and Nq (where q is 1-p) is at least 10. If this test fails the computations are not performed. This is a difference from releases before P-STAT version 2.22 release 2. If you wish to compare, results with previous jobs you can use the subcommands:

```
REQUIRE PROPORTION 0      and
REQUIRE INDEPENDENCE 5
```

There are a number of options when using the TEST subcommand. The format of the subcommand is:

```
TEST ARG1 ARG2 NUM NUM VARIABLE.LIST
```

TEST and the first argument are required. The final arguments are optional.

1. ARG1 `PROPORTIONS or INDEPENDENCE or MEANS` where PROPORTIONS and INDEPENDENCE refer to the formula that is used to test cells in the body of the table and MEANS requests a t-test on the means in the summary section of the table.
2. ARG2 `COMBINED.VALUES or PAIRED.BVAR or PAIRED.VALUES`. If this field is omitted, the tests are done comparing columns within individual banner variables.
3. NUM provides the desired level of significance. If no value is supplied, it is assumed to be .95 . A second optional test value may also be supplied.
4. VARIABLE.LIST is assumed to be all the banner variables that are defined. The contents of this list when either PAIRED.BVAR or PAIRED.VALUES are requested is discussed later in this chapter.

9.5 Testing the Cells

Two different tests for cell significance are provided. One is based on t-tests, the other on a test for independence of proportions. Although the formulas are not the same, the results are seldom different. A t-test is a test between 2 means. The way that a mean value is calculated for a cell is to consider the total for the column as if it were composed of a series of 0's and 1's. The 1's are those cases which fall in the cell. The 0's are those cases which do not fall in the cell.

Figure 9.6 illustrates the SURVEY command and the resulting printout using the t-test method. In the t-test approach a cell with 51 replies in a column with a total count of 191 is viewed as 51 replies of 1 and 140 replies of 0. Thus the mean and sum for the cell can be computed. In this example the totals of all the zero and one values is 51. The total number of cases is 191. This produces a mean of .267. This procedure is repeated for the next

column, producing a mean of .1821 (65 divided by 357). These values can now be compared using the standard t-test formula with degrees of freedom equal to one.

The tests for significant differences between cells can also be done either between the columns of the variables or between each column and the combined values of the other columns for that variable.

```
TEST PROPORTIONS,          test for each variable in turn
TEST PROPORTIONS COMBINED.VALUES
     test each column against the combined values
TEST PROPORTIONS .99 Age   test of variable Age at .99 sig. level
TEST PROPORTIONS .95       use .95 as the level of significance
```

Figure 9.6 Significance Test

```
SURVEY T3,  LABELS 'test.lab';
     BANNER Region,  STUB Gender,
     TEST PROPORTION .90,
     NO MISSING,   SUMMARY CONTAINS MEANS;

                      ====== Region ======

                 Total  North  South
                Sample   East   East   West
                            A      B      C

Total             990    191    357    442
Sample          100.0% 100.0% 100.0% 100.0%

Gender

female            182     51     65     66
                 18.4%  26.7%  18.2%  14.9%
                          BC

male              808    140    292    376
                 81.6%  73.3%  81.8%  85.1%
                                 A      A

Mean             1.82   1.73   1.82   1.85

.90 significance level used for t-test of proportions
```

In Figure 9.6, females from the Northeast appear to differ from females from either of the other two regions. Thus A is different than B and A is different from C. Possible alternate labelling might have been:

```
female            182     51     65     66
                 18.4%  26.7%  18.2%  14.9%
                                 A      A
or:
female            182     51     65     66
                 18.4%  26.7%  18.2%  14.9%
                          BC     A      A
```

Having the letters under both of the columns adds no additional information and if B is then different from C, causes confusion and clutter in the printout. The labelling choice was to use the letters once and place them under the column with the larger column percent.

When the individual columns are tested against the combined values, the letter beneath the column indicates that column is different from the combined values of the other columns. If the letter is upper case, the indicates that the column percent for that column is greater than the column percent of the combined other values. A lower case letter indicates that the column percent for that column was less than the combined column percent.

Figure 9.7 Significance Test: Combining Values

```
SURVEY T3,  LABELS 'test.lab';
      BANNER Region,  STUB Gender,
      TEST PROPORTION COMBINED.VALUES .95,
      NO MISSING,   SUMMARY CONTAINS MEANS;
```

```
                    ====== Region ======

              Total  North  South
             Sample   East   East   West
                         A      B      C

Total          990    191    357    442
Sample       100.0% 100.0% 100.0% 100.0%

Gender

female         182     51     65     66
              18.4%  26.7%  18.2%  14.9%
                        A             c

male           808    140    292    376
              81.6%  73.3%  81.8%  85.1%
                        a             C

Mean          1.82   1.73   1.82   1.85

.95 significance level used for t-test of proportions
Values of each banner variable are tested against the combined other values
```

9.6 Multiple Test Values.

It can be useful to supply more than one test value.

```
TEST PROPORTION .99  .95,
```

When this is used upper case characters are used to label the cells which are significant at the higher test value and lower case characters are used to label cells which are only significant at the lower test value. This feature has restrictions.

The first restriction is that the higher value must be supplied first in the subcommand. The number of columns on the page is limited to 26, the number of letters in the alphabet. This feature has a conflict with COMBINED.VALUES and MARK ALL TESTS both of which use the difference between upper case and lower case letters to represent differences.

9.7 Tests of Independence

In the second approach, a test for the significance of the difference between two independent (uncorrelated) proportions can be done as follows.

$$Z = \frac{P_1 - P_2}{Sp_1 - p_2} = \frac{P_1 - P_2}{\sqrt{pq\,[\,(1/N1) + (1/N2)\,]}}$$

where p is weighted proportion:

$$p = \frac{f1 + f2}{N1 + N2} \qquad \text{and } q = 1 - p$$

f1 is the value of the cell in the first column. f2 is the value of the cell in the second column. N1 and N2 are column totals.

Using the numbers in Figure 9.7, the first test is to see of the proportion .267 (51 divided by 191) is significantly different from.182 (65 divided by 357). That is, is there a difference between column 1 respondents and column 2 respondents.

The figures in the equation for the first test:

```
P1 =.267        P2 =.182
f1 = 51         f2 = 65
N1 = 191        N2 = 357

p = (51 + 65) / (191 + 357) =.212
q = 1 -.212 =.788
```

produce a Z value of 2.322. Z may be interpreted as a deviate of the unit normal curve *provided that N1 and N2 are reasonably large and that p is neither very large nor very small.* An arbitrary test is used to determine that it is reasonable to interpret Z in this way. If the smaller value of p or q multiplied by the smaller value of N is not greater than 10, the test is not done.

A Z score of 2.322 is greater than 1.96, the value that one would expect for a difference significant at the 5% level. A Z score of 2.58 is significant at the 1% level. A two-tailed test is done as the result may differ at either end of the distribution.

9.8 Controlling and Labelling the Tests

In a large table with many cells, the number of false positive results is considerable. Cells with very small counts are very apt to produce results that look significant when the cell size precludes a meaningful result. Cells with very small counts can be excluded from the cell tests by using the REQUIRE subcommand.

```
REQUIRE Count 5,
REQUIRE Expected.Value 5,
```

Either the cell count or the expected value can be tested. In a standard test, the test is not done if either cell has a value that is less than or equal to the test value. Using REQUIRE will make it easier to see those cells that can reasonably be expected to have significant difference results.

REQUIRE can also be used to ensure that even empty cells are tested. The assumed REQUIRE value is “1”. “REQUIRE 0” requests that the tests be done on all cells.

When Significance tests are used, a line is printed at the bottom of the table annotating the action that was taken including the level of significance used in the tests. This labelling can be changed by supplying the appropriate codes in the labels file after the SURVEY.LABELS variable. SURVEY.LABELS is discussed in the chapter “SURVEY: Enhancing Layout and Appearance”.

There are four codes that can be used:

```
(301) tests of the mean
(302) cell tests using t-tests of proportions
(303) cell tests of independence
(304) Values of each banner variable are tested against the combined
other values
```

The next five values are used only when REPORT ALL RESULTS is used and none of the tests have been significant.

```
(310) used with TEST ALL RESULTS and CHI
(311) used with TEST ALL RESULTS and F
(312) used with TEST ALL RESULTS and TEST PROP
(313) used with TEST ALL RESULTS and TEST INDEPENDENCE
(314) used with TEST ALL RESULTS and TEST MEANS
```

A labels file, named Surv.lab, might contain:

```
SURVEY.LABELS (301) ' '
              (302) 'Test only cases with expected values > 5'
              (303) 'Independence tests on all variables' /
```

The command might look like:

```
SURVEY Paceset, LABELS 'Paceset.lab'  'Surv.lab';
BANNER Age Sex, STUB Num.TV,
TEST MEANS, TEST PROPORTIONS, REQUIRE EXPECTED.VALUE 5;
```

Using this labels file, there will be no message about the means tests. The message about proportions contains just the text for code 302 taken from the labels file. These messages come before any bottom titles. If there are no cases that have any significant differences on the tests, the message is not printed even though the columns have the letter codes.

When any of the tests are done using banner variables, all the columns for a given variable must fit on a single page. This restriction is done because of the limited number of characters available to label the columns.

9.9 Weighted Tables and Significance Tests

When the table is weighted the significance tests are done using the weighted data unless unweighted data are requested. The subcommand is:

```
USE UNWEIGHTED.BASE,
```

The default is:

```
USE WEIGHTED.BASE,
```

When the unweighted base is used, the values for the cells are also unweighted. When the weighted base is used both the cells and the base are weighted.

9.10 Missing Data and Significance Tests

The assumption is that missing data is invalid and is excluded from the significance tests. The base, whether weighted or unweighted is the good (non-missing) count. However, in a survey it is possible that the missing data can be interpreted as a zero that has been coded as missing so that it is not reflected in the summary statistics but should be included in the significance tests. This can also be accomplished with the USE subcommand.

```
USE TOTAL.N,
```

If the data are unweighted and you wish the significance tests on the unweighted total n you can use:

```
USE UNWEIGHTED.N,
```

Because the default setting for the base of column percents is the total and the default setting for the significance tests is the base (good.n), the results can look confusing when there is missing data. If you have missing data, you can specify

```
COLUMN PERCENTS BASED GOOD.N
```

or, if your percents are based on the TOTAL.N and you wish the same base for your significance tests:

```
USE TOTAL.N
```

If you wish to have your tests done using the same base as your column percents, you may specify "USE PERCENTS". For multiple response stubs the subcommand "USE RESPONSES" is also available.

Figure 9.8 Labelling the Significance tests.

```
SURVEY T3,  LABELS 'test.lab';
            BANNER Region,   STUB Gender, NO MISSING,
            MARK ALL TESTS,  TEST PROPORTION .90,
            SUMMARY CONTAINS MEANS;
 $
                            ====== Region ======

                     Total  North  South
                    Sample   East   East   West
                               A      B      C

      Total            990    191    357    442
      Sample         100.0% 100.0% 100.0% 100.0%

      Gender

      female           182     51     65     66
                     18.4%  26.7%  18.2%  14.9%
                              BC      a      a

      male             808    140    292    376
                     81.6%  73.3%  81.8%  85.1%
                              bc      A      A

      Mean            1.82   1.73   1.82   1.85

      .90 significance level used for t-test of proportions
```

9.11 Labelling Significance Tests

If there are significant differences between columns, the column which has the larger test value is labelled with the letter of the column with the smaller test value. If is possible to request that both columns be labelled. This is done by using the subcommand MARK ALL TESTS. This causes both columns to be labelled. The column with the larger test value is given the upper case character that represents the other column. The column with the lower test value is labelled with the lower case letter of the other column. See Figure 9.8

If the tests are done against the combined values, the default is to use the character that labels the column to indicate a significant result and the case to indicate the direction. This can be changed by using the CHARACTER subcommand. For example:

```
CHAR TESTS '+' '-',
```

causes the plus sign (+) to be used for a column if the column percent is greater than that of the combined values and a minus sign (-) if the column percent is less than that of the combined values.

Figure 9.9 Paired Variables in Significance Tests

```
----------The Command----------


SURVEY Tsig;
   BAN Var2  Var3 ( 1 3 ), STUB Var1,
   TEST PROP PAIRED.BVAR Var2 Var3,
   NO SUMMARY, NO MISSING;


----------The Output----------
```

		======= VAR2 =======			======= VAR3 =======		
	Total Sample	1	2	3	1	2	3
		A	B	C	D	E	F
Weighted Total	944	222	170	40	340	172	-
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%	
VAR1							
1	364	22	140	40	140	22	-
	38.6%	9.9%	82.4%	100.0%	41.2%	12.8%	
			E		**A**		
2	580	200	30	-	200	150	-
	61.4%	90.1%	17.6%		58.8%	87.2%	
		D				**B**	

9.12 Significance Tests on Paired Banner Variables

If two banner variables are related in some way, perhaps because the study was done in waves, it is possible to compare the corresponding values of the variables. *All the significance tests assume that the input columns are independent. If this assumption is not true, the results are meaningless and these operations should not be used*.

```
BAN Gender Q1A98 Q1A99,
TEST PROP PAIRED.BVAR Q1A98 Q1A99,
```

If you have paired banner variables, the names do not need to be provided. There are some restrictions on the use of this test in addition to the assumption of independence.

Figure 9.10 Choosing the Columns for Significance Tests

```
----------The Command----------


SURVEY Tsig;
  BAN Var2 Var3,  STUB Var1,

  TEST PROP PAIRED.VALUES  Var2 1  Var2 3  var2 2  var3 2,
  TEST MEANS PAIRED.VALUES Var2 1  Var2 2,

  SUMMARY CONTAINS MEANS, NO MISSING;


----------The Output----------


                    ======= VAR2 ======= ==== VAR3 ===

              Total
             Sample      1      2      3      1      2
                         A      B      C             D

Weighted        944    222    170     40    340    172
Total         100.0% 100.0% 100.0% 100.0% 100.0% 100.0%

VAR1

       1        364     22    140     40    140     22
              38.6%   9.9%  82.4% 100.0%  41.2%  12.8%
                                 D      A

       2        580    200     30      -    200    150
              61.4%  90.1%  17.6%         58.8%  87.2%
                                        A            B

Mean           1.61   1.90   1.18   1.00   1.59   1.87
                           B
.95 confidence level used for t-test of means
.95 confidence level used for t-test of proportions
```

---

1. Each pair of variables must fit on the same page of the output
2. The paired variables must have the same range of values. If they do not, the command produces an error message.
3. Empty columns cannot be squeezed out.

The BVAR.PER.PAGE subcommand can be used to help insure the placement of the banner variables. This subcommand is only appropriate when the variables can be meaningfully compared. The tests are done comparing each value of the first variable in the pair with the same value in the second variable of the pair.

```
BAN Q1A.98  TO Q3A99,
TEST PROP PAIRED.BVAR,
BVAR.PER.PAGE 2,
```

If there is a problem with the observed ranges of the variables, you may supply the ranges to be used:

```
BAN Q1A98 ( 1 3 ) Q1A99 Q2A98 Q2B99 ( 1 3 ) Q3A98 ( 1 3 ) Q3B99,
```

The use of ranges in this manner overrides the default which uses the values actually observed in the data file. Figure 9.9 illustrates the used of PAIRED.BVAR is such a situation. When PAIRED.BVAR is used, one test is done for each value of the paired variables. There is no within variable comparisons. This type of comparison might be appropriate for comparing the results of a current study with those of a previous study.

A variable can be used more than once in a table. However, if significance testing is done, only the final use of the variable is used in the tests.

9.13 Significance Tests Between Individual Columns

It is possible to limit the significance tests that are done to just the columns which you think might produce interesting results. This is done by using the PAIRED.VALUES keyword in the TEST subcommand and adding the variable value pairs to be tested.

```
TEST PROP PAIRED.VALUES Var4 1 Var4 3 Var6 1 Var7 1,
```

In this example two pairs of tests are done and four banner points will be labelled with heading letters.

If you wish to supply the test threshold, the test values precede the paired value information

```
TEST PROP PAIRED.VALUES .9 Var4 1 Var4 3 Var6 1 Var7 1,
```

The assumption is that the data in these columns are independent, not paired or somewhere between independent and paired. Special care must be taken in this respect when using tests across variables. As always these usages of significance tests should not be just blind fishing. When many tests are performed a few will be significant on the basis of chance. When you are using selected paired tests of independent columns on the basis of a hypotheses, a significant result is more meaningful than one obtained by a catchall do all possible tests situation.

9.14 TREND SUBCOMMAND

The TEST TREND subcommand produces an additional banner column which contains a measure of the trend of the data and the chi square. The magnitude of the trend value is an indication of the direction which that row is taking given the progression of the column totals. The assumption is that the values of the banner variables provide a measure over time. Since the best predictor of the next value is assumed to be the immediately previous value, the higher values are given a greater weight in the regression equation.

Figure 9.11 provides an example in which the totals increase each measuring periods while the frequency of the first row remains the same. The negative trend statistic reflects that fact that the cells represent a decreasing proportion of the column totals. The expectation is that this proportion decrease further given more data. The positive trend statistic for the second row indicates a situation where there is an upwards trend. The cell values are increasing at a faster rate than the column totals.

Figure 9.11 Trend Analysis: Example 1

```
---------- The Command ----------
 SURVEY Work,
 BAN Var2, STUB Var1,  TEST TREND,
 LAYOUT LABELS TOTALS QUESTION BODY;
 $


 ---------- The Output ----------

                    =============== VAR2 ===============

              Total                                 Trend &
             Sample      1      2      3      4       Chi

Weighted        220     40     50     60     70
Total         100.0% 100.0% 100.0% 100.0% 100.0%

VAR1

       1         40     10     10     10     10    -1.45
              18.2%  25.0%  20.0%  16.7%  14.3%    2.17

       2        180     30     40     50     60     1.45
              81.8%  75.0%  80.0%  83.3%  85.7%    2.17
```

Figure 9.12 Trend Analysis: Example 2

```
                    =============== VAR4 ===============

              Total                                 Trend &
             Sample      1      2      3      4       Chi

Weighted        220     40     50     60     70
Total         100.0% 100.0% 100.0% 100.0% 100.0%

VAR3

       1         44      8     10     12     14     0.00
              20.0%  20.0%  20.0%  20.0%  20.0%    0.00

       2         40     10     10     10     10    -1.45
              18.2%  25.0%  20.0%  16.7%  14.3%    2.17

       3        136     22     30     38     46     1.15
              61.8%  55.0%  60.0%  63.3%  65.7%    1.37
```

In Figure 9.12 the stub variable with 3 values illustrates three different situations. The first row reflects exactly the pattern of the column totals. The second row has values that are the same for each banner point and there fore are decreasing relative to the column totals. The third row has values which increase more rapidly than the column totals.

Figure 9.13 Trend Analysis: Example 3

```
                    =============== VAR2 ===============

                 Total                                 Trend &
                Sample      1      2      3      4       Chi

 Weighted         220     40     50     60     70
 Total          100.0% 100.0% 100.0% 100.0% 100.0%

 VAR1

       1           66     20     16     16     14     -3.23
                 30.0%  50.0%  32.0%  26.7%  20.0%   11.37

       2          154     20     34     44     56      3.23
                 70.0%  50.0%  68.0%  73.3%  80.0%   11.37
```

The trend statistic is an indication of the slope of the regression line. This can be seen by looking at the SURVEY output in Figure 9.13 and the plot in Figure 9.14.

The plot in Figure 9.14 shows the slope of three lines: the columns totals, the first row and the second row. The slope of the first row is steeper than the slope of the total while the slope of the second row is negative. A negative trend statistic indicates a slope that is increasing at a slower rate than the column totals. A plot of the data in Figure 9.12 would show that the second row which has a negative trend value would be flat. The values are not decreasing. Rather it is the rate compared to the rate of the totals that is decreasing.

The trend statistic is not presented with any measure of its significance. It is useful as an additional confirmation when the data seems to have a pattern over time. Empty columns in the banner are ignored. Empty cells when there is a column total are treated as zero.

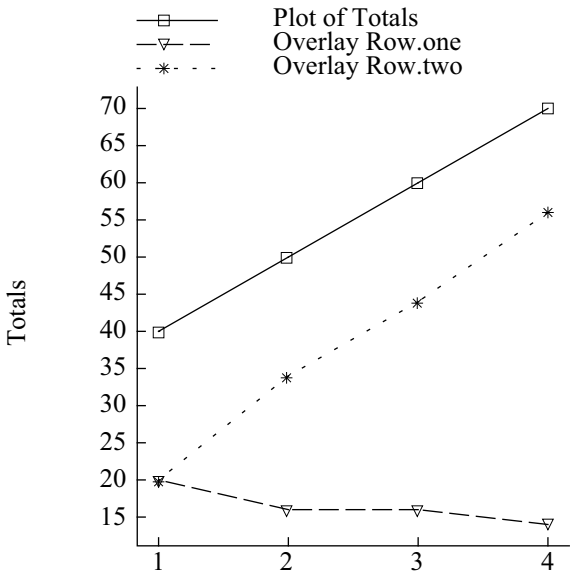
```
TEST TREND
```

with no further arguments produces a trend statistic for all the defined banner variables. If there are multiple banner variables, you can choose which will have a trend statistic by supplying the variable names:

```
BANNER V1 to V5, TEST TREND V1 V4,
```

The chi-square statistic reflect how closely the cell frequencies are to the expected frequencies given the column totals and the row total for that row. The closer the expected value is to the actual value, the lower the chi-square.

Figure 9.14 Plot of the Trend Values



SUMMARY

SURVEY

```
SURVEY T3,  LABELS 'test.lab';
        BANNER Region,  STUB Gender,
        TEST PROPORTION .90,
        SUMMARY CONTAINS MEANS;
 $
```

Optional Subcommands:

The variable/values required by the four arithmetic operations can be supplied in three ways. If the arguments are two variable names they must have the same ranges and the output is as many columns as that range. The arguments are a single variable name followed by one or more values, it is assumed that operation is to be performed on the values of that variable. The final form is a list of 1 or more variable/value pairs. ADD and MULTIPLY can operated on more than two values. SUBTRACT and DIVIDE are limited to paired values.

ADD 'cs' list of variable/values

causes the values provided in the variable/value list to be added together forming a single new banner column which has the supplied character string as a label.

SUBTRACT 'cs' vn vn or vn nn vn nn

causes the second value provided in the variable/value list to be subtracted from the first value. This creates a single new banner column which has the supplied character string as a label.

MULTIPLY 'cs' list of variable/values

causes the values provided in the variable/value list to be multiplied, forming a single new banner column which has the supplied character string as a label

DIVIDE 'cs' vn vn or vn nn vn nn

causes the first value provided in the variable/value list to be divided by the second value. This creates a single new banner column which has the supplied character string as a label.

MARK ALL TESTS

requests that all cells involved in the tests be marked not just the cell with the larger test value.

REQUIRE COUNT / EXPECTED.VALUE nn
PROPORTION / INDEPENDENCE nn

provides a base level for the cell contents below which testing is not to be done. This value is based either on the frequency in the cell or its expected value. If the value is zero, all cells are included in the tests.

When the arguments are PROPORTION or INDEPENDENCE, the test value that is normally a 10 can be altered. This should only be done when trying to compare results with previous output before these tests were implemented.

TEST nn

requests that t-tests be done for each pair of columns within a banner variable. The assumed level of significance is .90 . This may be changed by providing a number such as .95 or .99 to control the significance level. Tests may also be done on the means in the summary section. This is requested by :

```
TEST MEANS,
TEST MEANS .95,
TEST MEANS .95 Age Education,
TEST MEANS COMBINED.VALUES .99 Age Occupation,
```

TEST PROPORTIONS / INDEPENDENCE nn nn

request that tests be done between the cells of the banner variables. Two tests are available. PROPORTIONS uses a t-test method. INDEPENDENCE uses a test of independence. The level of significance can be specified. Usually the tests are done between each pair of columns of the banner variable. The test may also be requested of each column against the combined values of the other columns for that banner variable..

```
TEST PROPORTIONS COMBINED.VALUES .95,
TEST INDEPENDENCE COMBINED.VALUES .99 95,
```

Usually the tests are done on all the banner variables. However, a list of variables can be provided and the tests will then be on just those variables.

```
TEST PROP .95 Age Education,
TEST PROP COMBINED.VALUES .99 Age Occupation,
```

TEST arg arg nn nn variable list

this form is used when the selection of which columns to test is not the default which tests every banner point within each of the banner variables. The first argument is required and is always one of:

```
PROPORTIONS       INDEPENDENCE       MEANS
```

The second argument, which is not required, may be one of:

```
COMBINED.VALUES            PAIRED.BVAR        PAIRED.VALUES
```

If COMBINED.VALUES is used, the tests are done one each selected banner point against the total of the other values for that variable. If variable names are not provided in the variable list, all banner points are used. If PAIRED.BVAR is used, there must be two variables in the variable list and the tests are done between each of the banner points for the first variable against the corresponding banner point in the second variable. If PAIRED.VALUES is used, the variable list contains pairs of variables and values to define exactly which banner points are to be compared.

TEST TREND variable list

The variable list is optional. If it is not supplied the trend statistic is supplied for all the banner variables. These variables are assumed to represent data collected over time: quarterly, yearly, etc. The most recent values receive the most weight in the regression equation. The results are shown in a new banner column which contains the trend statistic on the first line and a chi-square on the second line.

USE WEIGHTED.BASE / UNWEIGHTED.BASE / WEIGHTED.N / UNWEIGHTED.n / RESPONSES / PERCENTS

Significance tests can be based on either the weighted or unweighted good count or the weighted or unweighted total count. USE BASE and USE TOTAL.N in a weighted table are assumed to mean the weighted base or totals.

If weighted base or totals are selected the weighted values of the cells are used. If unweighted base or totals are requested, the unweighted cell values are used. The total counts are used when the missing values represent a response such as “no opinion” which might be coded a zero rather than a not applicable

or not asked situation where the case is truly missing. If you have both types of missing, use the MISSING subcommand which allows you to remove selected types of missing from the table.

```
MISSING  M1,
```

causes all cases that are MISSING2 or MISSING3 from the table. Use of 'USE TOTAL.N' will cause the cases which have a MISSING1 to be included in calculating the significance tests and omitted from the means in the summary section.

```
USE PERCENTS
```

causes the significance tests to be based in the same way that the column percents are based. Since the default for column percents is the TOTAL.N and the default for significance tests is GOOD.N, the results may not show a difference that would appear if the column percents were also based on the GOOD.N. This is particularly useful in tables where the percents are the only contents in the body of the table.

```
USE RESPONSES
```

is available for multiple response tables. This will be the same as specifying that your percents are based on RESPONSES and USE PERCENTS is included.

10
SURVEY: Other Features

This chapter deals with an assortment of topics that do not fit naturally into the previous five chapters. The major topics to be covered are:

- Definition and include files
- Table of Contents
- Output files
- Producing many tables for a series of subgroups
- Methods for handling variables with a large range but few values
- Sizes and program limitations
- Other Helpful Hints

10.1 THE DEFINITION FILE, RESET AND RESTART

The definition file can be used to store any subcommands that are used in your SURVEY commands. Most people have preferences in table appearance. Individual tables may require changes, but the basic look will often be the same for a whole project. The definition file is an easy way to store the preferences in a single place, accessible by every SURVEY command. Changes may then be made to individual subcommands. The identifier that is used is “DEF” and it is followed by the name of the external file which contains the subcommands.

```
SURVEY Paceset,  LABELS 'Paceset.Lab',  DEF 'Mydefs';
```

The definition file may contain any SURVEY subcommands, including all the PostScript font settings, titles, DEFINE statements, and even the STUB and BANNER subcommands used to define rows and columns. Because system settings are done before the definition file is read, any settings made in the definition file become the current settings.

If, in the course of the run, the RESET subcommand is used, all of the original system values except the BANNER variable, the STUB variable and the subtotals or nets are restored. These three may be individually reset with:

```
RESET BANNERS,
RESET STUBS,
RESET SUBTOTALS,
```

RESET does not restore the values set in the external file of definitions.

```
RESTART
```

The RESTART subcommand does a RESET, then resets the banners, the stubs and the subtotals, and finally rereads the definition file and processes the subcommands found there. RESTART has no arguments.

Figure 10.1 contains an example of a definition file and its usage. In this figure the definition file contains the LAYOUT subcommand and four other subcommands which control the final appearance of the table. The definition file usually contains all the settings that you prefer. It is then very easy to change individual settings as needed for a specific table.

Figure 10.1 The Definition File

Mydefs is an external file containing the following SURVEY subcommands:

```
LAYOUT QUESTION LABELS TOTALS BODY SUMMARY,
PLACES PERCENTS 2,
ROW.TOTALS ON RIGHT,
USE VALUES, SUBTOTALS BEFORE LOW;
```

The P-STAT command which uses the definitions

```
SURVEY Paceset, LABELS 'Paceset.lab',
   DEF 'Mydefs';
BAN Age,  STUB Sex,  NO SUMMARY;

$
```

```
Sex

              ======== Age =======

               Under  30 to   Over   Total
                  30     50     50  Sample

Total             28     27     23      80
Sample        100.00 100.00 100.00  100.00

Male              20     11      7      39
                71.43  40.74  30.43   48.75

Female             8     16     15      40
                28.57  59.26  65.22   50.00
```

10.2 Include Files

INCLUDE files are like DEF files. Both contain a series of subcommands that are to be executed. However, the DEF file is specified once in the *command*. It is immediately executed and when there is a RESTART it is reread and re-executed. An INCLUDE file is specified as a *subcommand* and is executed only once when it is mentioned.

The include file can contain just a few settings, or it can contain all the information needed to do an entire set of tables. In a project that has a great many DEFINE or DEFINE.NET subcommands, it is often useful to store them in a file which can be used over and over again in different SURVEY commands.

File Mynets contains:

```
DEFINE “Few television sets”  Num.TV 1 2,
DEFINE “Many television sets” Num.TV 3 TO 9;
```

```
SURVEY Paceset, LABELS 'Paceset.lab',
       DEF 'Mysubs';

  INCLUDE 'Mynets',
  BAN Age, STUB Num.TV, NO SUMMARY;
$
```

The INCLUDE file is opened, read and executed as it is encountered in the subcommand stream. It is then closed and forgotten (but not deleted). It may be reused with another INCLUDE subcommand, but neither RESET nor RESTART re-execute its contents.

10.3 Macros

In stream macros can be very useful in constructing SURVEY runs. Include files can only be used to hold subcommand information and they cannot contain arguments. Macros can be used in PPL, in command text or in subcommand text. Figure 10.2 contains 3 macros. Macro pscript contains identifiers that are to be used when the output is to be formatted for PostScript. Macro files contains the names of the files to be used and macro surv.template contains the standard SURVEY subcommands for the current project.

Figure 10.2 Macros

```
MACRO pscript $
/* add this macro when PostScript output is desired */
lines 50,
postscript, landscape
ENDMACRO $

MACRO files $
/* change this macro to point to the files for the current run */
labels 'e:\sbfiles\compcor\project3.lab',
pr 'e:\sbfiles\compcor\project3.ps'
ENDMACRO $

MACRO surv.template $
/* this macro contains the standard layout for this project */
tabnum after titles,
layout question labels totals missing summary subtotals body,
page labels, commas,
column row percents based good.n,
places means 0
ENDMACRO $

SURVEY Projfile, !!pscript,  weight wtnum,  !!files ;
!!surv.template,
BANNERS Q43w Q99 Gender,  STUBS Q23a TO Q23i;
$
```

In Figure 10.2 the final identifier or subcommand is NOT followed by a terminating comma. The commas are added as needed when the macro is invoked.

10.4 TABLE OF CONTENTS

A table of contents can be produced for each individual SURVEY command. Alternatively a table of contents may be started in one SURVEY command and added to in subsequent SURVEY commands. When the study is complete, it can then printed in a SURVEY command that does nothing but print the contents. The table of contents contains a table number, page number and a description of the table. Unless a table title has been provided, the description is either the stub variable name or the extended label. Table titles and table numbers are features, designed especially for the table of contents, which may also be used without producing a table of contents.

If CONTENTS is used as a *subcommand*, the contents are printed after the final page of the current command. The TITLES are the titles in effect when the last table is printed. The contents are not saved and, therefore, cannot be added to in subsequent SURVEY sessions.

```
SURVEY Paceset, LABELS 'Paceset.lab';
CONTENTS,
BAN Age,  STUB Own Income.groups,  GR.STUB VCR to Ans.Mach, ....
```

If CONTENTS is used as an *identifier*, the name of the file which is to contain the contents must be provided. The output is written to that file and is only displayed when the keyword DISPLAY is used to request that the contents be printed.

```
SURVEY Paceset, LABELS 'Paceset.lab'  CONTENTS 'Paceset.con';
```

The contents file is assumed to be a new file unless the argument “OLD” is added to the CONTENTS subcommand. When OLD is used, the existing contents are copied to a temporary work file. As each table is completed, the contents entry is added to this temporary file. When the command is complete, the temporary contents file is copied back to the permanent file.

```
SURVEY Paceset, LABELS 'Paceset.lab'  CONTENTS 'Paceset.con' OLD;
```

The contents are printed at the end of the SURVEY command’s printout only if the keyword “DISPLAY” is added to the CONTENTS subcommand. If the only purpose of the SURVEY command is to print an existing table of contents, a P-STAT system file must be available even though it is not used and CONTENTS must specify both OLD and DISPLAY. The TITLES can be supplied in a TITLES command or a SURVEY subcommand.

```
SURVEY Paceset, LABELS 'Paceset.lab'
       CONTENTS 'Paceset.con' OLD DISPLAY;
TITLES '.DATE.';
$
```

The formatting of the contents output depends on the output width of the page. The first 18 spaces are saved for the table number and page number. Specify an output width of at least 98 if you have long extended variable labels. If the output width is not wide enough to hold the entire label, it will be broken across 2 output lines.

10.5 Table Number and Page Number

If the table number and page number are not supplied, it is assumed that they both start with the number “1”. If you are adding information to an existing table of contents, it is important to supply the correct starting table and page numbers.

```
SURVEY Paceset, LABELS 'Paceset.lab',
       CONTENTS 'Paceset.con' NEW;
SURVEY Paceset, LABELS 'Paceset.lab'
       PAGE.NUMBER  14,
       TABLE.NUMBER 10,
       CONTENTS 'Paceset.con' OLD DISPLAY;
```

The page number is used in the table of contents, but prints on each page of the table if you have a TITLE which contains either the .PAGE. or .RPAGE. system variable. The table number is used in the table of contents

and also appears in the table following the titles and any table titles. This can be controlled by using the TABNUM subcommand

```
TABNUM AFTER TITLES,    or
TABNUM AFTER TABLE.TITLES,
```

The starting table number may be supplied as either an identifier or a subcommand. The table number is always included in the contents but may be excluded from the table itself by using the NO TABLE.NUMBER subcommand.

```
SURVEY Paceset, LABELS 'Paceset.lab'
       CONTENTS 'Paceset.con' NEW;

TABLE.NUMBER 5,  NO TABLE.NUMBER,
BAN Sex, STUB Age;
$
```

If the subcommand form is used, it is important to provide the table number first before a NO TABLE.NUMBER request. If you want table numbers and do request a table of contents, the table number must be supplied as a subcommand.

If you have several SURVEY commands in the same run, the table number is remembered and carried across commands unless you reset it specifically in each command. This feature when used with .RPAGE. in the titles helps when building a table of contents.

10.6 Table Titles

Every table, where a table is defined as a single stub variable, a GROUP.STUB collection of variables or a multiple response stub, may have a multi-line title. This title is used in the table of contents and it also prints below any of the 9 possible top titles. These table titles are associated with the stub variable and, as long as there are no changes in the banner variables, layouts and options, do not require additional passes of the data file.

```
TABLE.TITLE 'Ideal family size was asked only of young women',
STUB Family.Size,

TABLE.TITLE 'Stores in the East are all large super markets'
            'Stores in the West are both large and small',
GR.STUBS Q21 TO Q25,
```

The title precedes the STUB subcommand and applies to a single table.

```
TABLE.TITLE 'Political Attitude Scales',
STUB Q403 Q405,
TABLE.TITLES 'Political Party Activities',
STUB Q513a,
```

Here, variable Q403 has the title printed on each page and included in any table of contents. Q405 has no title. Q513a has a title. All three stub variables are processed in the same pass of the data because they have the same banners and table layouts. If you wish to provide titles for all the stub variables, each one must be entered separately following its own table title. For example:

```
TABLE.TITLE 'Political Attitude Scales',
STUB Q403
TABLE.TITLE 'Party Preference',
STUB Q405,
TABLE.TITLES 'Political Party Activities',
STUB Q513a,
```

Regular TITLES apply to all of the tables until they are changed or reset. TABLE.TITLES apply to a single table only. The variable names or extended labels are used in the table of contents for any table that has no table

title. Table titles are particularly useful for identifying multiple response and group stub tables, tables in which the individual variable names or extended labels do not adequately describe the group of variables as a whole.

Figure 10.3 Table of Contents With Table Titles

The commands

```
SURVEY Paceset,  LABELS 'paceset.lab',
       CONTENTS 'paceset.toc' NEW;

   BANNER Own,
   TABLE.TITLE
      'July 1994 Ownership for Age groups',
   STUB Age,

   TABLE.TITLE
      'Ownership for Income groups: Means of Income',
   STUB Income.group,
   MEANS Age Income;

   NO MEANS,
   TABLE.TITLE
      'Multiple response asked about ownership of'
      'VCR CD Portable Phones and Answering Machines',
   MR.STUB VCR to Ans.Mach;
$

SURVEY Paceset, CONTENTS 'Paceset.toc' OLD DISPLAY;
   TITLES T1 LEFT '.date.';
```

```
 The Table of Contents


 Mon Aug  8  1994


  Page    Table   Contents


    1        1   July 1994 Ownership for Age groups

    2        2   Ownership for Income groups: Means of Income

    3        3   Multiple response asked about ownership of
                 VCR CD Portable Phones and Answering Machines
```

Table titles can be centered over the table, right justified, left justified (assumed) and turned off for printing in the table while still appearing in the contents. The subcommands are;

```
LEFT JUSTIFY TABLE.TITLES,
RIGHT JUSTIFEY TABLE.TITLES,
```

```
CENTER TABLE.TITLES,
NO TABLE.TITLES,
```

NO TABLE.TITLES turns off the printing of the titles in the table proper, but it does not turn off the use of the table titles in the table of contents.

There is no limit to the number of table titles that you can supply. All table titles that are defined use the same justification. Unlike the standard TITLES, a change does not require a new pass of the data. TABLE.TITLES can be used in any SURVEY command whether or not there is a table of contents.

10.7 TABLES FOR SUBGROUPS

There are two different but easy ways to produce a large number of tables for a series of subgroups. The first uses the BY identifier in the SURVEY command. The file must be in sort order on the variables that define the subgroups. The identifier "BY" is followed by the names of 1 to 15 variables which determine by group membership. These variables may be either numeric or character. With BY, a single SURVEY command can produce, for example, tables for all the states in the USA or for all major cities within the states.

```
SORT Bigsur,   BY City State,  OUT Bigsur $
SURVEY Bigsur, BY City State, ... ;
```

The second method is to incorporate the SURVEY into a P-STAT MACRO and use the SUBFILES command. This may seem like a bit more work but it provides slightly more control over the labelling, it permits medians on an external variable, a feature that is not available when using the BY identifier, and it can be used when other P-STAT commands also reference the same subgroups. When BY is used the file must be in sort order on the BY variables. When SUBFILES are used, the file need not be sorted. The major differences between the two methods are:

Feature	BY Identifier	SUBFILES
Number of commands	1 - SURVEY only	Unlimited
BY Identification	Required	Flexible, using TITLES
External median	Not Permitted	Permitted
Overall totals	Assumed	Not available
Missing BY group	Assumed	Not available
Value labels for BY variables	Automatic	Must be programmed
Sort	Required	Not required

10.8 The BY Identifier

Figure 10.4 shows the sort step and SURVEY command necessary to produce tables first for males and then for females using a single variable from the Paceset data. When there are multiple stub variables, all the tables are printed for the first BY group, followed by all the tables for the second BY group. Missing data on the BY variables is also considered a group. If you do not wish to see the missing BY group, it is easily removed with the P-STAT Programming Language (PPL):

```
SURVEY Paceset ( IF Sex MISSING, DELETE ),
    LABELS 'paceset.lab',   BY Sex;
```

Usually the BY identifier produces an extra table containing the overall totals for all by groups. NO OVERALL.TOTALS causes this table to be omitted from the printout.

Figure 10.4 SURVEY: Using the BY Identifier to Loop Through Surveys

```
SORT Paceset,  BY Sex,  OUT Paceset $

SURVEY Paceset [ IF Sex MISSING, DELETE ],
    LABELS 'paceset.lab',   BY Sex;
    TITLES '.DATE.',
    BANNER Num.TV, STUB Own ,
   NO MISSING,    NO SUMMARY,   NO OVERALL.TOTALS;

                  Mon Aug  8  1994

BY:  Sex = Male

                    == Number of television sets in ==
                    ==          your home:          ==

              Total                                 5 or
             Sample      1      2      3      4    more

Total            39      4     18     11      4      2
Sample        100.0% 100.0% 100.0% 100.0% 100.0% 100.0%

Respondent ownership of electronic items

no                5      -      3      1      1      -
               12.8%         16.7%   9.1%  25.0%

yes              34      4     15     10      3      2
               87.2% 100.0%  83.3%  90.9%  75.0% 100.0%

                  Mon Aug  8  1994

BY:  Sex = Female

                    == Number of television sets in ==
                    ==          your home:          ==

              Total                                 5 or
             Sample      1      2      3      4    more

Total            40      2      4      7     22      5
Sample        100.0% 100.0% 100.0% 100.0% 100.0% 100.0%

Respondent ownership of electronic items

no                6      1      -      2      2      1
               15.0%  50.0%         28.6%   9.1%  20.0%

yes              34      1      4      5     20      4
               85.0%  50.0% 100.0%  71.4%  90.9%  80.0%
```

Each table is labelled with the values of the BY group variables, including the text “BY:”, the variable names and the values. When there is a labels file, the extended label is used instead of the variable name, and value labels are used instead of the numeric codes. The labelling of the BY groups cannot be turned off but it can be changed by providing a SURVEY.LABELS entry in the labels file with an alternate value for the numeric code 210.

```
SURVEY.LABELS (210) 'Sorted Groups' /
```

SURVEY.LABELS is described in the chapter “SURVEY: Enhancing Layout and Appearance”.

10.9 Using SUBFILES

SUBFILES has two very distinct advantages over the BY capability. First, the file need not be sorted on the BY variables. Second, SUBFILES is always called from a macro and it is not limited to a single command. In a SUBFILES loop you can have any number of commands in addition to the SURVEY command. When you have a multi-command situation, SUBFILES is the solution of choice. Figure 10.5 shows the commands needed for a simple macro with a SUBFILES loop. Figure 10.6 shows the RUN command used to execute the macro and the resulting printout.

The labelling of the BY levels is done by using a TITLES command with a scratch variable. When SUBFILES is in effect, a scratch variable is created for each of the BY variables.

```
SUBFILES work, BY Sex;
```

causes a scratch variable named “#Sex” to be created. It contains the current value of the BY variable. Sex is a numeric variable and SUBFILES does not support labels for the BY variable levels. This problem is solved by first using the SUBSTITUTE.VL command to change Sex from numeric to character with values taken from the labels file.

MACROS with SUBFILES should also be considered when the survey is done on a regular basis or when there are many similar tables to be produced with different BY groups. The macro in Figure 10.5 can be used for different files and variables without change.

```
RUN Surloop, Paceset Pace95 $
Run Surloop, Sex Age $
```

Figure 10.5 MACRO With SUBFILES: Looping Through the Surveys

```
MACRO Surloop $

   SUBSTITUTE.VL  Paceset, OUT work,
          VARS Sex,  LABELS 'Paceset.lab' $

   SUBFILES work,  BY Sex $

      TITLES T1 '.DATE.',
             T3 LEFT 'BY: Sex = #sex' $

      SURVEY SUBFILE, LABELS 'Paceset.lab', TITLES;

         BANNER Num.TV,  STUB Own,
         NO MISSING,  NO SUMMARY; $

   ENDSUBFILES $

ENDMACRO $
```

Figure 10.6 The RUN Command and the Printout

```
    RUN Surloop $

                    Mon Aug  8  2008


 BY: Sex = Male

                      == Number of television sets in ==
                      ==          your home:          ==

                 Total                                 5 or
                Sample      1      2      3      4     more

 Total             39      4     18     11      4      2
 Sample         100.0% 100.0% 100.0% 100.0% 100.0% 100.0%

 Respondent ownership of electronic items

 no                 5      -      3      1      1      -
                 12.8%         16.7%   9.1%  25.0%

 yes               34      4     15     10      3      2
                 87.2% 100.0%  83.3%  90.9%  75.0% 100.0%



                    Mon Aug  8  1994

 BY: Sex = Female

                      == Number of television sets in ==
                      ==          your home:          ==

                 Total                                 5 or
                Sample      1      2      3      4     more

 Total             40      2      4      7     22      5
 Sample         100.0% 100.0% 100.0% 100.0% 100.0% 100.0%

 Respondent ownership of electronic items

 no                 6      1      -      2      2      1
                 15.0%  50.0%         28.6%   9.1%  20.0%

 yes               34      1      4      5     20      4
                 85.0%  50.0% 100.0%  71.4%  90.9%  80.0%
```

MACROS with SUBFILES should also be considered when the survey is done on a regular basis or when there are many similar tables to be produced with different BY groups. The macro in Figure 10.5 can be used for different files and variables without change. For example to change the name of the file from Paceset to Pace95:

```
RUN Surloop, Paceset Pace95 $
```

To run the macro replacing variable Sex with variable Age:

```
RUN Surloop, Sex Age $
```

However, if the macro is to be run with different files and different variables, it really should be programmed so that any file and variable names that are likely to change are unique strings flagged to indicate that substitution is expected. For example:

```
MACRO Surloop,  FLAG '&' $
   SUBSTITUTE.VL  &FILE, OUT work,
           VARS &VAR,  LABELS '&FILE.lab' $

   SUBFILES work,  BY &VAR $
```

When the macro is executed, each instance of &FILE and &VAR that is found in the macro is replaced by the argument that is supplied in the RUN command.

```
RUN Surloop,  FILE Paceset,  VAR Sex $
```

To make the Surloop macro run with proper substitution, “Sex” in the TITLES command must also be changed to “&VAR”.

The ability to substitute arguments at execution time makes the macro capability a useful tool whenever you have surveys that are repeated at regular intervals. SUBFILES adds to this capability when the same display is required for multiple subgroups. MACROS are described in detail in the P-STAT User’s Manual.

Figure 10.7 P-STAT Output File from SURVEY

```
SURVEY Paceset,  LABELS 'Paceset.Lab',
       WEIGHT Weight;
     BAN Age Own, Stub Sex, MEDIAN Income,
     BODY CONTAINS COUNTS UNWEIGHTED.N SUMS MEDIAN,
     OUT Pfile, NO PRINT;
 $
 LIST Pfile, PLACES 0, SKIP 2 $
```

Row		C1.Age	C2.Age	C3.Age	C4.own	C5.own
Variable	Contents	.1	.2	.3	.1	.2
R1 Sex 1	Counts	18	8	4	4	26
R2 Sex 2	Counts	7	15	19	6	36
R1 Sex 1	Unweighted Count	20	8	3	4	27
R2 Sex 2	Unweighted Count	8	16	15	6	34
R1 Sex 1	Sums	712022	247149	124492	111442	972221
R2 Sex 2	Sums	216718	439306	684257	160207	1229483
R1 Sex 1	Medians	36700	26900	32500	25700	34700
R2 Sex 2	Medians	31000	25300	34400	26700	32500

10.10 P-STAT SYSTEM FILE OUTPUT

Each group of tables in a SURVEY command can be saved in a P-STAT system file. The contents of the system file depend on the options that have been selected for the body of the table. There are 1 to 4 data areas which are computed and stored for each tables group that is requested. They include the counts (weighted), the unweighted counts, the sums and the medians of external variables. Summary statistics are not available in the system file output.

Figure 10.7 contains the SURVEY command and a listing of the resulting output file. The subcommand OUT requires the name for the P-STAT file to be created. There is one variable in the output file for each column in the body of the table plus 1) a column to describe the stub variable and its value and 2) a column to describe the contents of the table. The number of rows is the number of defined stub values times the number of data areas (1 to 4).

The first variable has the name “Row Variable”. It is a character variable which identifiers the stub variable involved. The second variable, named “Contents”, is also a character variable. It identifies the contents of the row. The remaining variable names all begin with the letter “C” followed by the column position, as much of the variable name as space permits, and the value associated with that column. In a nested table, the variable information is taken from the inner nest level.

The P-STAT output file always contains the counts. If NO BODY is requested when OUT is used, the body information will be calculated for use in the OUT file even though it will not appear in the printed table. The table will always print unless the NO PRINT subcommand is used.

Figure 10.8 Multiple Row and Column Variables in an Output File

```
SURVEY Paceset, LABELS 'Paceset.Lab',
       WEIGHT Weight;

    BAN Age Income.groups,  STUBS Own Sex,
      PLACES COUNT 2,
      BODY CONTAINS COUNTS UNWEIGHTED.N,
      OUT Pfile2,  NO PRINT;
$
LIST pfile2, PLACES 2,
     GAP 1,  SKIP 2 $
```

```
                                                Income Income Income
Row                          C1.Age C2.Age C3.Age   Grou   Grou   Grou
Variable Contents               .1     .2     .3     .1     .2     .3

R1 Own 1 Counts                1.73   5.92   3.79   0.96   9.18   0.
R2 Own 2 Counts               23.07  20.48  25.01   0.96  41.39  20.31

R3 Sex 1 Counts               17.89  10.98   9.06   0.    20.82   8.95
R4 Sex 2 Counts                6.91  15.42  18.73   1.93  28.74  11.36

R1 Own 1 Unweighted Count      2.00   6.00   3.00   1.00   9.00   0.
R2 Own 2 Unweighted Count     26.00  21.00  20.00   1.00  41.00  20.00

R3 Sex 1 Unweighted Count     20.00  11.00   7.00   0.    21.00  10.00
R4 Sex 2 Unweighted Count      8.00  16.00  15.00   2.00  28.00  10.00
```

The output file cannot have more variables that the maximum allowed in the version being used. Since this ranges from 500 to 3000 or more, that should not be a limitation. There are no limits to the number of rows. Figure 10.8 shows the command and the resulting output file with two banner variables and two stub variables.

10.11 PROGRAM LIMITS

P-STAT comes in several sizes: WHOPPER I through WHOPPER VII The size does not affect features, but it does affect the amount that can be processed in a single pass of the data file. There are 5 different areas which increase as the program size increases. They are:

1. Command storage space to hold the text of a single command
2. A buffer to hold the subcommand text
3. Output vectors for each page of printout
4. Label storage space
5. Common area used for the body of the table
6. Generalized work space shared by the system and the command for PPL, summary vectors, sort vectors for ranking, net and define information and much more.

The command storage area is generally large enough to handle any PPL required by the SURVEY command. WHOPPER II (the standard supported size for PC/Windows and Linux) label storage area is 1,00,000 words. This increase with each size and in WHOPPER VII 32,000,000 words are available. As an example of storage requirements, a stub variable with 100 values and labels averaging 60 characters uses 1700 words of the label storage area.

There is a buffer which is used by the language parser to hold the subcommands as they are read. This buffer determines the maximum number of arguments that can be processed in a single subcommand. This, therefore, limits the number of variables that can be referenced in a single BANNER or STUB subcommand. This starts at 1600 words for WHOPPER II and increases to 6,400 words in WHOPPER VII. The actual number of variable names is somewhat less because the keywords in the given subcommand are also held in this array.

There is no limit to the number of columns (banner points) in a table as long as the body of the table and the summary vectors can fit in the internal work areas. There is a limit to the number of columns that can print on a single page. WHOPPER II, permits 200 columns plus the row totals on a single page. This increases by 100 for each WHOPPER size until WHOPPER VII permits 700 columns per page. The OUTPUT.WIDTH setting must also be large enough to store the print characters. The maximum output width is 5000 characters per page.

The common area is 400,000 words in WHOPPER II. It increases to 10,000,000 words in WHOPPER VII. This space is used to hold the body of the table. Since double precision storage is assumed, the available space for the table cells is half this amount. If the command requests external means or medians for banner variables or in the cells of the table, this amount is again cut in half. In the worst case, using a WHOPPER II version of P-STAT with a survey command and cell means there are 50,000 words available for the cells of the tables. This is enough space to hold 2000 rows of table information with 25 banner points.

DEFINE.NETS are also stored in the common area. Each DEFINE.NET adds 1 to the number of rows used by a multiple response stub variable. A given *single* table must be able to fit in the available common area. If you define a series of tables and they cannot all fit in the common area, the program does as many tables as it can in a single pass of the data file and then continues rereading the data and processing the tables until they are all complete.

The generalized work space is shared by P-STAT's executive routines and by individual commands. All of the information about open files and PPL is stored in the lower part of the workspace. The SURVEY command gets what is left. WHOPPER II has 1,000,000 words and WHOPPER VII has 32,000,000 words of work space.

A typical survey may request as many as 100 double precision vectors from this work area. A SURVEY command with 10 banner points and 20 stub variables totaling 100 rows requires 210 (20 plus 1 for the row label

multiplied by 10) double precision locations to store each of the summary vectors. Summary vectors are required for the total counts, the good counts, each of the missing values, the responses, sums and sums squared for the standard deviations and many more. If there is not enough space to hold all the summary vectors for the tables in a single pass of the data file, multiple passes are made.

The use of medians on external variables makes very heavy use of the common area. The larger the area, the more efficient the processing of medians. There are enhanced versions of WHOPPER with yet larger spaces available for special projects. If you are finding space a problem, call P-STAT or email sales@pstat.com.

10.12 Limitations of Specific Features

There are some combinations of features that are not supported.:

1. BY and MEDIAN on an external variable
2. A table with a multiple response stub variable and a multiple response banner variable cannot also have a regular banner variable if the CROSS.COMPARE subcommand is used.
3. The TRANSPOSE subcommand cannot be used when the STUB variable plus the row total has more rows than the maximum number of print positions. The TRANSPOSE subcommand cannot be used with external medians.
4. Cell means and means of a variable other than the stub variable cannot be used with COUNT and MQ.STUBS. (Tables which compute totals rather than frequencies.)

10.13 CHARACTER DATA

The SURVEY command accepts character variables for the rows (STUBS) and columns (BANNERS) with some restrictions. If the character variable has values with more than 24 characters,the data cannot accurately be mapped to integers. When this occurs:

1. All the values for those variables is set to missing 3.
2. A list of the problem variables is printed at the beginning of the output.

This list of problem variables can be omitted if the verbosity level is set to 1.

Character variables with values that can be mapped are saved, sorted, and assigned numbers which are based on their sort position in the collating sequence. The 24 characters are used as the labels for the assigned numeric codes.

The SURVEY command reads the data to get the actual ranges of all the variables and to map *ALL* character variables to numeric codes. Mapping the character variables takes both time and space. Therefore, it is a very good idea to drop any character data which you do not need in a given SURVEY command.

```
SURVEY Myfile [ DROP Country State ],
SURVEY Myfile [ DROP .CHARACTER. ],
```

The use of ".CHARACTER." is an easy way to drop all character variables from the file as it is passed to the SURVEY command.

If you do have long values and you wish to use them in the SURVEY command, use the MAP command as the first step to create a new data file and a new labels file. Run the SURVEY command using the new data file and both the original labels file and the new labels file. MAP can be used to recode sparse numeric variables and character variables with values up to 78 characters long. If you use MAP without providing a list of variables it will map ALL of the character variables in that file.

```
MAP MyFile, NEW.LABELS New.lab $
SURVEY NewFile, LABELS Orig.lab New.lab;
```

Because the assigned numeric values are data dependent, character variables can present problems when using features such as DEFINE which are based on value or position. Consider a 3 case file with 2 variables;

```
              Assigned                          Assigned
    Visited   Number      Wished.to.visit       Number

    Germany   2           England               3
    England   1           Australia             1
    Spain     3           China                 2
```

A define such as:

```
DEFINE '2 countries' Visited Wished.to.visit 1 2,
```

will include England in the subtotal for variable Visited but will not include it for the variable Wished.to.visit. Defines can only be constructed when you know the actual character values for a given variable and their sort order.

If the two variables are used in a multiple response stub, the assigned numeric values for the combined variables in the example above start with a 1 for China and end with a 5 for Spain. SURVEY handles the character values properly in a multiple response situation. Every data set and every subset of the file can have a different mapping of the values. Using DEFINE with multiple response character values can only be done if you first examine all the values for the multiple response group. A good way to do this is with the COUNT command.

Figure 10.9 COUNT command: An Aid in Constructing Defines

```
COUNT World,
      COMBINE 'countries' Visited 2,
      NONE,
      VALUES
      NO LIST,
      OUT Work $
LIST WORK, N $

      value

 1    Australia
 2    China
 3    England
 4    Germany
 5    Spain
```

Figure 10.9 illustrates the command and the printout that is produced. Counts are requested for the combination of the two character variables. With this information it is much easier to write appropriate defines for the SURVEY command.

```
DEFINE 'European Countries' Visited 3 to 5,
MR.STUB Visited Wished.to.visit,
```

See Figure 10.18 for an example of a macro that generates the defines automatically from the COUNT command output and a data file with information about the contents of the defines.

10.14 The MAP Command

If the character fields are longer than 24 characters or if a group of character variables will be used as a single multiple response stub variable, the easiest way to handle character data is to use the MAP command. MAP creates a new output file with the character data recoded so that all the variables in a multiple response group are mapped to the same numeric values. The MAP command will map up to 78 characters of a character variable and will use the characters to create an appropriate labels file.

Figure 10.10 The MAP Command

```
File Banks

  Id  Checking              Savings               Business

   1  First National Bank   First National Bank   -

   2  Intermediate Trust    Southern Building     City and Country
      Co                    and Trust             Loan Co

   3  Federal Savings &     Southern Building     Southern Building
      Loan Co.              and Trust             and Trust
```

```
MAP Banks,  OUT Bank.map,
    NEW.LABELS 'Bank.lab',
    VAR Checking Savings Business $
```

```
File Bank.map

Checking  Savings  Business

       2        1         -
       3        2         1
       1        2         2
```

```
File Bank.lab

Checking
(1) 'Federal Savings & Loan Co.'
(2) 'First National Bank'
(3) 'Intermediate Trust Co'
/
Savings
(1) 'First National Bank'
(2) 'Southern Building and Trust'
/
Business
(1) 'City and Country Loan Co'
(2) 'Southern Building and Trust'
/
```

Figure 10.10 illustrates the use of the MAP command with three character variables. Each variable is mapped separately. The output file Bank.map contains the mapped values. File Bank.lab contains the original character values structured in label file format. First National Bank has a value of 2 on variable Checking and a value of 1 on Variable Savings. This is just fine as long as the three variables are used separately. However, if you wish to used them as a multiple response variable in SURVEY, the MAP command can be instructed to handle a list of variables as a multiple response group.

Figure 10.11 contains a MAP command that is appropriate if all three of these variables are to be recoded with the same values. The values of all the variables in the group are combined and sorted so that the value that is first alphabetically is recoded as a 1. In Figure 10.11 the values for variable Checking are recoded into 2, 3, and 4 rather than the 1, 2, and 3 that are the result in Figure 10.10 with individual recodes.

Figure 10.11 MAP With Multiple Response Groups

```
   MAP Banks,  OUT Bank2map,
       NEW.LABELS 'Bank2.lab',
       MR.GROUPS 'banks' Checking TO Business$


File Bank2map

 id  Checking  Savings  Business

  1         3        3         -
  2         4        5         1
  3         2        5         5



File Bank2.lab

Checking
Savings
Business
(1) 'City and Country Loan Co'
(2) 'Federal Savings & Loan Co.'
(3) 'First National Bank'
(4) 'Intermediate Trust Co'
(5) 'Southern Building and Trust'
/
```

10.15 SPARSE DATA VALUES

Most surveys are composed of integer data and the range of the variables is often very small. For this reason the SURVEY command assumes that the values in the rows and columns are consecutive integers. The easiest and fastest way to access this type of data is to allocate space for the entire possible matrix of rows and columns. These assumptions may not be appropriate for some variables. Any variable that has a very large range but only a small number of unique values can use up far more of the work areas than it actually needs.

Consider a variable such as an occupation code with numbers from 1 to 999. However, in any single survey, there may be no more than 40 or 50 different occupations represented. A single table with ten banner points uses 9,900 words of the common area when no more than 500 are actually required.

Figure 10.12 MAP of Sparse Data and Associated labels

File Store

```
   Best     Best
Veggies  Service  Cleanest   Sex   Age

   2066     3234       ---     1     2
    ---     1345      3056     2     1
   2123     9021      1345     1     3
   1345     3056       ---     2     2
   9021     2066       ---     2     3
   2066      ---      9021     1     1
```

File Store.lab

```
    Best.Veggies
    Best.Service
    Cleanest
     (1345) Bob's Grocery
     (2066) Ewing P&Q
     (2123) Freehold Apex
     (3056) Hopewell Apex
     (3234) Hopewell Market
     (9021) Piperville P&Q /
```

```
MAP Stores,  OUT Store2,   REPORT,
    OLD.LABELS 'store.lab',   NEW.LABELS 'store2.lab',
    MR.GROUPS 'Stores' Best.Veggies to Cleanest  $
```

```
                              number
         group  old      new      of
name      size  value  value   cases

Stores       3  1345       1       3
                2066       2       3
                2123       3       1
                3056       4       2
                3234       5       1
                9021       6       3
```

FILE Store2.lab

```
  Best.Veggies
  Best.Service
  Cleanest
  (1) Bob's Grocery
  (2) Ewing P&Q
  (3) Freehold Apex
  (4) Hopewell Apex
  (5) Hopewell Market
  (6) Piperville P&Q
```

If the survey is a one time affair, the best way to handle sparse data is to recode it at the very beginning into integers with a small range. If the data are already entered, the MAP command is the easiest way to do this type of a recode.

Figure 10.12 contains a small data set with just such sparse values. The data used to illustrate the problem and the possible solutions are from a survey of grocery stores across the nation. Customers were asked a few simple questions about the reasons why they shopped at a particular market. The problem variable in such a survey is the store identification. When there are thousands of possible stores, a numbering system is required that gives each store its own identification number that can be used year after year as the survey is repeated.

Figure 10.13 DEFINE with Multiple Response Variables

```
   Report from the MAP command

                                  number
            group  old      new      of
     name    size  value  value   cases

     Stores     3  1345       1       3
                   2066       2       3
                   2123       3       1
                   3056       4       2
                   3234       5       1
                   9021       6       3


SURVEY Stores ( GEN Store = 0).  LABELS 'store.lab';
   DEFINE 'Summers County' best.veggies 2 3,
   DEFINE 'Winters County' best.veggies 4 5,
   DEFINE 'Others' best.veggies 1 6,

   PERCENTS BASE RESPONSES,
   LAYOUT LABELS SUBTOTALS,
   ASSIGN Store Best.Veggies TO Cleanest,
   MR.STUB Best.veggies TO Cleanest,
   BAN Age Sex, GAP 2;

                      ======== age =======   ==== sex ====

               Total    Under  31 to 61 and
              Sample       30     60  older     Male Female

Summers            4        1      1      2        3      1
County          30.8%    25.0%  25.0%  40.0%    42.9%  16.7%
-------

Winters            3        1      2      -        1      2
County          23.1%    25.0%  50.0%           14.3%  33.3%
-------

Others             6        2      1      3        3      3
------          46.2%    50.0%  25.0%  60.0%    42.9%  50.0%
```

The numbers cause one problem. The associated labels cause another problem. The SURVEY command has no capability for compressing the storage of sparse numeric values. The MAP command handles sparse numeric variables very nicely. Each sparse variable is recoded so that the lowest observed value becomes a “1”. The next lowest value becomes a “2”. The highest value has a code which is the number of unique values that are found for that variable in that file. The MAP command can be used with multiple response groups of sparse numeric variables in the same way that it can be used with character variables.

Figure 10.14 Automating the Creation of the Defines

1. File define.dat containing codes and the text of define labels.
   If the labels are very long, they can be entered on a separate line.

```
1345  'Other'
9021  'Other'
2066  'Ewing and Freehold'
2123  'Ewing and Freehold'
3002  'Mercer County'
3056  'Mercer County'
3222  'Mercer County'
3234  'Mercer County'
3888  'Mercer County'
```

2. RUN command for the macro to produce usable defines
   given the data in the P-STAT file Stores.

```
RUN Make.defines,
    FILE  Stores ( KEEP Best.Veggies TO Cleanest ),
    VAR1  Best.Veggies,
    VAR2  Cleanest,
    NUM 3 $
```

3. **define.def**, the file created by the macro

```
DEFINE 'Ewing and Freehold'
 Best.Veggies TO Cleanest
2 3 ,
DEFINE 'Mercer County'
 Best.Veggies TO Cleanest
4 5 ,
DEFINE 'Other'
 Best.Veggies TO Cleanest
1 6 ,
;
```

4. SURVEY command using the defines in a DEF file

```
SURVEY Stores ( GEN store = 0 ),
       LABELS 'store.lab',
       DEF    'define.def';
  ASSIGN store Best.Veggies TO Cleanest,
  MR.STUB Best.Veggies TO Cleanest, BAN Sex;
```

The new P-STAT system file and the new labels file created by the MAP command can now be used in the SURVEY command without using an excessive amount of the storage areas. Because the three variables with similar codes have been recoded as an MR.GROUP, they can be used in SURVEY as single stubs and banners or as multiple response stub and banner groups.

You cannot construct defines for character variables or mapped numeric variables unless you know the mapped numeric codes. The MAP report contains the information needed to determine the recoded position of each value in the mapped variable. Figure 10.12 includes the report from the MAP command. Figure 10.13 repeats that report and includes a SURVEY with DEFINES that are based on the MAP report.

Figure 10.15 MACRO To Create Appropriate Defines Automatically

```
MACRO Make.defines, flag '&' $


MAP &FILE, MR.GROUPS 'dummy' &var1 TO &var2,
   VAR &var1 TO &var2,  NO LIST,
   REPORT work$

MAKE workd, VARS old.value:c text:c60, FILE 'define.dat',
     DELIMITER BLANK $

LOOKUP
  work ( KEEP old.value new.value ),
  TABLE workd,  NO FILL,
  OUT   workx $

SORT workx, BY text, OUT workx $

PRINTER.SETTINGS 'define.def', OW 80, NO ECHO,
  PAGE.CHARACTER ' '  $

TEXT.WRITER workx [ IF FIRST ( text ) GEN #n = 0,
   PUT @NEXT <<DEFINE '>> text  <<'>>
       @NEXT << &var1 TO &var2 >> @NEXT;

   INCREASE #n;
   PUT new.value << >> ;
   IF #n GT 10, SET #n = 0, PUT @NEXT;

   IF LAST ( text ) PUT <<,>>;
   IF LAST ( .FILE. ) PUT @NEXT <<;>> ],

PR 'define.def', STREAM $

CLOSE 'define.def' $

MACEND $
```

A production environment is more efficient when routine steps are automated. In addition, automation improves accuracy. The MAP report can be saved in a P-STAT system file which can then be used as input to a MACRO designed to produce the DEFINE statements automatically. This macro requires a control file which links original values with the appropriate subtotal. The advantage of this approach is that the same file of definitions can be used with different data files which result in different mappings. In Figure 10.14 file “define.dat” contains the codes and the subtotal text.

Each record in the define control file contains 2 fields. The first is the original (pre MAP) code. The second field is the labels text. (If the label is very long, it may be placed on a separate record.) This file is used as input to a free format MAKE command.

Figure 10.16 Automating DEFINE: Macro Output

File workx, the LOOKUP output file

```
old      new
value  value  text

2066       2  Ewing and Freehold
2123       3  Ewing and Freehold
3056       4  Mercer County
3234       5  Mercer County
1345       1  Other
9021       6  Other
```

File **define.def**

```
DEFINE 'Ewing and Freehold'
 best.veggies TO cleanest
2 3 ,
DEFINE 'Mercer County'
 best.veggies TO cleanest
4 5 ,
DEFINE 'Other'
 best.veggies TO cleanest
1 6 ,
;
```

SURVEY command to use the definitions

```
SURVEY Stores, LABELS Stores.lab;
INCLUDE 'define.def',
BAN ....   ,    STUB .... ;
```

The first step in the macro is to create a file with all the possible codes and the associated text that is to be used in the define. Each unique text will become a DEFINE. All the values that have the same text are included in the define. This data file is processed in the macro by a free format MAKE command.

Figure 10.15 contains the entirety of the macro. The steps in the macro are:

1. Use MAP to produce an output file with the variables combined to get the sort order that is used in a multiple response situation.

2. Create a P-STAT system file from the data set containing codes and definition text.

3. Use LOOKUP to create an output file containing only those codes that are present in the current data file.

4. Sort that file so that all the codes for a define are together.

5. Set the printer attributes and then use TEXT.WRITER to write the defines to the 'define.def' output file. This file is now ready to use in a SURVEY command with the DEF identifier or as an INCLUDE file.

Figure 10.16 shows the intermediate file, workx, created by the LOOKUP command and the contents of the output definition file. The codes 1345 and 9021 are both included in a DEFINE to be labelled 'Other'. You will notice that the file 'define.def' references codes 1 and 6 rather than codes 1345 and 9021. This reflects the mapping that is done in the macro.

10.16 The ASSIGN subcommand

Labels take up a great deal of the available work area. When a tables group is completely defined (i.e., the semicolon is processed), the labels that are needed for that group of tables are read from the labels file and stored in the labels work area. This is a finite resource. A 60 character label requires 17 words of storage. One word is needed for the value. A second word is needed for the length of the label and 15 words are needed to store the label which is packed 4 characters per word. A table which has 500 such labels requires 8500 words in the labels storage area. Even in the WHOPPER II size there are only 50,000 words normally available for label storage.

In this situation, the best solution may seem unlikely. If the numeric values, regular, sparse, or mapped from character variables are themselves converted to character format, the SURVEY command will store the character form of the number as the label. Thus given the data used in Figure 10.16, the value 1 will be given the label “1234”, the value 2 will be given the label “2066”, etc. Even with very large numbers the maximum storage required for these labels is 3 or 4 words. The ASSIGN subcommand is used to solve the problem of the getting the “real” label to print.

ASSIGN requests that the labels for one or more character variables, which have values that are really numbers, be taken from another variable in the labels file. The name of the ASSIGN variable, the variable whose labels are to be used, immediately follows the ASSIGN subcommand. This is then followed by the names of one or more variables in the file whose labels are to be replaced by the ASSIGN variable labels.

```
ASSIGN Store Best.Veggies,
```

This statement can be paraphrased as “Assign the values and labels for variable Store to the variable Best.Veggies”. The ASSIGN can be made for a single variable or for a list of variables.

```
ASSIGN Store Best.Veggies TO Cleanest,
```

The labels for an ASSIGNED variable are not read into memory but are processed directly from the labels file. This is less efficient and involves more I/O, especially when the ASSIGN variable is used to provide labels for several pseudo-character variables. However, when ASSIGN is used some tables with exceptional amounts of labelling can be processed without cutting back on the length of individual labels.

Figure 10.17 Creating Character Variables from Numeric Variables

```
MODIFY Stores
    [ DO #J USING Best.Veggies to Cleanest;
      GEN V(#j)( 'X'xxxxxxxxxxxxxx ):C = CHARACTER ( V(#));   ENDDO ;

      KEEP .NEW. .OTHERS.;
      DROP Best.Veggies TO Cleanest;

      DO #J Using X?;  RENAME v(#J) TO ( oxxxxxxxxxxxxxx );
      ENDDO;
    ],
  OUT Store2 $
```

Figure 10.18 ASSIGN with Sparse Data and Multiple Response Stub

```
SURVEY Stores ( GEN Store = 0 ), LABELS 'store.lab';
   NO MISSING, NO TOTALS,
   NO SKIP,  MARGIN 20, GAP 2
   PERCENTS BASED RESPONSES,
   BAN Age Sex
   ASSIGN  Store  Best.Veggies TO Cleanest,
   MR.STUB        Best.Veggies TO Cleanest;
```

		======== Age =======			==== Sex ====	
	Total Sample	Under 30	31 to 60	61 and older	Male	Female
Which are your preferred markets?						
Bob's Grocery	3	1	1	1	1	2
	23.1%	25.0%	25.0%	20.0%	14.3%	33.3%
Ewing P&Q	3	1	1	1	2	1
	23.1%	25.0%	25.0%	20.0%	28.6%	16.7%
Freehold Apex	1	-	-	1	1	-
	7.7%			20.0%	14.3%	
Hopewell Apex	2	1	1	-	-	2
	15.4%	25.0%	25.0%			33.3%
Hopewell Market	1	-	1	-	1	-
	7.7%		25.0%		14.3%	
Piperville P&Q	3	1	-	2	2	1
	23.1%	25.0%		40.0%	28.6%	16.7%
Base	6	2	2	2	3	3
Responses	13	4	4	5	7	6
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%

ASSIGN can be used with STUB, MR.STUB, GROUP.STUB and BANNER variables. The only requirement is that the variables be character variables that are really numbers. There is no limit to the number of ASSIGN variables that can be used in a SURVEY command.

If you plan to use this feature, you can either declare the variables as character variables when you create the file or you can make the necessary conversion using the MODIFY command and the P-STAT Programming Language (PPL). When building the file with character variables specify the width accurately (C4, for example) and enter lead zeros for smaller numbers (0034, for example).

Figure 10.17 contains a MODIFY command which takes an existing file with numeric variables and creates a new output file with the numeric variables replaced by character variables. This file can now be used in SURVEY with the ASSIGN subcommand.

Figure 10.18 shows the command and the printout using the ASSIGN subcommand to link the character variables in the Stores file with a dummy variable named Store in the labels file. The Store variable is created as the Stores file is given to the SURVEY command. It is set to 0, but the value does not matter. It is there only as a link to the variable in the labels file.

10.17 OTHER FEATURES

The rest of the chapter contains an assortment of unrelated topics, including:

1. More about dummy variables in the banner
2. Setting up runs with test data so that they will run when the real data are available
3. Some suggestions about labels
4. Additional TITLES features.

10.18 Dummy Variables in the Banner

Often a study has a series of questions which requires a yes or no response. These questions may then be coded as “1” for yes responses and zero or missing for the no responses. These variables are not multiple response variables in the usual sense. However, they can be thought of as multiple response because the respondents have had an opportunity to answer yes to all of them. When these variables are used in a banner, they can be used as individual variables or they can be collected and used as a multiple response group. The desired labelling determines which is more appropriate.

When there are no value labels provided, single value banner variables are labelled with either the variable name or with the extended variable label. If there are labels for the single values, the banners are labelled in the same way as any other banner variable. If the banner variables are a related group, it may make a better display to treat them as a multiple response banner variable. The labels for the first variable are used to label the group of variables and its extended label extends over all of the variables in the group.

Since multiple response banner variables must be coded from 1 through n where n is the number of values, it may be necessary to do a recode so that the “1” on the second variable becomes a “2” and the “1” on the third banner variable becomes a “3” and so on. The easiest way to do this is to use a DO loop with 2 scratch variables.

```
SURVEY Myfile
  [  DO #J #N USING Q33 Q44 Q55 TO Q59;
     IF V(#J) = 1,  SET V(#J) = #N,
                  F.SET V(#J) = .M3. ;
     ENDDO: ],   LABELS .....
```

The first scratch variable, “#J”. points to each of the variables in the USING variable list. The second scratch variable, “#N”, takes on the values 1, 2, 3, etc. increasing each time the loop is executed.

Figure 10.19 Floating and Stationary Titles

Floating titles adjust to the width of the table

```
 Top left title                  This is the right title

                      == Number of television sets in ==
                      ==          your home:          ==

                Total                                5 or
               Sample      1      2      3      4   more

 Top left title  This is the right title

                      ==== Sex ====

                Total
               Sample   Male Female
```

Stationary titles are formatted within the current OUTPUT WIDTH

```
 Top left title                      This is the right title

                      == Number of television sets in ==
                      ==          your home:          ==

                Total                                5 or
               Sample      1      2      3      4   more

 Top left title                      This is the right title

                      ==== Sex ====

                Total
               Sample   Male Female
```

10.19 Designing Test Runs

When you are getting ready to process a new survey, it is often useful to layout the tables and the labels before the data are even ready for processing. The goal is to be ready for the production stage and to make it as smooth as possible. The problem is that much of the layout is data dependent. If there are only two values for a stub variable, there will only be 2 rows printed. When the real data are available there may be many more values. There are two easy ways to provide data that approximate the expected real values.

1. Use ranges in the stub and banner definitions

   ```
   BAN Income ( 1 5 ), STUB Q3A ( 1 9 ),
   ```

2. Provide two cases of data, one with the lowest expected values for all the variables and one with the highest expected values for all the variables.

   ```
    1 0 1 10000 0 2 101
   ```

```
9 3 4 99999 9 5 800
```

The rows and the subtotals and nets do not usually print if there are no cases. Use FILL ROWS and FILL SUBTOTALS to force in the empty rows.

Much of the work in preparing for table production is in label preparation. Tuning the labels so that they are grammatical, descriptive and attractive on the printed page can all be done before data collection is complete. Using ranges and FILL subcommands will force all the labels to print so that you can see them as they will actually appear.

If the data are complete and you are laying out the tables you can save processing time by using only a few cases and supplying ranges to approximate the full data set.

```
SURVEY Mysur ( CASES 1 TO 100 )
```

10.20 More About Titles

TITLES are formatted to fit over each table, a space which is always less than or equal to the output width. Tables that do not have the same number of columns will have the titles formatted differently. If the titles are too wide for the table, the area is expanded as needed up to the defined output width. Thus title formatting floats as the table sizes change.

If you have a series of tables of different widths and you are using wide titles, it is possible that the title formatting on the narrower tables will cause some of the titles to run together. If the subcommand STATIONARY TITLES is used all the titles are formatted using the defined output width rather than the actual table sizes. These titles will be the same for all tables. FLOATING TITLES can be used to return the title formatting to its normal setting. Figure 10.19 shows the titles and the labels for two different banner variables, one narrow and one wide, first with floating titles, the default, and then with stationary titles with the output width set to 60.

TITLES can be joined with the JOIN subcommand which, even though it is documented in the chapter on PostScript support can be used even when PostScript is not desired.

SUMMARY

SURVEY

```
SURVEY Stores ( GEN Store = 0 ), LABELS 'store.lab',
    DEF 'Mydefs',  CONTENTS 'MyTOC' OLD DISPLAY;;

    BAN Age Sex
    ASSIGN  Store  Best.Veggies TO Cleanest,
    MR.STUB       Best.Veggies TO Cleanest;
$
```

Optional Identifiers:

BY vn vn

provides a list of variables on which the file is sorted. The tables are done for each combination of the by variable values.

CONTENTS 'fn' arg arg

provides the name of an external file to hold the table of contents. The arguments which follow the name of the external file are NEW or OLD and DISPLAY. The contents contain the page number, table number and either the variable names/extended labels or the table title.

DEF 'fn'

provides the name of an external file which contains subcommands to be executed as the SURVEY command begins.

PAGE.NUMBER nn

provides the starting page number if it is to be other than “1”. This is used in the table of contents and, if any of the titles contain the .PAGE. system variable, it is also used in the titles.

TABLE.NUMBER nn

provides the starting table number if it is other than “1”. This prints below the titles and is used in the table of contents.

Optional Subcommands:

ASSIGN vn vn vn

provides the name of a numeric variable whose labels are to be used instead of the character values in the variables which comprise the remainder of the argument list. These variables must have values that are actually numbers but which have been stored as character so that SURVEY maps them automatically into a list of smaller integers.

CONTENTS

requests that the SURVEY command finish with a table of contents. This is printed but not saved.

FLOATING TITLES

is assumed. TITLES are formatted to fit over the width of the tables. Thus the titles may be formatted differently for tables with different numbers of columns.

INCLUDE 'fn'

provides the name of an external file of subcommand text that is to be executed immediately. RESET and RESTART do not rerun the subcommands in the include file, but INCLUDE can be reused whenever necessary

OUT fn

Each group of tables in a SURVEY command can be saved in a P-STAT system file. The contents of the system file depend on the options that have been selected for the body of the table. OUT provides the name for this system file

RESET

requests that the system settings for layout, etc. be reset to the original values. RESET does not reset the stub or banner variables or the defined subtotals and nets. RESET does not reread the DEF file of subcommands.

RESET arg

where arg is one of the keywords STUB, BANNER, or SUBTOTALS. This causes an individual reset of the stub variable, the banner variable or the number of defined subtotals.

RESTART

like RESET except that it executes all the RESET commands and rereads and re-executes the subcommands in the DEF file

TABLE.NUMBER nn

changes the current table number to the numeric value that is supplied. If you wish the table number to print when you do not have a table of contents, you must use this subcommand rather than the identifier form. NO TABLE.NUMBER can be used to prevent the table number from printing on each page. If NO TABLE.NUMBER is used, the table number appears only in the table of contents.

TABLE.TITLES 'cs'

supplies a title for the next table, where table is either a STUB, or a GROUP.STUB or an MR.STUB. The title is used for a single table only. If it is present it is always used in the table of contents instead of the variable name or extended labels. NO TABLE.TITLES prevents it from being printed as an additional title on the table itself.

TABNUM AFTER TITLES/TABLE.TITLES

controls the placement of the table number. TABNUM BDFORE TABLE.TITLES can also be used.

OVERALL.TOTALS

is the assumed setting and causes a final page of printout when BY variables are used. The final table contains the totals for the entire file. NO OVERALL TOTALS can be used to prevent these final tables from printing.

STATIONARY TITLES

all titles are formatted using the defined output width. All titles for all tables will be positioned in the same way when this is used.

11
TITLES and LABELS

Titles and labels are necessary parts of a useful and attractive table. This chapter covers the TITLES command, basic labels formats, and the commands which save and check the labels.

- The TITLES command defines multiple top and bottom titles for complete identification of listings, reports and other output. The TITLES identifier may then be used in commands such as LIST, SURVEY and PLOT to cause any defined titles to print on each page of output.
- The SAVE.LABELS command accepts labels for categorical or missing values and extended labels for variable names, saving them in an external file for subsequent use. The LABELS identifier may then be used in commands such as LIST, SURVEY, TABLES, EDA and ANOVA to request that labels in the designated external file be used to replace actual values.

11.1 TITLES

Titles are defined using the P-STAT *command* TITLES:

```
TITLES  T1  'Top Title' ,
        B1  'Bottom Title'  $
```

and they are "turned on" by using the *identifier* TITLES in P-STAT commands that produce output:

```
LIST  BookFile,  TITLES  $
```

TITLES (or TITLE) may be used to define up to nine top titles and three bottom titles for each page of P-STAT output. Individual control is provided for the left, center, and right sections of each title. Titles are printed whenever a P-STAT command that has titles support both 1) incorporates the identifier TITLES and 2) issues a page change.

Top titles are always at the top of the physical page. The placement of bottom titles depends on the relation of the LINES setting to the true page size. The page size of output directed to a printer or disk file is controlled by the LINES setting. This is set to 59 at the start of a P-STAT run but can be changed with the LINES command:

```
LINES 45  $
```

by setting lines in a PRINT.PARAMETERS command, or by using LINES as a general identifier within any P-STAT command:

```
LIST  BookFile,  TITLES,  LINES 45,  PR  'LPT1'  $
```

When LINES is used as a *command*, it remains in effect for *all* subsequent commands whose output is directed to a printer or disk file. When LINES is used in PRINT.PARAMETERS, it affects only the specific print destination. When it is used as an *identifier*, it is in effect *only* for the command in which it is included and overrides any previous settings. The SCREEN *command* sets the number of lines used for output directed to the terminal:

```
SCREEN  18  $
```

Titles are controlled by the LINES setting, not by the SCREEN setting. A long printout directed to the terminal fills the screen and then waits for a carriage return before printing the next screen. Top and bottom titles print as they will when the printout is directed to a device controlled by the LINES setting.

11.2 Defining Titles

In its simplest form, TITLE or TITLES requires a character string containing the title information bounded by a pair of single or double quotes. This defines a single title:

```
TITLE  'P-STAT Initial Data Run'  $
```

It is equivalent to:

```
TITLE  T1  'P-STAT Initial Data Run'  $
```

"T1" specifies that this title is defined for the *first* line of the top of the page. When just TITLE is used, the text is assumed to be defined for the first line of titling. To provide a *second* line of title information, the identifier T2 is required:

```
TITLE  T2  '1-way FREQ —  Numeric Variables'  $
```

If the previous two titles are defined and the TITLES identifier is used in a command, both titles appear on the top two lines of each page of output:

```
                P-STAT Initial Data Run
          1-way FREQ - Numeric Variables
```

A portion of the title information may be replaced or redefined subsequently in a run. The command:

```
TITLE  T2  'Regression Predicting SAT Scores'  $
```

replaces the previously defined second line of titling as follows:

```
                P-STAT Initial Data Run
          Regression Predicting SAT Scores
```

If title two is defined when title one has *not* been defined, the first title prints as a line of blanks. However, if only the first two titles are defined, only two lines of titles print. If title one and title three are defined, three lines print. The second is a line of blanks. An individual title may be explicitly set to blank by using the keyword BLANK or BLANKS:

```
TITLE  T3  BLANK  $
```

This sets the third title to a line of blanks. Any previous contents are replaced with blanks. Note that the keyword BLANK does *not* have quotes around it. If it did, the word BLANK would print in the center of the second line of titling.

Multiple lines of titling may all be supplied in a single TITLE command:

```
TITLES  T1  'Put this title first'  ,
        T2  'Put this title second' ,
        B1  'Put this title at the bottom'  $
```

"T" followed by a number indicates which top title follows. Optional commas may be used to separate the lines of titling. "B" followed by a number indicates which bottom title follows. These titles are all centered within the line defined by the current *page width.*

Page width is set by the OUTPUT.WIDTH (OW) command and general identifier. An output width of 80 is assumed for the terminal and 132 for printers and disk files. Use OUTPUT.WIDTH as a command or identifier to reset page width:

```
LIST  BookFile,  TITLES,  OUTPUT.WIDTH 80,  PR 'LPT1'  $
```

(See the chapter in the P-STAT Manual "Utility Commands For General Use" for more information on defining page widths for the terminal and for print destinations.)

11.3 Justifying and Centering Titles

The keywords LEFT, CENTER and RIGHT are used to specify where title text is to be placed. When they are not used, the title text is centered. The command:

```
TITLE  'Top Title 1'  $
```

is equivalent to:

```
TITLE  T1  LEFT    '           ' ,
           CENTER  'Top Title 1' ,
           RIGHT   '           '  $
```

The text for a left title is left justified in the line. The text for a right title is right justified in the line. The center title is centered. If the three separate strings are longer than the length of the line, they overlap one another. LEFT may be shortened to L, RIGHT to R, and CENTER to C. Commas are optional between the left, right and center sections of titling.

Any section of any title can be replaced without disturbing other sections previously defined:

```
TITLE  T1  L  'Personnel'  C  'Males',   R  '1984-85'  $
LIST   Staff
  ( IF Sex =  'Male',     RETAIN ),  TITLES  $

TITLE  T1  C  'Females'  $
LIST   Staff
  ( IF Sex =  'Female',   RETAIN ),  TITLES  $
```

Any portion of a line of titling may also be set to blanks:

```
TITLE  T2  C  BLANKS  $
```

11.4 Making Titles with Borders Using FILL

Sections of any title that are not defined are set to blank. These two commands are equivalent:

```
TITLES  T1  C  'Chemistry 101'  $
TITLES  T1  L  BLANK   C  'Chemistry 101'   R  BLANK  $
```

The undefined sections of any title and any space between defined sections may be filled with a specified character, rather than with blanks, to create a border or divider. The identifier FILL is used:

```
TITLES  T1  L  '101'    C  'Chemistry 101'    R  '101'  FILL '=' ,
        T3  L  'Mr. Shaw'   R  'Room 143'  FILL '.'  $
```

Only a *single* character may be designated as the FILL character, and only the line of titling specified previous to the identifier FILL is filled with that character. Commas may be used to separate the left, right and center sections of titling and the FILL character definition:

```
TITLES  T1  L  '101',   C  'Chemistry 101',   R  '101',  FILL '=' ,
        T3  L  'Mr. Shaw',  R  'Room 143',  FILL '.'  $
```

The FILL character for a given line of titling remains as specified, even when the text strings for that line are changed. The FILL character must be explicitly reset to a blank:

```
TITLES  T3  FILL  ' '   $
```

Now the third line of titling just defined does not have any FILL other than blanks.

11.5 Using SHOW To Examine Titles

The identifier SHOW may be incorporated at the *end* of any TITLES command to show all currently defined titles. Figure 11.1 contains the definitions for 4 top titles and a bottom title, with the addition of the identifier SHOW:

Figure 11.1 Defining and Showing TITLES

```
  TITLES  T1  L  '101',   C  'Chemistry 101',   R  '101',  FILL '=' ,
          T3  L  'Mr. Shaw',  R  'Room 143',
          T4  C  'Class List'  FILL '-' ,
          B1     'Page .PAGE.',
          SHOW  $

TOP:
101==========================Chemistry 101==========================101

Mr. Shaw                                                      Room 143
--------------------------------Class List--------------------------------

BOTTOM:
                                    Page 0
```

This command:

```
TITLES,  SHOW  $
```

shows the same defined titles again. The SHOW identifier lets you check how any defined titles look in print. The top and bottom titles are identified separately but there is no printout between them and you cannot see the effects of the LINES setting. The SHOW identifier should be the final identifier in the TITLES command to ensure that all the titles are displayed.

11.6 Turning Titles OFF and ON

Defined titles exist, but they appear on output only when the identifier TITLES is used in specific commands that produce output. Defined titles remain in effect until:

1. replacement titles are defined,
2. the titles are explicitly turned off, or
3. the titles are reset.

TITLES may be turned off by using the keyword OFF:

```
TITLES  OFF  $
```

No titles print now, even when the TITLES identifier is used in a LIST or SURVEY command. The titles that were previously defined continue to exist; they are merely turned off. They may be turned back on by using the keyword ON:

```
TITLES  ON  $
```

All existing titles now can print, if the TITLES identifier is used in an appropriate command.

The *number* of titles that print can be selectively controlled. Specific lines of titling may be turned off or on:

```
TITLE  T5  OFF  $
```

This turns off all titles from the fifth one through the last. Titles 1 through 4 continue to print.

Defined titles may be “undefined” individually or simultaneously with the keyword RESET:

```
TITLE   T3  RESET  $     or
TITLES  RESET      $
```

Titles that have been reset no longer exist. Thus, they may not be turned back on — they have to be defined again.

11.7 Using System Variables in Titles

There are numerous P-STAT system variables that improve the usefulness of titles. When these variables are encountered in a title, the current system value is substituted. For example:

```
TITLE  T1  L  'File  .FILE.'    R  'Page  .PAGE.' ,
       T2     'Describe the run in center of Title 2' ,
       B1  R  'Note:  This is on the Bottom Right'   $
```

The name of the current P-STAT system file is substituted for .FILE., and the current page number since the command began is substituted for .PAGE.

The system variables that are particularly useful in titles are as follows:

1. .DATE. This is the date set when the current command began, in character form. (It is equivalent to .CDATE., described below.)
2. .TIME. This is the time when the current command began, in character form. (It is equivalent to .CTIME., described below.)
3. .FILE. This is the name of the current file.
4. .PAGE. This is the current page number since the *command* began.
5. .RPAGE.This is the current page number since the *run* began.

The system variables .DATE. and .TIME. may be prefaced with N, X, R or C:

```
.NDATE.                    .NTIME.
.XDATE.      .NXDATE.      .XTIME.      .NXTIME.

.RDATE.      .NRDATE.      .RTIME.      .NRTIME.
.CDATE.      .NCDATE.      .CTIME.      .NCTIME.
```

The N specifies the *numeric* form of the date or time, rather than the character form. The X specifies the *exact* date or time when the system variable is processed. The R specifies the *run* date or time — when the current run began. The C specifies the *command* date or time — when the current command began. The numeric form of exact, run and command dates or times may also be specified. The dates and times are printed as they are represented in the computer system on which P-STAT is being used.

The general identifier PAGE.NUMBER resets the system variable .PAGE. It has effect only when .PAGE. is used in defining titles and the TITLES identifier is used in a command:

```
LIST  Parts,  TITLES,  PAGE.NUMBER 8  $
```

Normally, .PAGE. is reset to 1 at the start of each command. In this example, .PAGE. is reset to 8. If .PAGE. is used in a TITLE definition, it prints with the value specified after PAGE.NUMBER. The identifier PAGE.NUMBER may be used only with the TITLES identifier and in the commands that produce output, such as LIST, SURVEY, PLOT and TABLES. The identifier RUN.PAGE.NUMBER resets .RPAGE. in a similar manner. NOTE: TITLES can also be used as a general identifier for many commands including LIST, SURVEY, PLOT and TABLES.

11.8 Using Scratch Variables in Titles

Scratch variables can be used in titles. If the scratch variable is created in a command other than the command which requests the titles, it must be created as a permanent scratch variable. If the scratch variable is created in the command which requests the titles it may be either a temporary or a permanent scratch variable.

```
PPL  (  GENERATE  ##Month:C = 'November'  ) $
```

```
TITLES T1  'ABC, Inc.  Report for the Month of ##MONTH',
       T2  'District #District.No' $

LIST ABCAll ( IF District = 12, RETAIN )
            ( GENERATE #District.NO = District ),
     TITLES $
```

The permanent scratch variable ##Month is created in a PPL command early in the run. The temporary scratch variable #District.No is created as the file is passed to the LIST command. Note that the second pound sign for the permanent scratch variable is required whenever it is used. The titles that are produced with the listing will have "November" substituted for ##Month and "12" substituted for #District.No.

11.9 PostScript Features

When PostScript is used for the printout the titles can be given different fonts. Any of 9 different fonts can be selected for the titles. The left, center and right sections of each title line must use the same font. The pieces of a title (left, right and center) can be joined by using the JOIN subcommand. The chapter "SURVEY: Enhancing Table Appearance" has a section on controlling the title fonts and on joining the titles.

11.10 LABELS

Value labels and extended variable labels contribute to the clarity of information in output. *Value* labels are used in listings and surveys in place of numeric values; extended *variable* labels are used in place of variable names. Labels of either type are read from external disk files. The SAVE.LABELS command writes label files. Alternatively, an editor or word processing package can be used outside of P-STAT to write labels in a disk file. Regardless of how the labels are written in the file, the format of the labels is the same.

Value labels are supported in the following commands:

```
ANOVA       BOXPLOT    COUNTS,     EDA           LIST
MAP         PLOT       SAS.IN      SAS.OUT       SPSS.IN
SPSS.OUT    SURVEY     TABLES      TEXT.WRITER   SUBSTITUTE.VL
```

Extended variable labels are used by:

```
BOXPLOT    LIST       MAP          PLOT         SAS.IN
SAS.OUT    SPSS.IN    SPSS.OUT     SURVEY       SUBSTITUTE.XL
```

The LABELS identifier requests that value labels replace numeric values in the current command's output, and it gives the name of the label file:

```
SURVEY  MR124,  LABELS 'MR124.lab' ;
```

The USE.XL identifier in the LIST command requests that extended variable labels replace the variable names:

```
LIST    Trout,  LABELS  'TroutLab',  USE.XL  $
```

The SURVEY command (part of the *TABS* package) automatically uses any extended variable labels in the label file — USE.XL is not necessary. This makes it possible to have a single labels file with both types of labels for surveys, but to use only value labels in listings where there is less room in column headings for extended variable labels. Each command uses different features of the labels capability.

11.11 Value Labels

Value labels are character strings that replace the numeric values of a variable. The format for value labels is that each variable name is followed by a series of *values*, which must be in parentheses, and *labels*, which may be up to 80 characters long including any bounding quotation marks. The labels for each variable end with a slash (/). A dollar sign ($) indicates the end of all labels:

```
Sex        (1) Male         (2) Female   /
Education  (1) High School  (2) College  /  $
```

The text need not be quoted unless it contains a left parenthesis or a slash. Quotes are also needed to supply a blank label or leading blanks. The quotes may be single quotes, double quotes or angle brackets. You may not include the chosen delimiter as part of the text. The following examples are all acceptable.

```
Q1  (1) "it's ok" (2) <<it's "so-so">> (3) "it's <= 33" (M1) << >> /
```

Each of the following labels is incorrect and will cause an error message:

```
Q1  (1) 'it's ok' (2) it's "so-so"   (3) <it's <= 33> (M1)     /
```

The angle brackets are particularly useful when proofreading labels text. It is easier to see that there is a pair of matching right brackets for each pair of left brackets than it is to see the matching quotes. It is also easier for P-STAT to detect and provide error messages for the angle brackets.

Labels can be provided for integers, missing values, real numbers, and for character values. However, most commands, including the SURVEY command, only use labels for integers and missing values.

```
Q1 (1) yes (2) not (M1) No Answer (M2) Not Applicable (M3) Don't know
```

M1, M2, and M3 represent the three system missing values.

Character values which are enclosed in parentheses must also be enclosed in quotes or angle brackets and may be no more than 78 characters long:

```
Gender ('m') male ('f') female /
```

Long character strings, whether value labels, character values or extended labels should not be broken across 80 character records. Instead multiple records should be used:

```
Q1 (1)
<<This is the label for value 1 of question Q1.It is a long label!>>
```

The match between the value in the P-STAT system file and the value in the labels file is usually case independent. This can be controlled by adding an "X" before the label:

```
State ( X'NJ') New Jersey ....
```

Only those cases with a value that exactly matches "NJ" get the New Jersey label. The value "nj', for example, will not be a match.

When there are several variables which have identical value labels, the labels need not be respecified. The variable names may be provided as a list followed by the values:

```
Quest.1   Quest.2   Quest.8  TO  Quest.12   Purchase
(1) Yes    (2) No    (3) Undecided  /
```

The same labels are used for all the variables, Quest.1, Quest.2, Quest.8 through Quest.12 and Purchase. The slash indicates the end of the value label definitions and that set of definitions becomes the most recent set of value labels.

Instead of providing a list of variables before the labels are defined, the most recent set of labels can be reused by including an “@” after the variable name or variable list. To assign the most recent set of labels to a list of variables use this format:

```
Q1A TO Q13B @ /
```

The "@" sign followed by a variable name can be used to indicate that the value labels associated with a previous variable should also be used for the current variable:

```
Quest115 @ Quest1 /
```

requests that the labels associated with variable Quest1 be used for Quest115. The Quest1 labels must have been defined or an error will occur.

Once the labels are written in an external file, they may be referenced with the identifier LABELS, followed by the name of the label file in single or double quotes:

```
LIST  Staff,   LABELS  'LabFile'  $
```

In this example, the file named Staff is listed using the labels that have been written in the external file named LabFile.

11.12 Extended Variable Labels

Extended variable labels are character strings of up to 78 characters that replace the names of variables (which are limited to 16 characters). Not all of the commands which use value labels also use extended labels, and the actual length that is used depends on the command and the situation.

The format of extended variable labels is simply a string in quotes or angle brackets immediately following one or more variable names. If there are multiple names they will share the same extended label.

```
Var1 Var2 Var3  <<extended variable label>> /
```

A slash indicates the end of the labels for a variable or variable list, and a $ indicates the end of all labels for all variables. Value labels may follow the extended variable label:

```
Variable.Name  <<extended variable label >>
  ( value ) label  ( value )  label ...          /
```

or the extended labels and variable labels may be entered separately:

```
Quest1 “How do you rate your senator's performance?” /
Quest2 “How do you rate your mayor's performance?” /

Quest1 Quest2 (1) excellent (3) mediocre (5) awful /
```

If the *value* labels for a variable are the same as those of the previous variable, you can use the “@” notation to point to the previous variable. The value labels need not be repeated:

```
Income1 <<Head of Household:  Which income category are you in?>>
(1) Under 20,000      (2) 20,000 to 30,000  (3) 30,000 to 40,000
(4) 40,000 to 50,000  (5) 50,000 to 60,000  (6) 60,000 to 70,000
(7) Over 70,000 /
Gender (1) Male (2) Female /
Income2 <<Spouse:  Which income category are you in? >> @Income1 /
```

There can be, in theory, an unlimited number of extended labels for a single variable. Each label is enclosed in quotes or paired angle brackets. Each label must be 76-78 characters or less. You can put 2 or more extended labels on a single record as long as they fit completely. Any single label should not be broken in the middle. SURVEY and SUBSTITUTE.XL are currently the only commands which makes use of these extra labels. A special feature is the <<&&>> and <<& &>> labels which are used to indicate concatenation. These labels are not printed. They serve as instructions to the command that is using the labels. Commands which do not use the extra extended labels simply ignore them when they appear in the labels file.

```
Q1
<<This is the first line of text for question Q1  The ampersands >>
<<&&>>
<<indicate that the lines should be joined with no blank between>>

<<This is more text for question Q1.  There will be>>
<<& &>>
<<a blank inserted before it is joined with this line>> /
```

There is no special limit to the number of pieces in an extended variable label. There must be a text row both before and after a joiner row. If the joined row is too wide to fit in the output width allocated for the table, it will be printed as a single row.

```
Q33
<<Short line might be used for banner>>
<<& &>>
<<This has an expansion to be used when the variable is a stub>>
<<& &>>
<<Further text will print with previous if there is room>>
<<&&>>
<< if not it will print on the next line with this text.>> /
```

11.13 Multiple Labels Files

Any command which uses labels can process multiple labels files. This makes a variety of arrangements possible depending on your needs. A single file might be organized so that each variable in the file is defined in full with its extended label followed by its value labels. A variation on this is the use of the @ varname to avoid the repetition of identical value labels. This is more economical in terms of both human and computer resources, but if the file has thousands of variables, it makes it more difficult to look at the labels for any single variable.

A second useful type of organization is to have the extended labels in one file and the value labels in a second file. If the file is heavily used by both LIST and SURVEY which support different features for the extended labels, you might have two files of extended labels. The order in which they are given to the command determines which label wins when a given variable has labels in more than one file. The rule is that the last one wins.

Another useful organization is to have a labels file for variables that are usually included in your projects. For example, a file of demographic information might have standard labels for occupation, education, etc. If a variable in the labels file is not one of the variables in the P-STAT system file it is ignored. If a standard variable is recoded for a particular project, its new definition would be included in the labels for the new project.

```
List Project9,  LABELS 'demog.lab' 'Project9.lab' $
```

Any variable with value labels in both files will get the value labels from 'Project9.lab'. If the value labels are present in both files and the extended label is only present in the first file, then the extended label is taken from the first file and the value labels from the second file.

A variable may refer to a variable in another labels file for its value labels (@ prev.var) as long as the labels files are used in the proper order. If the files are not in the proper order, the premature reference will cause an error condition. It is safer to use the @varname reference only within a single labels file.

11.14 SAVE.LABELS

The SAVE.LABELS command is a utility command which makes an external disk file of value labels and/or extended variable labels. When that file is referenced in a subsequent command, the labels replace values and/or variable names in the output produced by that command. SAVE.LABELS checks the format of the labels and, when PSTAT.FILE is used to specify the corresponding P-STAT system file, reports the number of variables for which labels are provided.

SAVE.LABELS requires a name for the external file in which the labels are to be written. The name is supplied within single or double quotes:

```
SAVE.LABELS 'Psfile.lab';
```

The labels are supplied as subcommands after the command. Either value labels or extended variable labels or both types of labels may be supplied in a single SAVE.LABELS command.

After value and variable labels are written with the SAVE.LABELS command, their usage is requested by including identifiers such as LABELS and USE.XL in appropriate commands:

```
LIST Psfile,   LABELS 'Psfile.lab', USE.XL $
SURVEY Psfile, LABELS 'Psfile.lab';
```

Figure 11.2 illustrates the SAVE.LABELS command with the PSTAT.FILE identifier and a LIST command which uses the label information. The LABELS identifier: 1) gives the name of the external file of labels, and 2) requests that value labels replace the variable names. The USE.XL identifier: 1) requests that extended variable labels replace the variable names, and 2) points to the external file following LABELS as the location of the extended variable labels. Both types of labels may be supplied for some or all of the variables in a P-STAT system file, and for some or all of the values of a variable.

Figure 11.2 Saving and Using a Labels File

```
SAVE.LABELS  'LabFile', PSTAT.FILE Myfile ;

  Sex   <<Sex of Respondent>>
        (1) Male         (2) Female        /
  Education  <<Highest Schooling Completed>>
        (1) High School  (2) College       /
  Income <<Total Family Income>>             /
  Residence
        (1) Apartment (2) Single family (3) duplex    /
  $

THE REPORT WHEN PSTAT.FILE IS USED

  7 records were processed.
  The file will now be checked for errors.
  No errors were found.

  2 extended variable labels and 7 value labels
  were found for the variables in the P-STAT file.


LIST  Census,  LABELS  'Census.lab',  USE.XL  $
```

11.15 Checking Labels

After the SAVE.LABELS command writes the labels in the specified external file, the labels are checked for correct format. Missing slashes, unclosed quoted strings, and unknown characters within the parentheses are identified and reported. To turn off the automatic checking of labels, include NO CHECK in the SAVE.LABELS command:

```
SAVE.LABELS  'MeatLamb.lab',  NO CHECK ;
```

CHECK is assumed.

The P-STAT system file that corresponds to the labels file — the one with which the labels are to be used — may be input to SAVE.LABELS using the PSTAT.FILE identifier:

```
SAVE.LABELS  'MeatLamb.lab',  PSTAT.FILE  LambStat ;
```

When this is done, the number of variables for which value and/or variable labels are provided is reported.

You do not need to use the SAVE.LABELS command to enter your labels. Any editor that creates an ASCII file can be used. The one advantage of SAVE.LABELS is that it automatically checks the labels for syntax and balanced parentheses. If you have entered the labels using an external editor, the CHECK.LABELS command

may be used to check the format of one or more existing label files. Its arguments are the quoted names of the label files. The corresponding P-STAT system file may be supplied, if desired:

```
CHECK.LABELS  'MeatPork.lab',  PSTAT.FILE  PorkStat  $
```

CHECK.LABELS checks the format of labels and counts the number of variables in common between the system file and label files, the same way that SAVE.LABELS does. You should always check the format of your labels before using them in LIST, SURVEY, etc.

SUMMARY

TITLES

```
TITLE  'Division 12 Report'  $

TITLE  T1  LEFT    'File:  .FILE.'
           CENTER  'Summary Statistics'
           RIGHT   '.DATE.',
       T3  FILL    '-',     SHOW  $


TOP:
File:                    Summary Statistics              Fri Jun 24  1988

--------------------------------------------------------------------------------
```

The TITLES (or TITLE) *command* defines from one to nine top titles and from one to three bottom titles. Titles may be centered, left or right justified. Titles are used in output produced by commands such as LIST, SURVEY (in *TABS* — P-STAT's enhanced crosstabulation software), and PLOT, when the *identifier* TITLES is included in those commands. SHOW may be included as the last identifier in the TITLES command to show any defined titles. System variables for the current filename and current date may be used in titles.

Required:

TITLES 'cs'

provides a character string in quotes that contains the text of the title. At least one character string is required. Additional optional identifiers position the string. When none are used, the string is centered and used as the first title.

Optional Identifiers:

B1 to B3

specifies that a bottom title is being defined:

```
TITLE  B1  BLANK  B2  'Figure 2A'  $
```

It is followed by a number between 1 and 3 that indicates which bottom title is being defined.

BLANK

must follow a T or B identifier that points to a specific title line by number. That title line is set to blanks. (BLANKS is a synonym.)

CENTER

indicates that the text that follows is to be centered in the title. (C is an abbreviation for CENTER.)

FILL 'c'

specifies a single character to be used to fill the space in the line between defined titles. The FILL specification refers just to the line of titling preceding it. Initially, the fill character is a blank. When no strings are defined for a specific title line (as for T3 in the example at the beginning of the summary) and FILL is used, the entire title line is filled with the specified character.

LEFT

indicates that the text which follows is to be left justified in the left portion of the title line. (L is an abbreviation for LEFT.)

OFF

turns titles off. OFF may follow a specific title reference such as T4 or B2, indicating that titles from that number on are to be turned off, or it may follow TITLES directly, indicating that *all* titles are to be turned off.

ON

turns on titles which have been turned off. It may be used to apply to a specific title reference or to all titles.

RESET

cancels all defined titles. It may optionally have an argument and cancel the specified title:

```
TITLES  T3  RESET  $
```

RIGHT

indicates that the text which follows is to be right justified in the right portion of the title. (R is an abbreviation for RIGHT.)

SHOW

requests that any defined titles be shown on the terminal or current output device. The titles appear as they will when used in the output of a P-STAT command.

T1 to T9

specifies that top titles are being defined:

```
TITLE  T2  LEFT  'Dated  .DATE.'  $
```

T is followed by a number between 1 and 9 which indicates which top title is being defined.

SAVE.LABELS

```
SAVE.LABELS  'LabFile' ;

  Sex        <<Sex of Respondent>>
             (1)  Male         (2)  Female      /

  Education  'Last Schooling Completed'
             (1)  High School  (2)  College     /
  $
```

The SAVE.LABELS command specifies a name for an external file in which labels are to be written. The variable and value labels are provided at the subcommand level. The labels are checked for formatting

errors unless NO CHECK is specified. When PSTAT.FILE is used to input the P-STAT system file with which the labels are to be used, the variables in common are noted.

Extended *variable* labels replace variable names in listings and tables. Their format is:

```
Variable.Name  <<Here goes a longer variable label>>
```

V*alue* labels replace numeric values. Their format is:

```
Variable.Name  (value) label  (value) label    /
```

Value labels and extended variable labels may be up to 80 characters long including any surrounding quotes. Not all commands that support labels can handle the full 80 characters. Check the documentation for individual commands for details. Each variable name is followed by a series of values and labels with the values placed in parentheses. The value labels for each variable name must end with a slash (/). A "$" signals the end of the labels.

A label file is referenced using the LABELS identifier in commands that produce labeled output, such as LIST and SURVEY:

```
LIST  RFT,  LABELS  'RFT.lab'  $
```

Required:

SAVE.LABELS 'fn'

provides a name in quotes for an external file in which value and variable labels are to be written.

Optional Identifiers:

CHECK

requests that the label file be checked for formatting errors after it is written. CHECK is assumed unless NO CHECK is used.

PSTAT.FILE fn

specifies the P-STAT system file with which the labels are to be used. The SAVE.LABELS report notes the number of variables for which value labels and extended variable labels are provided.

CHECK.LABELS

```
CHECK.LABELS  'LabFile',  PSTAT.FILE  Quest12  $
```

The CHECK.LABELS command checks an existing label file for formatting errors. When PSTAT.FILE is used, the number of variables found in both the P-STAT system file and the labels file are included in the CHECK.LABELS report.

Required:

CHECK.LABELS 'fn'

supplies the quoted name of the external file in which value and variable labels have been written.

Optional Identifiers:

PSTAT.FILE fn

specifies the P-STAT system file with which the labels are to be used. The number of variables for which value labels and extended variable labels are provided is noted.

12
BALANCE and SAMPLE

Most researchers weight the cases in their sample so that the marginal and total counts in the tables produced by SURVEY reflect those of the population from which the sample was drawn. The SURVEY command permits weighting using the WEIGHT identifier or, for weighting specific tables only, using the COUNT subcommand.

The actual weights assigned to the cases must be calculated prior to using SURVEY. Sometimes the calculations are relatively simple ones designed to produce the desired marginal counts while maintaining the total count. Most often, however, the relative sizes of the counts in each cell also should be preserved. When this is the case, the weighting procedure is called *sample balancing*.

This first part of this chapter covers the BALANCE command which produces the appropriate weights so that a sample better represents known population parameters. The output from the BALANCE command includes all of the cases in the original file with a new weight variable which is the number calculated to produce the desired totals while maintaining as nearly as possible the relative sizes of the individual table cells.

The second part of this chapter covers the SAMPLE command. SAMPLE is used whenever you wish to produce a random subset of a larger file which matches that file in critical ways. The output of the SAMPLE command is a subset of the input file with a new weight variable added. These new weights equal either the total number of cases in the original file or the sum of a weight variable in the input file.

12.1 SAMPLE BALANCE

The BALANCE command does sample balancing so that the marginals and totals in tables match those of the population or other control group, whose marginals (*controls*) and totals are supplied. Weights for each unique combination of the balance variables — that is, for each of the cells defined by the balance variables — are computed. Output files of the actual and adjusted cell frequencies, the observed and adjusted marginals, and the input data plus the weights may be requested. The input file may be weighted initially either with the frequencies for each subgroup or with an initial case weight. A name may be supplied for the weight variable in the output file.

12.2 Background

Simple weighting, sometimes called weighting by the marginals, attempts to duplicate the control marginals and totals by calculating the cell count that would be expected in the population, given those marginal totals. The ratio of the expected cell count to the observed cell count is the weight given all cases in that cell.

For example, this table shows the observed cell counts and marginals in a sample of 100 cases:

		========= Age ========		
	Total Sample	20-30	30-40	40-50
Total Sample	100	40	35	25
Sex				
Male	47	22	16	9
Female	53	18	19	16

The control (desired) marginals are 50 and 50 for the males and females and 30, 40 and 30 for the three age groups. Based on these marginals, which maintain the total of 100 cases, the counts expected in each of the cells are calculated. For example, in the cell for males aged 20 to 30, 15 cases are expected: 50 times 30 divided by 100. The ratio of 15 (expected) to 22 (observed) is the weight given all cases in that cell: 15 divided by 22 is .6818.

The weights that are calculated this way for this data are:

Sex	Age	Wt
1	1	0.6818
1	2	1.2500
1	3	1.6667
2	1	0.8333
2	2	1.0526
2	3	0.9375

When these weights are used by the SURVEY command to weight the cases, this table is produced:

		========= Age ========		
	Total Sample	20-30	30-40	40-50
Total Sample	100	30	40	30
Sex				
Male	50	15	20	15
Female	50	15	20	15

The cell counts are now adjusted so that the marginals equal the control marginals and the total count is maintained. (For those who would like it, a macro to do weighting by the marginals is available from P-STAT.) Notice, however, that the relative size differences among the cell counts have been lost — males aged 20 to 30 was a relatively large group in the unweighted or original sample, whereas males aged 40 to 50 was a relatively small one.

Sample balancing, while attempting to match the control marginals and the desired total count, seeks to adjust the initial cell count as little as possible. The weights that are calculated by sample balancing using the BALANCE command are:

Sex	Age	Wt
1	1	0.8145
1	2	1.2550
1	3	1.3334
2	1	0.6805
2	2	1.0486
2	3	1.1141

When these weights are used by SURVEY, this table is produced:

		========= Age ========		
	Total Sample	20-30	30-40	40-50
Total Sample	100	30	40	30
Sex				

```
Male              50      18      20      12
Female            50      12      20      18
```

Sample balancing minimizes the adjustments to the cell counts as much as possible, while matching the control marginals.

The BALANCE command uses an iterative approximation to the least squares adjustment of W.E. Deming.[1] The least squares adjustment attempts to enforce the marginal controls, while minimizing the sum (over all cells) of the squared residuals (differences between the observed and adjusted cell frequencies) per unit response. In other words, the adjustment is proportional to the observed cell frequencies, in addition to the control marginals.

BALANCE iterates until the sum of the squared differences between the observed marginals and the controls is less than .1 or the maximum number of cycles has been reached. An alternate iteration threshold and/or an alternate maximum number of cycles may be specified to control the computing time. This is generally only an issue when there are very many balancing variables with many values.

Sample balancing is a *multivariate* matching of the sample to the population or control group. In BALANCE, the number of balance variables and the numbers of classes (values) of each of the balance variables are *not* limited. Rather, the number of unique combinations of the balance variables — the number of cells for frequencies — that are *non-zero* are limited. Zero or "empty" cells are not stored. This makes possible sample balancing when there is very large number of balancing cells, but many of them are empty.

The number of cells permitted depends on the size of P-STAT in use and the number of possible combinations in the control variables. Since this is very data dependent it is not possible to provide the exact number. The range goes from approximately 34,000 in Whopper1 to 300,000 in Whopper 4 and 3,000,000 in Whopper 7.

The number of cells is the product of the number of classes plus five of all the balance variables, minus any empty cells. For example, with three balance variables with these numbers of classes: Sex 2, Region 4, and Income 5, (2 + 5) * (4 + 5) * (5 + 5) or 630 cells are required when all contain data. The additional five classes provide storage for the three types of missings and miscodes. Storage must be provided for them in case they are present in the data. All cells are double precision to store large numbers accurately.

12.3 Producing Case Weights

The BALANCE command requires an input P-STAT system file of data and a second P-STAT file of the control (desired) marginals. One of the three possible output files should be requested. The simplest usage of the command is:

```
BALANCE  File,  CONTROLS  FileCon,  OUT  FileWts  $
```

Here, the output file containing all the input data plus the newly calculated weights is requested. The weights are the values of a new variable named "Weight...."

Figure 12. 1 shows two P-STAT system files that illustrate how BALANCE works — on the left is the data file and on the right is the file of control marginals. (This data is from the Deming book. The filename "Dem107" includes the page number on which the data appear.) There are two balance variables, Age and State.

This data file is weighted — each case is not a single respondent, but a count of respondents in each cell defined by the balance variables. Sometimes, even when the cases are individual respondents, counts of dollars spent or items purchased may be desired. In these circumstances, the WEIGHT identifier must be included in BALANCE to provide the name of the weight variable.

The file of controls gives the desired marginals. These marginals are sometimes called "slice totals" when there are three or more balance variables (three or more dimensions) because the marginals are the totals for the values of each balance variable summed over all values of the other balance variables — that is, the totals of "slices from the cube." A control marginal is provided for each value of each balance variable and, if desired, for each of the three types of missing values.

---

[1] Deming, W.E. (1943). *Statistical Adjustment of Data*, John Wiley & Sons, New York.

Figure 12.1 Sample Balancing: Input Files of Weighted Data and Controls

FILE Dem107:

Age	State	Count
1	1	3623
1	2	1570
1	3	1553
1	4	10538
1	5	1681
1	6	3882
2	1	781
2	2	395
2	3	419
2	4	2455
2	5	353
2	6	857
3	1	557
3	2	251
3	3	264
3	4	1706
3	5	171
3	6	544
4	1	313
4	2	155
4	3	116
4	4	1160
4	5	154
4	6	339

FILE Dem107C:

Name	Value	Count
State	**1**	**5252**
State	**2**	**2395**
State	**3**	**2432**
State	**4**	**15766**
State	**5**	**2330**
State	**6**	**5662**
Age	**1**	**22877**
Age	**2**	**5285**
Age	**3**	**3462**
Age	**4**	**2213**

The variables in the file of controls may have any names, but they must be in this order:

1. a character variable that names the balance variables,
2. a numeric variable that gives the values of the balance variables in *ascending* order (1 through n, and then missing1, missing2 and missing3, if supplied), and
3. a numeric variable that gives the control marginals, which may be *counts* or *proportions*.

When the controls are *counts*, the sum of the control marginals of any balance variable should equal the sum of the marginals of any other balance variable. This sum is the desired total, which may be the same as the observed total or larger or smaller than it to either inflate or deflate the sample size. In Figure 12.1, the sum of the State marginals equals the sum of the Age marginals. This sum (33,837) is the sample total, which will be maintained during weighting.

When all the control marginals for a balance variable are not provided, they are computed by the BALANCE command when possible. For example, if the case with the control marginal for State value 1 was missing from the file Dem107C, the value 5252 could be computed by subtraction. (The value 5252 is the sum of the other State marginals subtracted from the desired total, which is the sum of the Age marginals.) When there is not one set of complete marginals from which the desired total can be computed, the total should be input as the initial case of the controls file. Figure 12.2 I (File Dem107C) shows how this is done. The desired total is the value of the third variable in the *first* case; the other variables in that case have missing values. Now, any single missing marginals

can be computed. When more than one marginal of a balance variable is missing, the values without marginals are lumped together and a single control is computed for the grouping.

Figure 12.2 Supplying the Total and Either Counts or Proportions as Controls

I. Controls are counts —

FILE Dem107C:

Name	Value	Count
-	-	33837
State	2	2395
State	3	2432
State	4	15766
State	5	2330
State	6	5662
Age	1	22877
Age	2	5285
Age	3	3462

II. Controls are proportions —

FILE Dem107P:

Name	Value	Count
-	-	33837.000
State	1	0.155
State	2	0.071
State	3	0.072
State	4	0.466
State	5	0.069
State	6	0.167
Age	1	0.676
Age	2	0.156
Age	3	0.102
Age	4	0.065

When the controls in the CONTROLS file are *proportions* and not counts, the sum of the controls of any balance variable should equal one. BALANCE computes the control marginals by multiplying each proportion by the observed total. An inflated or deflated control total may be supplied in the first case to be used instead of the observed total. Figure 12.2 II (File Dem107P) illustrates this.

To hold the marginals of a given balance variable *constant*, that is, at the same proportions they are in the sample, values of zero should be supplied for *all* classes of that balance variable. For example, to maintain the values of Age in the same proportions to the total that they are in the data file, the values of all four Age control marginals in the controls file should be zero.

Figure 12.3 shows the BALANCE command for the Dem107 files and the report that is produced. The WEIGHT identifier is used because the input data file is weighted with frequencies for each age/state subgroup. The OUT, FREQS and MARGINALS identifiers are used to request all three possible output files. WEIGHT.NAME supplies an alternate name for the weight variable in the OUT file.

The report gives the number of unique combinations of the balance variables and the largest and smallest observed counts for any single combination. The expected and the observed sample totals are printed; here, they are the same. The number of passes and cycles is given, as is the final sum of the squared residuals. Iteration stops when the sum of the squared residuals is .1 or when the number of passes made equals three times the number of balance variables. *One* pass is as many cycles through all the cell frequencies as there are balance variables. The THRESHOLD and/or PASSES identifiers may be used to specify alternate values to control the number of iterations. Both take a single numeric argument.

Figure 12.4 shows one of the output files, the file of marginals and residuals. The observed marginals, the expected or control marginals and the adjusted marginals are included in the file. The residuals are the differences between the control marginals and the adjusted marginals. The output file of frequencies contains the actual and the adjusted cell frequencies for each combination of the values of the balance variables. The output file requested by OUT contains all the input data and the weights.

Figure 12.3 The BALANCE Command and Report

```
BALANCE  Dem107,   CONTROLS   Dem107C,  WEIGHT  Count,  OUT  Dem107W,
  FREQS  Dem107F,  MARGINALS  Dem107M,  WEIGHT.NAME   Wt.Factor  $


24 unique combinations of the balance variables were found.
24 cases were successfully processed.

10538 was the largest weighted count for any single combination.
116 was the smallest weighted count for any single combination.

Expected total population is                  33837.
Total weighted count for the sample is        33837.

2 passes were made.
Each pass contained 2 cycles except for the
final pass which contained only 1 cycle.

The final sum of the squared residuals is .004686451.


File Dem107F contains the frequencies for all unique combinations.

File Dem107M contains the observed, expected, and adjusted marginals.

File Dem107W contains the input variables plus variable Wt.Factor.
```

12.4 Using the Weighted Data

The output file from the BALANCE command that contains the newly calculated weights may be used as the input file to the SURVEY command (as well as to all the other P-STAT commands that permit weighted data). When the data file was weighted prior to the use of BALANCE, the true weight is the product of the original weight and the new weight.

Figure 12.5 illustrates this situation. PPL (the P-STAT programming language) sets the weight variable to the product of the old and new weights as the file is input to SURVEY:

```
SURVEY   Dem107W     ( SET  Wt.Factor =  Wt.Factor * Count ),
LABELS  'Dem107.Lab',  WEIGHT  Wt.Factor ;
```

The name of the weight variable is specified after the WEIGHT identifier (or, alternatively, after the COUNT subcommand).

Figure 12.4 The Output File of Adjusted Marginals

```
LIST  Dem107M  $
```

Variable	Value	Observed	Expected	Adjusted	Residual
State	1	5274	5252	5252.00	0.0000000
State	2	2371	2395	2395.00	0.
State	3	2352	2432	2432.00	0.0000000
State	4	15859	15766	15766.00	0.0000000
State	5	2359	2330	2330.00	0.0000000
State	6	5622	5662	5662.00	0.
Age	1	22847	22877	22876.95	0.0030015
Age	2	5260	5285	5285.00	0.0000082
Age	3	3493	3462	3462.03	0.0009989
Age	4	2237	2213	2213.03	0.0006779

Tables produced by SURVEY reflect the weighting in the cell counts, percentages, marginals and totals. The table in Figure 12.5 may be compared with the input files shown in Figure 12.1. The table marginals equal the controls (excluding occasional rounding errors), and the cell counts reflect both the adjustment necessary to achieve that agreement and the counts in the original cells. Adjustment of the initial counts is minimized as much as possible.

12.5 Missing and Undefined Values

Control marginals should be supplied for *all* values of the balance variables *actually observed* in the data. When controls are not supplied for some values or when controls are supplied for nonexistent values, various problem situations arise. It is wise to use the COUNTS command to get the frequencies for all values of all balance variables before building the CONTROLS file and using BALANCE.

A common situation is to supply controls for just the *good* values of the balance variables, but not for any missing values. When that happens, the weight variable is set to zero in all cases with missing values of the balance variables. Since cases with zero weights are ignored by SURVEY, those cases are excluded from the tables. When all cases, including those with missing values of the balance variables, should be included in the tables, control marginals for missing values must be included in the CONTROLS file.

When controls are not supplied for some good values of the balance variables, these *undefined* values are lumped together into an “other” category. If the total of the control marginals for that variable is less than the population total, then the controls for the “other” category are set to the difference. This category lists as missing type3 in the FREQS and MARGINALS files. If the total equals the population total, these values are treated just like missing values and they are given zero weights.

When control marginals are supplied for missing or other values of the balance variables in the CONTROLS file and no such values exist in the data file, an error results and BALANCE prints a message that convergence can not occur. An empty (zero or nonexistent) cell cannot be weighted.

Figure 12.5 Making a Table with Weighted Data

```
SURVEY   Dem107W     ( SET  Wt.Factor =  Wt.Factor * Count ),
LABELS  'Dem107.Lab',  WEIGHT  Wt.Factor ;

  BANNER  Age,  STUB  State,
  NO PERCENTS,  CELL.WIDTH 10,  NO MISSING ;
```

```
Weighted by: Wt.Factor

                          ================ Age ================

                  Total
                 Sample   7 to 13   14 & 15   16 & 17  18 to 20

Weighted
Total             33837     22877      5285      3462      2213

State

Maine              5252      3613       781       550       309

New Hampshire      2395      1588       401       251       155

Vermont            2432      1608       435       270       119

Massachusetts     15766     10492      2451      1681      1142

Rhode Island       2330      1662       350       167       151

Connecticut        5662      3914       867       543       338

Base              33837     22877      5285      3462      2213
Unweighted           24         6         6         6         6
Total
Mean               3.72      3.73      3.72      3.66      3.76
S.D.               1.55      1.57      1.54      1.54      1.49
```

To avoid problem situations, use the COUNTS command before attempting sample balance. COUNTS tallies the frequencies of all unique (good and missing) values of all variables in the input file. This list can be checked to ensure that the BALANCE controls file has an entry for each possible control value.

```
COUNTS  Dem107  (KEEP  Count  State  Age),  WEIGHT  Count,
  NONE, VALUES,  OUT  Dem107F  $
```

The COUNTS command may be weighted, if desired. (See the COUNTS chapter in "*P-STAT: Introductory Manual*" for more information.)

Figure 12.6 SAMPLE: Selecting a Given Percent of Cases

```
FILE Tsamp

Value  Group   Weight

    1      1    3.145
    2      2    2.789
    3      1    4.023
    4      2    3.086
    5      1    4.598
    6      2    2.362
    7      1    3.005
    8      2    3.482
    9      1    4.186

SAMPLE Tsamp,  PERCENT 20,  OUT Samp20 $

----------------SAMPLE completed----------------
Input file Tsamp has 9 cases and 3 variables.
All cases were available for selection.

A 20.00 percent sample was requested.
Given 9 usable cases, 1.80 cases were sought.

OUT  file Samp20 has 2 cases and 4 variables.
(Variable SAMPLE.WEIGHT has been added.)


OUTPUT FILE FROM THE FIRST EXECUTION OF SAMPLE

    FILE Samp20

                        sample
Value  Group   Weight   weight

    6      2    2.362     4.5
    9      1    4.186     4.5


OUTPUT FILE FROM THE SECOND EXECUTION OF SAMPLE


    FILE Samp20

                        sample
Value  Group   Weight   weight

    2      2    2.789     4.5
    8      2    3.482     4.5
```

12.6 SAMPLE

The SAMPLE command selects a random sample from an input file. The size of the output file can be specified as either the desired number of cases or the desired percent. The command uses random numbers whose sequence is initialized differently each time the command is invoked. It is highly likely that a different sample will be selected even though the command is re-executed without change. Figure 12.6 has a very small data file with three variables. This file is used in the following examples to illustrate SAMPLE with all possible options.

In Figure 12.6, the SAMPLE command requests a 20% sample of the entire file without considering either the group variable or the weights. The reports from running this command and the output file from two successive executions of the command are also shown. The resulting report and the sample weights are the same, but the selected cases are different.

The selected cases go into the OUT file. They are in the same order as they were in the input file. The unselected cases can be written to the optional OTHERS file. These cases are also in the same order as they were in the input file.

Figure 12.7 Random Selection of Subgroups

```
SAMPLE Tsamp,  PERCENT 33,  OUT Samp33,  BY Group $

 ---------------SAMPLE completed---------------
 Input file Tsamp has 9 cases and 3 variables.
 All cases were available for selection.

 A 33.00 percent sample was requested.
 Given 9 usable cases, 2.97 cases were sought.

 OUT  file Samp33 has 4 cases and 4 variables.
 (Variable SAMPLE.WEIGHT has been added.)

   1 BY variable was used to define groups.
   2 BY groups were found.
   2 is the most  cases selected for any BY group.
   2 is the least cases selected for any BY group.


File Samp33



                        sample
 Value  Group   Weight  weight

     1      1    3.145     2.5
     2      2    2.789     2.0
     3      1    4.023     2.5
     8      2    3.482     2.0
```

BY variables can be supplied to define subgroups within the file. If BY variables are used, each possible subgroup is sampled to the same extent as the file as a whole. For example, if 40 cases have the same values as each other on BY variables AGE and REGION, a 25% sampling will select exactly 10 of them. A group of 41 cases causes 11 to be selected because the default is to convert 10.25 into the next integer (a CEILING function). There-

fore, the number of selected cases in the sample will tend to be a few more than the requested number. If BY is not used, the entire file is treated as one group.

A SAMPLE.WEIGHT variable is added to the variables in the OUT file such that the weights of each BY group add up to their group size in the input file. The sum of the weights in the resulting sample is therefore equal to the number of usable cases in the input file.

Figure 12.7 shows the SAMPLE command, the report and the contents of the output file with a single BY variable and the desired percent set at 33. A request for 33 percent of the entire file results in an output file with 3 cases. When BY variables are used, an attempt is made to provide an output file with 33 percent of the cases in each subgroup. Here you can see the oversampling that results. To get at least 33 percent of each group requires selecting 2 cases from each group for a total of 4 cases, 1 more than the target for the file as a whole. Notice, however, the sample weights for selected cases sum properly to 9, the number of cases in the file.

12.7 Selection Criteria

The selection criteria must be supplied. It can be either a percentage or the total number of cases.

```
CASES 200,
```

This asks for 200 cases to be selected. Figure 12.8 shows the command, the report and the output file when a simple selection is made.

```
PERCENT 20,
```

Instead of using CASES, one can ask for a certain percent of the file to be selected. If BY is not used, 20% of the cases in the file will be selected. If BY is in use, 20% of the cases with non-missing BY values will be selected.

```
BY religion region,
```

Up to 15 BY variables can be provided. The order of the BY variables does not matter. They can be any mixture of numeric and character. There is no limit to the number of combinations of the BY values found in the input file. Any case with a missing value on one of the BY variables is not eligible for inclusion in the sample.

Figure 12.8 SAMPLE: Selecting Cases

```
SAMPLE Tsamp,  CASES 2,  OUT Tsamp02 $

 ----------------SAMPLE completed----------------
 Input file Tsamp02 has 9 cases and 3 variables.
 All cases were available for selection.

 A sample of 2 cases was requested.

 OUT  file work has 2 cases and 4 variables.
 (Variable SAMPLE.WEIGHT has been added.)


 File Tsamp02
                         sample
 Value  Group   Weight  weight

     3      1    4.023     4.5
     5      1    4.598     4.5
```

The power of this command lies in maintaining the proportionality of the various BY groups. Suppose we wish to draw a sample of about 200 cases from an input file of 1000 cases (a 20 percent sample) while controlling for 2 BY variables, RELIGION and REGION.

If 34 cases in the input file have values of CATHOLIC and MIDWEST on the 2 BY variables, we would like 20 percent of the 34, or 6.8 cases to be selected. 6.8 is adjusted to 7, so we will get 7 cases in the output file for this BY group. Each of the 7 will have a value on the SAMPLE.WEIGHT variable of 34/7, or 4.857 .

Figure 12.9 SAMPLE: The WEIGHT Variable

```
SAMPLE Tsamp,
       PERCENT 33,
       OUT Samp33,
       BY Group,
       WEIGHT Weight $
```

Value	Group	Weight	sample weight
2	2	2.789	5.56328
4	2	3.086	6.15572
5	1	4.598	9.92307
9	1	4.186	9.03393

12.8 Weighted Data

When a weight variable is named in the SAMPLE command, the generated sample weights are scaled to equal the sum of the input weight variable rather than the total number of cases. The number of cases is not affected. That is determined by the CASES or PERCENT identifier.

```
WEIGHT Weight.varname
```

WEIGHT is the identifier that is used to supply the name of the weight variable. Figure 12.9 illustrates A SAMPLE command using a WEIGHT variable and shows the resulting output file.

The total for the values of variable WEIGHT for all the cases in input file Tsamp is 30.676. Since we have used the WEIGHT identifier, the totals of sample.weight for the selected cases must also equal 30.676. The total of the weights for all the cases in group 1 is 18.957 and the total of the weights for the cases in group 2 is 11.719. After the cases are selected, the generated sample weights are apportioned in the same ratio as the input weights of the selected cases.

Group	Weight	% of subtotal	Sample weight	% of subtotal
2	2.789	47.4723	5.56328	47.4723
2	3.086	52.5277	6.15572	52.5277
subtotal	5.875		11.71900	
1	4.598	52 3452	9.92307	52.3452
1	4.186	47.6548	9.03393	47.6548
subtotal	8.784		18.95700	
total	14.659		30.67600	

The 5 cases in group 1 have a total of 18.957 on variable Weight. This is the target total sample weight for the selected cases in the subgroup. The 2 selected cases have a total of 8.784 on variable Weight. The weight of the first case is 4.598 or 52.3452 percent of the total for the selected cases. Thus the first case is given a sample weight that is .523453 *18.957 = 9.92301. The second case is given a sample weight that is .476548 * 18.957 = 9.03393 .

Usually the selection criterion (CASES or PERCENT) determines the number of cases that are to be selected. However, if there is a weight variable in the input file, it is possible to apply the selection to the sum of the weights rather than to the number of cases. The identifier is USE.WEIGHT. Figure 12.10 illustrates USE.WEIGHT. Here CASES 10, is used even through there are only 9 cases in the file. However, since the selection is based on the values of the WEIGHT variable, only 4 cases are selected.

Figure 12.10 SAMPLE: Using Weights to Control Selection

```
SAMPLE Tsamp,  OUT Samp10,  BY Group,   CASES 10,
       WEIGHT Weight,  USE.WEIGHTS $


----------------SAMPLE completed---------------
Input file Tsamp has 9 cases and 3 variables.
All cases were available for selection.

A sample with input weights summing to 10 was requested.

OUT  file Samp10 has 4 cases and 4 variables.
(Variable SAMPLE.WEIGHT has been added.)

  1 BY variable was used to define groups.
  2 BY groups were found.
  2 is the most  cases selected for any BY group.
  2 is the least cases selected for any BY group.


File Samp10

                          sample
Value  Group   Weight     weight

    5      1    4.598   11.46446
    6      2    2.362    4.73653
    7      1    3.005    7.49254
    8      2    3.482    6.98247
```

12.9 Controlling the Subgroup Sizes

There are 5 identifiers which can be used to control the cases that are selected. MIN.GROUP and MAX.GROUP are only used when there are BY groups. Each requires an integer argument specifying the minimum or maximum number of cases to be used in any single group. They can be used together.

Figure 12.11 shows three SAMPLE commands and the resulting output files. The first command provides no controlling information and the resulting file has 5 rows, three for group 1 and 2 for group 2. The second command specifies MIN.GROUP 3 and the resulting output file 6 rows, 3 for each group. The third command

specifies "MAX.GROUP 2", and the resulting output file contains 4 rows, 2 for each group. In all three cases the resulting sample weights total 9, the number of rows in the original file.

Figure 12.11 SAMPLE: MIN GROUP and MAX.GROUP

```
SAMPLE Tsamp, PERCENT 50, OUT Samp50, BY Group$

   FILE Samp50

                              sample
   Value  Group   Weight      weight

       1      1    3.145    1.666667
       2      2    2.789    2.000000
       3      1    4.023    1.666667
       6      2    2.362    2.000000
       7      1    3.005    1.666667


   SAMPLE Tsamp, PERCENT 50, OUT Samp50, BY Group,
          MIN.GROUP 3$

   FILE Samp50

                              sample
   Value  Group   Weight      weight

       1      1    3.145    1.666667
       2      2    2.789    1.333333
       3      1    4.023    1.666667
       4      2    3.086    1.333333
       6      2    2.362    1.333333
       9      1    4.186    1.666667


   SAMPLE Tsamp, PERCENT 50, OUT Samp50, BY Group,
          MAX.GROUP 2$

   FILE Samp50

                           sample
   Value  Group   Weight   weight


       1      1    3.145     2.5
       4      2    3.086     2.0
       5      1    4.598     2.5
       6      2    2.362     2.0
```

There are 3 identifiers which can be used with or without BY groups to control the number of cases selected. These are CEILING, ROUND, and FLOOR. CEILING (or CEIL) is the default. If the number of cases to be

selected for the file or for a BY group is fractional, the next highest integer value is used. This will, on average, produce a larger sample than the requested sample size.

When ROUND is used, .5 is added to the number of cases and the result is truncated. Thus 4.8 becomes 5, but 7.3 becomes 7. When FLOOR is used, the number of cases is truncated and the fractional part is dropped. 4.8 becomes a 4 . This will, on the average, produce a smaller sample than the requested sample size.

Figure 12.12 SAMPLE: Controls for Case Selection

```
.
     SAMPLE Tsamp,  PERCENT 50,  OUT Samp50,  BY Group,
            CEILING$
```

EXCERPT FROM THE REPORT

```
Input file Tsamp has 9 cases and 3 variables.
All cases were available for selection.

A 50.00 percent sample was requested.
Given 9 usable cases, 4.50 cases were sought.

OUT  file Samp50 has 5 cases and 4 variables.

FILE Samp50

                              sample
Value  Group   Weight         weight

    1      1    3.145       1.666667
    6      2    2.362       2.000000
    7      1    3.005       1.666667
    8      2    3.482       2.000000
    9      1    4.186       1.666667
```

```
     SAMPLE Tsamp,  PERCENT 50,  OUT Samp50,   BY Group,
          FLOOR$
```

EXCERPT FROM THE REPORT

```
OUT  file Samp50 has 4 cases and 4 variables.


FILE Samp50
                         sample
Value  Group   Weight    weight

    1      1    3.145       2.5
    2      2    2.789       2.0
    5      1    4.598       2.5
    8      2    3.482       2.0
```

12.10 Limitations

There is no limit to number of cases or, if BY is used, on the number of groups. A new variable, SAMPLE.WEIGHT, is added to the OUT file; obviously a variable with that name should not already exist. No more than 15 BY variables can be used.

Three temporary variables are used in the sampling process; XXXXX.START.LOC, XXXXX.RANDOM.NUM, and XXXXX.GROUP.SIZE. These names also should not already exist in the INPUT file, and the input file cannot exceed NV-4 variables, where NV is the maximum number of variables allowed in a P-STAT system file (3,000 or such).

When a SAMPLE command is invoked, it does some things itself and then generates as many as 10 other P-STAT commands (like SORT, twice) to do the bulk of the work. A thorough (some might say fulsome) report is issued when the command ends.

SUMMARY

BALANCE

```
BALANCE  Dem107,   CONTROLS   Dem107C,  WEIGHT  Count,
  FREQS  Dem107F,  MARGINALS  Dem107M,  OUT  Dem107.2   $
```

The BALANCE command does sample balancing so that the marginals and totals of the sample match those of a control group ("population"), whose marginals ("controls") and totals are supplied. Weights for each unique combination of the balance variables ("cells") are computed. Output files of the actual and adjusted cell frequencies, the observed and adjusted marginals and the input data plus the weights may be requested. The input file may be weighted initially, and a name may be supplied for the new weight variable.

BALANCE uses an iterative approximation to the least squares adjustment of W.E. Deming (*Statistical Adjustment of Data*, New York: John Wiley & Sons, 1943.) The least squares adjustment attempts to enforce the marginal controls, while minimizing the adjustments to the cell frequencies. The BALANCE command iterates until the sum of the squared differences between the observed marginals and the controls is less than .1 or another specified threshold, or the maximum number of cycles has been reached.

Required:

BALANCE fn

specifies the name of the input P-STAT system file of data. The file should contain all of the variables that the sample is to be balanced by, and it may contain additional variables and a weight variable. There should be a minimum of two balance variables. There is no limit to the maximum number of balance variables, although the number of cells defined by all balance variables is limited and depends on the size of P-STAT in use. The maximum on a standard PC version, for example, is 125,000.

Required Identifiers:

CONTROLS fn

specifies the name of the input P-STAT system file of expected marginals or controls. This file should have *three* variables (with any names) in this *order*: 1) a character variable containing the *names* of the balance variables, 2) a numeric variable containing the *values* ("classes") of each balance variable in *ascending order* and 3) a numeric variable contained the expected or control marginals ("slice totals"). The control marginals may be *counts* or *proportions* (decimal numbers); missing ("-") control values are not permitted. The classes are integers, followed by from one to three types of missings ("don't know", "refused", etc.), if desired.

The control marginals for *each* balance variable should sum to the same expected total or to one, if they are proportions. When all marginals are not supplied for at least one of the balance variables, the control or population total should be supplied in the *first* case of this file. The first two values should be missing and the third should be this total. When control marginals are not supplied for some classes of the balance variables, they are computed as the difference between the population total and the sum of the other classes. When the control marginals are zeros for *all* classes of a balance variable, the same proportions of the observed marginals to the observed total are used for the controls, thus holding that variable "constant". EXPECTED is a synonym for CONTROLS.

Optional Identifiers:

FREQS fn

supplies a name for and requests an output file of cell frequencies for all combinations of the balance variables. The file contains the balance variables, the observed (actual) frequencies, the adjusted (balanced or weighted) frequencies, and the cell weights.

INITIAL.WEIGHT vn

supplies the name for a weight variable when the input data are individual cases. The WEIGHT identifier is used when the data are already grouped and there is a weight variable of frequencies.

MARGINALS fn

supplies a name for and requests an output file of marginals. The file contains the classes of each balance variable, the observed, expected (control) and adjusted marginals, and the residuals (differences between the control and adjusted marginals).

OUT fn

supplies a name for the output file that contains all of the input data and the newly calculated weights. At least one output file must be requested when using BALANCE; typical this file of data and weights is the desired one. The weight variable is named "Weight...." unless the WEIGHT.NAME identifier is used to supply another name.

PASSES nn

specifies the maximum number of passes that the BALANCE command may make in computing the adjusted marginals. *One* pass is as many cycles through all the cell frequencies as there are balance variables. Normally, PASSES is set to three times the number of balance variables, and iterations stop when the sum of the squared residuals is less than or equal to the threshold value of .1 or that many passes have been made. Typically, it takes far fewer passes than the default and processing stops when the threshold value is reached. (See THRESHOLD also.)

THRESHOLD nn

gives an alternate value to use in determining when iterations should stop. THRESHOLD is set to .1 unless another value is specified. The sum of the squared residuals (differences between the control and adjusted marginals) is compared with this value and when it is less than or equal to it, the program halts. Higher values yield fewer iterations.

WEIGHT vn

provides the name of a weight variable in the input data file. Each case "counts" this value, instead of counting as one. Weights may be fractional numbers; negative and zero weights are ignored. WEIGHT is used when preliminary weights have been calculated or when the cases in the input file are grouped responses rather than single responses.

WEIGHT.NAME vn

supplies a name to be used in the output file of data and weights for the weight variable. The name should be a unique legal P-STAT name. When WEIGHT.NAME is not used, the weight variable is named "Weight....".

SAMPLE

```
SAMPLE BigFile,  BY Age Occupation Gender,  CASES 300,
   OUT Sample $
```

SAMPLE is used to obtain a random sample, which has a specified number of cases or a specified percentage of the cases, from an input file. If BY variables are specified to define subgroups in the file, each subgroup will be sampled to the same extent as it exists in the file as a whole.

Required:

SAMPLE fn

provides the name of the input file. PPL can be done as it is read. All cases are available for sampling except those with missing values on BY or WEIGHT variables.

Required Identifiers:

OUT fn

provides the name for the output file. This contains the selected cases in the same order as they occur in the input file. An extra variable SAMPLE.WEIGHT is added to the original variables.

CASES nn

provides the number of cases to be selected. Either CASES or PERCENT must be used. When you are using BY groups, the number of cases actually selected will usually be somewhat more than the number of cases requested. See identifiers CEILING, ROUND, and FLOOR.

PERCENT nn

provides the percent of cases. This must be a number between zero and 100. If BY is not in use the number of cases is that percent of the entire file. If BY is used, the number of cases is calculated for each of the BY subgroups. Either CASES or PERCENT must be used.

Optional Identifiers:

BY vn vn

provides the list of BY variables. There can be up to 15 variables and they can be a mixture of numeric or character data types. There is no limit to the number of subgroups that these variables define. It is not necessary to sort or group the file on the BY variables. The power of the program lies in maintaining the proportionality of the various BY groups.

MAX.GROUP nn

specifies a maximum number to be taken from any one BY group. It must be a number greater than zero. If a group has less than this number, all its cases will be included. MAX.GROUP cannot be used with USE.WEIGHTS.

MIN.GROUP nn

specifies a minimum number to be taken from any one BY group. It must be a number greater than zero. MIN.GROUP cannot be used with USE.WEIGHTS. If both MAX.GROUP and MIN.GROUP are set to the same value, the CASES or PERCENT value does not matter.

OTHERS fn

specifies a name for the output file of all those cases that were not selected for the sample. The order of cases in this output file is the same as the input file order.

USE.WEIGHTS

causes the CASES or PERCENT value to be evaluated according to the sum of the input weights rather than the total number of cases.

WEIGHT vn

provides a weight variable. When there is a WEIGHT variable, the output sample weights will total the input weights. The ratio of the new sample weights is the same as the ratio of the WEIGHT variable in the selected cases.

CEILING (or CEIL)

If the value for the number of cases to be selected is fractional, use the next highest integer.

ROUND

If the value for the number of cases to be selected is fractional, use the integer nearest to the value. 4.5 would become 5. 4.4 would become 4 .

FLOOR

If the value for the number of cases to be selected is fractional, just use the integer part. However, the minimum number of cases is 1, even if the value is less than 1.