Chapter 1

# Using the SAS® System to Apply General Principles of Efficient Survey Research

## Introduction

As one of the more widely used general-purpose software products, the SAS System is the ideal tool to use in survey research applications. This book is designed for survey research practitioners who have some experience with basic SAS DATA step programming. Although these advanced beginners and intermediate-level SAS users are the primary audience for this book, experienced researchers and SAS programmers should also find helpful information on reporting survey results in tables and graphs that exemplify the principles of graphical excellence. The sophistication of the material ranges from basic to advanced, but we have always tried to respect the needs of new users.

The beauty of the SAS System is that it enables an intermediate-level user, who developed skill through on-the-job training or nonformal instruction, to perform complex data analyses, data/file manipulation, and report-writing tasks in less time than the formally trained computer science graduate would require using basic languages. Readers will be shown how to use SAS to manage a personalized survey research process, report the results in customized tables and graphs, and perform data and file manipulation tasks that are common to all survey research applications. The major contribution of this book is the integration of information culled from numerous SAS Institute publications, classic survey methodology, principles of graphical excellence, and many years of survey research experience into a single source targeted to the needs of the survey research practitioner. Some of the major topics addressed are

- Selecting random samples
- Generating ID numbers
- Storing samples in ASCII or in SAS data sets
- Merging files to combine institutional records with survey data

- Generating personalized letters and envelopes
- Tracking respondents and conducting follow-up mailings
- Performing basic statistical analyses
- Reporting results in custom-designed tables and graphs.

To illustrate the use of SAS in the survey process, we will show examples of survey questions, SAS code, and resulting output. In a few cases, the SAS code shown may not be the most elegant or efficient, as our intention is to enable researchers with basic to intermediate-level SAS skills to accomplish sophisticated survey computing tasks without programmer support. Given the importance of decisions often made on the basis of survey research findings, we think that it is critical for users to have the confidence in their work that is inspired by an intuitive understanding of the SAS code used to conduct the survey process and to report the findings. In today's computing and business environment, these objectives are far more important than programming efficiency and the use of CPU time.

## Resources/Manuals

As a comprehensive, flexible system, SAS can meet the computing needs of both novices and advanced users in providing solutions to virtually any research application such as:

- Data analysis
- Report writing
- Graphics
- Data manipulation
- File manipulation
- Data entry.

The power and flexibility of the SAS System is reflected in extensive documentation that is often intimidating to beginners. Listed below are useful SAS resources as well as resources written outside the Institute that cover conducting and reporting survey research.

- *Introduction to Market Research Using the SAS System*
- *SAS Language: Reference, Version 6, First Edition*

- *SAS Procedures Guide, Version 6, Third Edition*
- *SAS Language and Procedures: Usage 2, Version 6, First Edition*
- *SAS/STAT User's Guide, Version 6, Fourth Edition, Volume 1* and *2*
- *SAS/GRAPH Software: Reference,Version 6, First Edition, Volume 1* and *2*
- *SAS Guide to TABULATE Processing, Second Edition*
- *SAS/STAT Software: Changes and Enhancements through Release 6.12*
- SAS Technical Support on the World Wide Web: http://www.sas.com/service/techsup/intro.html
- *SAS Users Group International on the Internet and on Usenet*[1]
- *Mail and Telephone Surveys: The Total Design Method*, by Don A. Dillman, John Wiley & Sons, 1978.
- *How to Conduct Your Own Survey*, by Priscilla Salant and Don A. Dillman. John Wiley & Sons, 1994.
- The *Visual Display of Quantitative Information*, by Edward R. Tufte. Graphics Press, 1983.
- *Elementary Survey Sampling*, 2nd Edition, by Richard L. Scheaffer, William Mendenhall, and Lyman Ott. Duxbury Press, 1979.

## General Principles of Efficient Survey Practice

**Avoid asking respondents for information that is available through electronic records.** It is surprising how much this rule is violated, probably because the investigators are not using SAS or they lack the necessary skills or computer support to manipulate data files. In many survey projects, the samples are either drawn from or can be electronically linked to organizational databases (e.g., student records, personnel files, car registration records, membership files, etc.) that contain a wealth of demographic information to assist in the analysis of research questions. It is important to note that mail and telephone surveys, because of cost, time, and reliability factors, are usually the least desirable method of data collection. A survey should only be undertaken if the data cannot be obtained from alternative sources. The advantages of not asking respondents to report information that can be

[1] See Appendix A for A Beginners Guide to SAS-L and Appendix B for additional SAS World Wide Web Resources.

obtained from other sources are three-fold: shorter survey forms and less cost; increased response rates; and more accurate or complete information. Respondents may also have a greater feeling of confidentiality if they are not asked to report personal characteristics such as grade point average, salary, age, race, sex, length of service, etc. If available in electronic form, this type of information can be merged with survey responses using ID numbers to create a combined file for analysis. An exception to this principle is found in surveys of sensitive topics (e.g., drug use, personnel/program assessment) where respondent identification numbers are not used to ensure respondents that their responses are not only confidential but anonymous.

**Obtain high response rates by using a personalized survey design.** The personalization of mail surveys (cover letters with individual names in upper- and lowercase, correct use of Mr. or Ms., etc.) will greatly enhance the professional appearance of the survey product and significantly increase response rates. SAS functions and procedures are designed to facilitate the computer tasks required to achieve a high degree of personalization in the survey process with a minimal amount of effort. Dillman's (1978) work[2] remains a classic primer on the use of personalization techniques, question formatting, cover design, and the tricks of the trade in obtaining high response rates.

**Standardize procedures for conducting surveys.** The use of standardized procedures and SAS programs in implementing a personalized mail survey and conducting follow-up mailings will enable clerical staff to administer the survey process with minimal supervision. Procedures for creating and editing name and address files, generating cover letters, envelopes, mailing labels, and keeping track of survey respondents can be standardized for numerous survey projects. For surveys repeated over time, the survey process and the reporting of findings can become routine and far less time-consuming than most one-time data collection and analysis projects. SAS macro language commands can be used to make numerous update revisions to filenames and to TITLE statements in longitudinal analyses based on different data sets for each reporting cycle.

**Create camera-ready tables and graphs with SAS.** By creating final tables and graphs directly with SAS, you save a tremendous amount of labor and greatly reduce the probability of typographical errors in reporting the survey findings. SAS/GRAPH software can produce color or gray scale graphs on almost any type of printer or graphic device. Custom-designed statistical tables can be created directly with SAS using PROC TABULATE and other procedures. SAS output can also be downloaded to word processing and spreadsheet programs on a

[2] D. A. Dilllman, *Mail and Telephone Surveys: The Total Design Method* (New York: John Wiley & Sons, 1978).

microcomputer. We have found it to be much more efficient and rewarding to use the full capabilities of the SAS System to produce camera-ready output than to rekey SAS output into a word processor or graphics program.

## Overview of SAS Procedures Used in Survey Research

The SAS System contains numerous procedures for manipulating, analyzing, and reporting data. These procedures are called procs, and it is important to note that SAS procs perform operations on variables *across* observations or records. SAS functions, which are discussed in the next section, perform operations on variables *within* a record. This section will present a brief description of procs that are useful in virtually all survey research applications. Readers not familiar with some of these procs will find time spent reviewing the details in the *SAS Procedures Guide* to be well spent.

### PROC FREQ

PROC FREQ is probably the single most useful SAS procedure for reporting data. As the name implies, PROC FREQ generates frequency tables and cross-tabs. PROC FREQ can also be used to perform chi square tests of independence. A limitation of PROC FREQ is that it will produce lengthy output for multiple cross-tabulations. For example, income group by race by sex will produce a table (race by sex) and a separate page for each value of income group.

### PROC SORT

PROC SORT is used to order observations according to the values of a specified variable(s). The sort order can be *ascending* or *descending.* PROC SORT is useful for arranging data prior to printing and is often used to order files according to an identification number prior to merging with another file. PROC SORT can also be used to eliminate records with duplicate values of a specified variable.

### PROC PRINT

PROC PRINT is used to print a SAS data set. In survey applications, PROC PRINT is used to list names and addresses prior to generating personalized letters and envelopes. Additionally, PROC PRINT is often used as an edit procedure for printing observations that have unusually high/low or invalid values. Several SAS procedures create output data sets, and PROC PRINT must be used to view them.

### PROC MEANS

PROC MEANS provides basic descriptive statistics (mean, N, standard deviation, kurtosis, etc.). PROC SORT and PROC MEANS (with a BY statement) can be used to analyze variables within subgroups, e.g., income by race and sex.

## PROC UNIVARIATE

PROC UNIVARIATE is similar to PROC MEANS but provides a more comprehensive set of descriptive statistics. It is important to note that PROC UNIVARIATE is one of the few SAS procedures that will generate the median[3] statistic. PROC UNIVARIATE will also provide normality plots and perform a test of normality for the sample distribution of the data. Similar to PROC FREQ, PROC UNIVARIATE will produce lengthy output when numerous variables are specified or subgroup analyses are requested.

## PROC SUMMARY

PROC SUMMARY is very useful for producing descriptive statistics (sum, N, means, standard deviation) for analysis variables by one or more subgroups or classification variables in a condensed output. By default, PROC SUMMARY produces an output data set and *no* printed output. The PROC SUMMARY output file may be printed with PROC PRINT. PROC SUMMARY is very useful for complex data manipulations and for creating aggregated data sets for displaying data in graphical form.

## PROC TABULATE

PROC TABULATE is designed to produce customized tables and is *ideally* suited for reporting the results of survey research. PROC TABULATE will generate descriptive statistics[4] (sum, N, mean, standard deviation), percentages, and calculate row and column totals. PROC TABULATE provides intermediate-level users with total control over column and row formatting. This powerful SAS procedure will enable nonprogrammers to generate customized camera-ready output.

## PROC FORMAT

PROC FORMAT is one of the more powerful, but underused, procedures in the SAS System. It is primarily used to assign value labels in printed output to enhance the readability of a report. It can also be used to efficiently group variable values into a smaller number of categories. This type of data manipulation is usually done in the DATA step with IF, THEN, and ELSE statements, but it can actually be done more efficiently with PROC FORMAT.

## PROC CORR

In addition to basic descriptive statistics, PROC CORR provides a correlation (Pearson Product Moment, Spearman, Kendall, Hoeffding) matrix and descriptive statistics for each numeric variable that is included in the analysis. PROC CORR will also compute a reliability index (Cronbach's Coefficient Alpha) for a specified set of psychometric scale items.

[3] Medians can also be generated by PROC FREQ (base SAS) and PROC CAPABILITY (SAS/QC).

[4] PROC TABULATE does not compute medians in Version 6 of the SAS System. The median statistic is available in the TABULATE procedure in Version 7.

### PROC CHART/GCHART

PROC CHART and PROC GCHART generate vertical and horizontal histograms and pie charts. PROC CHART is part of the base SAS product and will not produce customized output of presentation quality. PROC GCHART is part of the SAS/GRAPH product and gives users full control over color, fonts, labeling, and choice of symbols and patterns.

### PROC PLOT/GPLOT

PROC PLOT and PROC GPLOT create line and scatter plots. PROC PLOT is part of the base SAS product and will not produce customized output. Like PROC GCHART, PROC GPLOT is part of the SAS/GRAPH product and provides users with full control over color, fonts, labeling, and choice of symbols.

### PROC CONTENTS

PROC CONTENTS is used to display the characteristics (variable names, data type, number of observations, variable length, date created, etc.) of a SAS data set.
SAS can be used to read and save ASCII files or SAS data sets. SAS data sets are especially useful for research applications as they have the following advantages:

- Self-documenting
- No input statement for reading file
- Reduced processing time and disk storage requirements.

### PROC RANK

PROC RANK is used for computing ranks for numeric variables. This procedure creates a new SAS data set.  PROC RANK can also generate normal scores. PROC RANK can be used for nonparametric statistical analyses and as a tool for selecting random samples (see Chapter 2).

## SAS Functions and Automatic Variables

SAS functions are preprogrammed routines for performing operations on variables *within* an observation. The procs that were reviewed in the previous section perform operations *across* observations. There are a number of SAS functions that are designed to manipulate character data and to perform mathematical operations that are essential to efficient survey research practice. The character manipulation functions (TRIM, CONCATENATE, SUBSTR) are often used to transform name and address variables to meet the data requirements for a personalized mail survey process. Examples of how these functions are used are presented in Chapter 3.

SAS automatic variables are created by the SAS System and can be used in the DATA step to control which observations are output. This section will describe the use of the IF FIRST.*varname* and IF LAST.*varname* as a method of eliminating observations with duplicate ID values.

## CONCATENATE: Combine Variables

The CONCATENATION function is used to combine two or more character variables into a single character string as a new variable. The concatenation operator symbols **||** are used to concatenate variables. In the example below, NAME1 is a new variable that is created by concatenating FNAME and LNAME with a blank space between the two variables.

```
data address;
   input  @1 fname  $10  @12 lname  $20.;
   name1 = fname||lname;
   cards;
John Smith
;
```

**Note:** The value of NAME1 is `John    Smith` The TRIM function described in the following section will remove the extra spaces between the first and last name.

## TRIM Function: Remove Trailing Blanks

TRIM is used to remove trailing blanks in a text string. Trailing blanks occur when a variable is defined as having a maximum of X characters, but a particular observation uses *less* than the X length or number of characters. SAS will pad the used length with blanks. To illustrate, in the preceding example, the variable FNAME is concatenated with LNAME and the resulting new NAME1 variable has six unneeded blank spaces between the two names. The unnecessary blanks are included in NAME1 because `John` is four characters long and the variable FNAME was defined in the INPUT statement to have a width of ten characters. The solution to the trailing blanks problem is the TRIM function.

```
name2 = trim(fname)||’ ‘||trim(lname);
```

**Note:** The value of NAME2 is `John Smith`.

## SUBSTR: Extract Part of a Variable

The SUBSTR function is used to extract a character variable into two or more separate elements. The SUBSTR function can also be used on numeric variables, but this is not recommended as the results can be unpredictable. To substring numeric variables, the best approach is to first use the PUT function (described later in this section) to convert a numeric variable to a character string and then to use SUBSTR. The form of the SUBSTR function is

```
newvar = substr(oldvar,x,y);
```

Where:

```
newvar = new variable to be created.

oldvar = previously defined variable.

x = starting position/character of OLDVAR to begin SUBSTR
  operation.

y = ending position/character of OLDVAR to end SUBSTR
  operation.
```

**SUBSTR Example**

```
yr_mo = ‘9701’;
                                          Result
year = substr(yr_mo,1,2);                   97
month = substr(yr_mo,3,4);                  01
```

## UPCASE/LOWCASE: Change Variable Casing

The SAS functions UPCASE and LOWCASE are used for reversing the case of character strings. The use of all uppercase text in survey communications is to be avoided as it gives the appearance of “junk mail.” However, it is not uncommon for survey researchers to have to work with names and addresses that are stored in electronic data files as all uppercase characters. When researchers are presented with all uppercase name and address text, these functions are useful tools for personalizing survey cover letters and envelopes (see Chapter 5). The UPCASE and LOWCASE functions can be used with SUBSTR to selectively modify the case of a character string. Additionally, these functions can be used to standardize the case of character variables that have been stored inconsistently.

**UPCASE/LOWCASE Examples**

```
make = ‘FORD’;
model = ‘taurus’;
name = ‘MCDONALD’;
                                                Result
make = LOWCASE(MAKE);                           ford
model = UPCASE(MODEL);                          TAURUS
substr(name,2,1) = lowcase(substr(name,2,1));   McDONALD
substr(name,4,5) = lowcase(substr(name,4,5));   McDonald
```

## PUT: Convert Numeric Variables to Character

The PUT function converts numeric variables to character strings. Numeric variables should be converted to character strings before concatenation or substringing operations are performed. PUT can also be used to create a new variable that takes the values defined in a FORMAT statement. This is useful for sorting the values according to the format label rather than the variable value. The PUT function always results in a *character* string. Character-to-numeric conversions can be handled with the INPUT function as discussed in the next section.

The form of the PUT function is

```
newvar = put(oldvar,format);
```

Where:

`newvar` = new character variable.

`oldvar` = existing numeric variable.

`format` = width of character variable to be created (e.g., $3.).

In the following example, YEAR is initially defined as a numeric variable and is redefined as the character variable YR_CHAR.

**PUT Example**

```
year = 98;
yr_char = put(year,$2);
```

## INPUT: Convert Character Variables to Numeric

The INPUT function converts character to numeric variables. In working with permanent SAS data sets, you may need to perform numerical operations on variables that have been defined and saved with character formats. The form of the INPUT function is

```
newvar = INPUT(oldvar, format);
```

Where:

*newvar* = new numeric variable.

*oldvar* = existing character variable.

*format* = width and decimal specification of numeric variable to be created (e.g. , 3.0).

In the following example, TEMP is defined as a character variable and redefined as a numeric variable TEMP_N.

**INPUT Example**

```
temp = '98.6';
temp_n = input(temp,4.1);
```

## SUM: Add Variable Values within an Observation

As the name suggests, the SUM function adds the values of two or more variables *within* an observation and stores them as a new variable. The SUM function is very useful in scoring questionnaire items and creating scale scores. This type of operation can also be done without the SUM function by using the SAS arithmetic operators (+ = / * -). The key difference between the two methods is in how they handle missing data. The SUM function will *ignore* missing data as it sums the remaining variables specified. Conversely, when the + operator encounters a missing value, the sum of the expression is set to missing. Note also that survey research projects nearly always have a considerable amount of missing data. Here is an example of how the SUM function is used to add the values for survey items within an observation:

```
scale1 = sum(q01,q02,q03,q04);
```

## MEAN: Average Variables within an Observation

The MEAN function calculates the arithmetic average of a specified list of variables *within* an observation. (Recall from the previous section that PROC MEANS computes descriptive statistics *across* observations.) The same operation can also be performed for each observation using the SAS + and / operators. The advantage of the MEAN function is that it automatically determines the appropriate denominator and adjusts the denominator for missing values. This is a very useful feature for processing survey data and eliminates the tedious coding that would be needed to perform these functions using SAS arithmetic operators. An example of the MEAN function follows:

```
avg1 = mean(q01,q02,q03,q04);
```

## RANUNI: Generate Random Numbers

Most mail and telephone surveys are based on random sampling procedures, and SAS has a number of powerful functions for generating random numbers. RANUNI is recommended as a good all-purpose random number generator. RANUNI must be supplied a seed value to generate random numbers. The seed can be a string of numbers, e.g., 12345 or the value 0. If a nonzero string is used, RANUNI can duplicate the same results in a later run if the source file has not changed. If the value 0 is used, the seed value is based on the clock time of your host computer and cannot be replicated in a later run. Here are two examples of using RANUNI for random sampling.

| | **RANUNI Example** |
|---|---|
| `if ranuni(12345) <= .5;` | takes an approximate 50% random sample of observations that are read into a DATA step. |
| `x = ranuni(12345);` | creates a variable named X that contains a randomly generated number for each observation that can be manipulated to select a random sample of observations (see Chapter 2). |

## IF FIRST.*varname*/IF LAST.*varname*: Eliminate Duplicate Records

It is not uncommon to work with data sets that have multiple records with duplicate ID values. IF FIRST.*varname* or IF LAST.*varname* can be used to select a single ID value based on the order specified. IF FIRST.*varname* and IF LAST.*varname* are SAS automatic variables that are created when a BY statement is used in the DATA step. Before a BY statement can be used in the DATA step, the file must be presorted by the variable specified in the BY statement. In the example to follow, the highest SAT score is selected for students who have taken the test more than once. You could use the NODUP option on PROC SORT, but you would not necessarily get the highest SAT for each student. The NODUP option on PROC SORT is useful when it doesn’t matter which duplicate record is selected.

### IF FIRST.varname/If LAST.varname Example

```
data scores;
   input @1 id $3. @5 yr_mo $4. @10 sat 4. ;
   cards;
001  9704   970
001  9803  1100
002  9803   950
002  9704  1000
003  9803  1300
003  9704  1200
004  9704   850
004  9803   970
;
proc sort data = scores; by id sat;

data highest;
   set scores; by id;
if last.id;

proc print data = highest;
title ‘Highest SAT Scores by ID’;
```

```
                             13:40 Tuesday, June 18, 1998
Highest SAT Scores by ID

   OBS     ID    YR_MO      SAT

   1      001     9803     1100
   2      002     9704     1000
   3      003     9803     1300
   4      004     9803      970
```

**Note:** IF FIRST.*varname* is used in exactly the same way as IF LAST.*varname* to remove duplicate observations. The only difference is in whether the first or last occurrence of a multiple ID value is output to the SAS data set created.

## ROUND: Eliminate Extra Decimals

The ROUND function is used to eliminate unnecessary decimals when printing custom reports. Most SAS analysis procedures will produce output with eight decimal places. For most types of survey data, this level of precision is artificial and too detailed for a custom report. For ordinal and interval type data, we think it is good practice to report only one decimal place for means or percentages. The ROUND function allows you to specify the number of decimal places to retain with appropriate rounding. You can modify an existing variable and keep the same name or create a new variable. Here is an example of the ROUND function.

**ROUND Example**

```
x = 4.14639
                              Result
x1 = round(x,1);              4
x1 = round(x,.1);             4.1
x1 = round(x,.01);            4.15
x1 = round(x,.0001);          4.1464
```